

Oracle 10g

sous Windows

Couvre aussi
la version gratuite
Oracle Express Edition !

Gilles Briard

Avec la collaboration de la société Digora



EYROLLES

R. BIZOI. – **Oracle10g – Administration.**

N°11747, 2005, 744 pages.

C. SOUTOU. – **SQL pour Oracle – 2^e édition.**

N°11697, 2005, 480 pages + DVD-Rom.

G. BRIARD. – **Oracle9i sous Windows.**

N°11220, 2003, 1040 pages + DVD-Rom.

G. BRIARD. – **Oracle9i sous Linux.**

N°11337, 2003, 894 pages (format semi-poche).

R. BIZOI. – **Oracle9i : SQL et PL/SQL.**

N°11351, 2003, 468 pages.

F. BERQUÉ, S. FREZEFOND, L. SORRIAUX. – **Java-XML et Oracle.**

E-Commerce – EAI – Portails d'entreprise – Applications mobiles.

N°9149, 2001, 650 pages + 2 CD-Rom.

B. VIDAL. – **Applications mobiles avec Oracle.**

N°9251, 2001, 685 pages + 2 CD-Rom.

C. SOUTOU. – **De UML à SQL. Conception de bases de données.**

N°11098, 2002, 450 pages.

M. HERNANDEZ, J. VIESCAS. – **Introduction aux requêtes SQL.**

N°9272, 2001, 436 pages + CD-Rom.

Oracle 10g

sous Windows

Gilles Briard

**Avec la collaboration
de la société Digora**

EYROLLES



ÉDITIONS EYROLLES
61, bd Saint-Germain
75240 Paris Cedex 05
www.editions-eyrolles.com

Avec la contribution de Emmanuel Puybaret pour la réalisation du CD-Rom.



Le code de la propriété intellectuelle du 1^{er} juillet 1992 interdit en effet expressément la photocopie à usage collectif sans autorisation des ayants droit. Or, cette pratique s'est généralisée notamment dans les établissements d'enseignement, provoquant une baisse brutale des achats de livres, au point que la possibilité même pour les auteurs de créer des œuvres nouvelles et de les faire éditer correctement est aujourd'hui menacée.

En application de la loi du 11 mars 1957, il est interdit de reproduire intégralement ou partiellement le présent ouvrage, sur quelque support que ce soit, sans autorisation de l'éditeur ou du Centre Français d'Exploitation du Droit de Copie, 20, rue des Grands-Augustins, 75006 Paris.

© Groupe Eyrolles, 2006, ISBN : 2-212-11707-8

À mes Parents,
à ma sœur,
à mon frère.
Créer, non posséder,
œuvrer, non retenir,
accroître, non dominer.

Lao-Tseu

Table des matières

Avant-propos	XLIII
Objectifs de ce livre	XLIII
Partie I : Oracle 10g sous Windows	XLVI
Partie II : Principes de fonctionnement d'Oracle 10g	XLVII
Partie III : Installer et paramétrer Oracle 10g sous Windows	XLVIII
Partie IV : Outils et langages de programmation	XLIX
Partie V : Administration d'Oracle 10g	L
Partie VI : Annexes	LII

PARTIE I

Introduction

CHAPITRE 1

Pourquoi choisir Oracle 10g sous Windows ?	3
Opposer Linux/Unix et Windows pour les applications d'entreprise : vraie guerre ou faux débat	4

Vers une nouvelle stratégie de distribution	5
Une couverture de l'ensemble des besoins de l'entreprise	6
Prendre en charge avec une même technologie des projets d'envergure différente	6
Oracle 10g : une seule souche technologique	7
Permettre une réelle concurrence	7
Quelles utilisations possibles d'Oracle 10g ?	8
Oracle 10g : serveur de données	8
Pourquoi pas une base de données type Microsoft Access ?	9
Pourquoi pas une base de données « libre » ?	10
Les applications client-serveur	10
L'intégration de Windows dans le système d'information	11
Partition des applications	11
La réplication des données	12
Infocentres, datawarehouses et data marts	13
Serveur Internet/intranet	14
Architecture à trois niveaux	16
Et la technologie Java ?	17
Java et PHP, quel est le choix d'Oracle ?	17
Quelle version d'Oracle choisir pour Windows ?	18
Que contiennent les versions d'Oracle 10g pour Windows ?	18
Résumé	19

CHAPITRE 2

Les nouveautés d'Oracle 10g	21
Les apports d'Oracle 10g	21
Le Grid Computing	22
Les options d'automatisation	23
Vers une maintenance simplifiée (Oracle 10g)	23
ASM : Automatic Storage Management (Oracle 10g)	23
AWR : Automatic Workload Repository (Oracle 10g)	23
AMT : Automatic Maintenance Task (Oracle 10g)	24
ADDM : Automatic Database Diagnostic Monitor (Oracle 10g)	24

AMM : Automatic Memory Management (Oracle 10g)	24
STA : SQL Tuning Adviser (Oracle 10g)	24
Server Generated Alerts (Oracle 10g)	25
Installation d'Oracle 10g	25
Oracle 10g : une installation plus facile (Oracle 10g)	25
Nouvelles règles OFA (Oracle 10g)	25
Le moteur Oracle 10g	25
Nouveau tablespace SYSAUX (Oracle 10g)	26
Nouveau mode de gestion des tablespaces (Oracle 10g)	26
Allocation automatique des segments de stockage (Oracle9i)	26
Les tablespaces temporaires (Oracle9i)	26
Les tablespaces UNDO ou d'annulation (Oracle9i)	27
La période de rétention d'un tablespace UNDO (Oracle9i)	27
Réorganisation des objets base ouverte (Oracle9i)	27
Extension automatique d'un fichier de données (Oracle9i)	27
Les tablespaces transportables (Oracle9i)	27
Une base peut comporter plusieurs DB_BLOCK_SIZE (Oracle9i)	28
L'exploration des fichiers redo-log : le Log Miner (Oracle9i)	28
La gestion automatique des fichiers par Oracle (Oracle9i)	29
La suppression automatique des fichiers (Oracle9i)	29
Gestion du fichier d'initialisation (Oracle9i)	29
Valeurs dynamiques du fichier d'initialisation (Oracle9i)	29
Contrôle dynamique de la mémoire SGA (Oracle9i)	30
Nouvelle syntaxe pour démarrer une instance (Oracle9i)	30
Disparition de Server Manager (Oracle9i)	30
Disparition du CONNECT INTERNAL (Oracle9i)	30
Utilisation d'un fichier mot de passe (Oracle9i)	30
Modification du moteur Java (Oracle9i)	31
Les langages de développement	31
Nouveautés XML d'Oracle 10g	31
Nouveautés PL/SQL (Oracle 10g)	31
Nouveautés SQL : les expressions régulières (Oracle 10g)	31
Les nouveaux packages PL/SQL (Oracle 10g)	31
Les outils et assistants	32
Oracle Application Express (Oracle 10g)	32
Data Pump Export et Import (Oracle 10g)	32

Création d'une base Oracle (Oracle 10g)	32
Paramétrage d'Oracle Net (Oracle9i)	32
OEM : Administrer votre environnement Oracle (Oracle 10g)	33
Résumé	33

CHAPITRE 3

Comparatif technique Windows/Unix	35
Mise en place d'Oracle 10g	35
Création d'une base Oracle	38
Contrôle des instances par le système d'exploitation	40
Résumé	41

CHAPITRE 4

Environnement Windows : l'indispensable	43
La base de registre	43
Principes de la base de registre	44
Modifier une valeur de la base de registre	45
Les services	46
Principes des services Windows	46
Démarrer et arrêter un service Windows	47
Les outils standards de Windows	47
Explorateur Windows	47
Le Gestionnaire des tâches	49
Microsoft Management Console	50
L'Analyseur de performances	50
La gestion de l'ordinateur	51
Administrateur de disques	52
L'Observateur d'événements	52
Terminal Server	53
Regedit	53
Paramètres régionaux, date et heure	53
Les outils complémentaires	53
Process Viewer	54
Les autres utilitaires	54
Résumé	54

PARTIE II

Principes de fonctionnement d'Oracle 10g

CHAPITRE 5

Fonctionnement d'Oracle 10g sous Windows	57
Oracle 10g : une seule version	58
Les composants d'Oracle 10g sous Windows	58
Les composants « fichiers »	58
Les composants « programmes »	59
Les composants « mémoires »	60
Base de données et instance	61
Architecture interne d'Oracle 10g sous Windows	62
L'exécutable oracle.exe	63
Les différents threads d'oracle.exe	63
Les « sessions utilisateurs » d'Oracle	64
Oracle et la base de registre	65
L'utilisation du registre par Oracle	66
Démarrer et arrêter Oracle sous Windows	67
Quels utilitaires pour quelle tâche ?	67
Démarrer et arrêter Oracle sous Windows	68
Démarrer le service Windows OracleServiceSID	69
Démarrer l'instance Oracle	73
Arrêter l'instance Oracle	77
Arrêter le service Windows	78
Comment associer les threads aux process Oracle ?	79
L'utilitaire ORADIM	82
Comment déplacer une base Oracle existante vers une autre machine ? ...	86
Oracle Administration Assistant for Windows	86
Configurer les comptes Windows	87
Configurer le démarrage de l'instance Oracle	88
Définir la base par défaut et paramétrer les variables de l'Oracle Home ...	88
Résumé	89

CHAPITRE 6

Les clusters Oracle	91
Historique et enjeux des technologies cluster	91
Principes d'une architecture cluster Windows	92
Les deux types de clusters proposés par Oracle	93
Objectifs de Fail Safe et Real Application Server	93
Oracle Fail Safe	94
Oracle Real Application Clusters	95
Oracle et les serveurs multiprocesseurs	96
Quels liens avec Microsoft Cluster Server (MSCS) ?	97
Prérequis techniques pour Oracle Fail Safe et Oracle Real Application Clusters	97
Comparaison d'Oracle Fail Safe et d'Oracle Real Application Server ...	98
Fonctionnement d'Oracle Fail Safe	98
Cas d'utilisation d'Oracle Fail Safe	98
Principe de fonctionnement	99
Que deviennent les utilisateurs connectés si le serveur s'arrête ?	101
Fonctionnement d'Oracle Real Application Clusters	101
Cas d'utilisation d'Oracle Real Application Clusters	101
Principe de fonctionnement	102
Que deviennent les transactions en cours sur le nœud qui « tombe » ? ...	104
Le basculement vers un nœud survivant est-il automatique ?	104
La connexion est-elle automatique et le « contexte d'utilisation » est-il sauvegardé ?	104
Résumé	104

CHAPITRE 7

Les processus d'Oracle 10g	107
Les threads indispensables, optionnels et utilisateur	107
Les threads indispensables	108
Le threads DBWR	108
Les threads LGWR et CKPT	111
Le thread System Monitor	111
Le thread Monitor	112

Les threads optionnels	112
Le processus listener Oracle Net	113
Le thread ARCH	113
Le thread RECO	113
Le thread SNPnn	114
Les threads Dnnnn et Snnnn	114
Le thread Pnnnn	114
Modes single-task et two-tasks	115
Les threads utilisateur	117
Configuration sans réseau	117
Configuration réseau en mode client-serveur	118
Configuration MTS (Multi-Thread Server)	120
Résumé	121
CHAPITRE 8	
L'utilisation de la mémoire par Oracle 10g	123
Mémoire physique et mémoire virtuelle	123
Mémoire virtuelle sous Windows	124
Les zones mémoire utilisées par Oracle 10g	125
Les paramètres statiques et dynamiques	125
La zone SGA (System Global Area)	125
Le cache des blocs de données	127
Les paramètres de base et les paramètres avancés	128
Le tuning automatique de la SGA	129
Les buffers redo-log	130
La Shared Pool	130
La Java Pool	131
La zone PGA (Program Global Area)	131
Résumé	132
CHAPITRE 9	
Les fichiers d'une base Oracle 10g	133
Les fichiers Oracle 10g et leur rôle	134
Les logiciels Oracle	134
Les fichiers de données	134
Les fichiers redo-log	136

Les fichiers redo-log archivés	137
Les fichiers de contrôle	138
Le fichier d'initialisation	138
Les fichiers d'initialisation d'Oracle Net	141
Le fichier d'alerte de la base	142
L'observateur d'événements	143
Les fichiers trace d'Oracle Net	144
Les fichiers trace des utilisateurs	144
Faire coexister plusieurs versions Oracle	145
Qu'est-ce qu'un Oracle Home ?	145
L'architecture OFA proposée par Oracle	147
Bénéfices procurés par OFA	147
Caractéristiques d'une installation OFA	148
ORACLE_BASE et ORACLE_HOME	149
Où placer les fichiers des bases de données ?	149
Séparer les segments d'utilisations différentes	150
Comment nommer les fichiers des bases de données ?	150
Où placer les fichiers d'administration des bases ?	151
Avantages d'OFA	152
OFA : schéma d'installation des fichiers Oracle sous Windows	153
Résumé	156

CHAPITRE 10

Démarrer et arrêter une base Oracle 10g	157
Étapes d'installation d'Oracle 10g	158
S'assurer des prérequis techniques	158
L'installation	158
Après l'installation	158
Étapes d'une création de base	158
Le rôle du service OracleServiceSID	158
Étapes préliminaires à la création d'une base	159
Étapes postérieures à la création d'une base	159
Base de données et instance	159
Les privilèges administrateur	160
Le fichier d'initialisation	162

Démarrer une base Oracle 10g	162
Démarrage du service OracleServiceSID	162
Démarrage d'une instance	164
Montage de la base de données	164
Ouverture de la base de données	164
Démarrage en un seul ordre	164
Arrêter une base Oracle 10g	166
Les différents types d'arrêts	166
Le paramétrage du service OracleServiceSID	168
Fermeture de la base de données	169
Démontage de la base de données	169
Arrêt de l'instance	169
Fermeture en un seul ordre	169
Résumé	170
 CHAPITRE 11	
Transactions et accès concurrents	171
Les transactions	172
Notion de transaction	172
Début et fin d'une transaction	173
Les savepoints	173
Les segments d'annulation	174
Principe d'un segment d'annulation	174
Segment UNDO et rollback segment	174
Les lectures consistantes	175
La pose de verrous	177
Résumé	178
 CHAPITRE 12	
Le dictionnaire de données Oracle 10g	179
Le dictionnaire Oracle 10g	179
Création du dictionnaire Oracle 10g	180
Ajout de produits optionnels au dictionnaire	181
Les différents types de vues	182
DBA : les vues de l'administrateur Oracle 10g	182
ALL : les vues sur tous les objets accessibles	182

USER : les vues sur les objets de l'utilisateur	182
V\$: les vues dynamiques de la base	182
GV\$: les vues des bases en cluster	182
Les autres vues	183
Comment identifier la vue que vous cherchez ?	183
Comment obtenir la description des vues que vous cherchez ?	184
Liste commentée des vues	185
Résumé	198

PARTIE III

Installer et paramétrer Oracle 10g sous Windows

CHAPITRE 13

L'installation d'Oracle 10g sous Windows	201
Les versions d'Oracle Database 10g	202
Les différents CD-Rom disponibles	203
Utilisation de l'aide Oracle	204
La documentation sur CD-Rom	204
Le support technique	204
Les autres documentations	204
Les étapes de l'installation d'Oracle 10g	205
Étape 1 : vérification des prérequis	205
Niveau matériel requis	206
Espace disque requis	207
Versions Windows supportées par Oracle Database 10g	207
Environnement Java nécessaire	207
Environnement réseau indispensable	208
Vérification de la configuration réseau du serveur	208
Étape 2 : adapter et préparer Windows	208
Le respect des règles OFA (Optimal Flexible Architecture)	209
Étape 3 : installation d'Oracle 10g pour Windows	210
Utilisation du DVD fourni	210
Comment télécharger Oracle 10g ?	210
Installer les logiciels Oracle	210

Installation pas à pas des logiciels Oracle Database 10g	212
Connexion à la base initiale à l'aide de SQL*Plus	226
Vérification d'Oracle iSQL*Plus	227
Vérification d'Oracle Enterprise Manager	227
Les défauts de la base initiale	229
Si l'installation échoue	229
Automatiser l'installation	230
Conclusion sur cette étape	230
Étape 4 : post-installation	231
Les autorisations d'accès aux fichiers et aux répertoires	231
Lancer des travaux à partir du Job Scheduler	231
Les variables « par défaut »	231
Automatiser le démarrage et l'arrêt des bases Oracle	232
Automatiser les sauvegardes	233
Résumé de l'installation	233
CHAPITRE 14	
Création d'une base Oracle 10g	235
Pourquoi créer une nouvelle base de données ?	235
Une nouvelle application	235
La puissance de votre serveur	236
La nécessité de tester	236
Des applications différentes	236
Étapes préliminaires	236
Étapes de la création d'une base Oracle 10g	238
Les nouveautés apportées par Oracle Database Configuration Assistant .	238
Les étapes effectuées par DBCA	238
L'utilisation de DBCA	239
Le fichier d'initialisation	248
Point sur la création de la base	253
Les fichiers composant la base	253
L'étape finale	262
Les ordres de création d'une base Oracle 10g	264
Le fichier de lancement	264
Le fichier d'ordres SQL	265

Le fichier de création de la base	267
Les fichiers complémentaires de création de la base	269
Les scripts SQL complémentaires	270
L'étape de postcréation	270
Le fichier d'initialisation de la base	271
Création de la base	273
En cas de problème	273
Les étapes postérieures à la création de la base	274
La gestion des droits d'accès aux répertoires NTFS	274
La gestion des droits d'accès à la base de registre	274
Configurer les options d'Oracle	275
Configurer Oracle Net	275
Accéder aux caractéristiques de la nouvelle base	275
Démarrer automatiquement la base	276
Intégrer la nouvelle base dans les sauvegardes	276
Tester l'accès à la base	276
Travailler sur la base	276
Résumé de l'installation	277

CHAPITRE 15

Oracle Net, le middleware Oracle	279
Instant Client, la nouveauté d'Oracle 10g	279
Qu'est-ce que le middleware Oracle Net ?	280
Cas des serveurs Web connectés à Oracle	283
Comment établir le dialogue entre plusieurs serveurs Oracle ?	284
Oracle Net	285
Le Listener Oracle	286
Principe d'une connexion Oracle Net	286
Fonctionnement d'Oracle Net avec un firewall ou un Proxy	287
Administration du Listener	288
Les assistants Oracle Net : netca et netmgr	288
Configuration du Listener	289
Configuration d'Oracle Net avec netca	290
Première étape : configuration du Listener	290

Deuxième étape : méthode de la résolution de noms	294
Troisième étape : configuration des alias	295
Configuration d'Oracle Net avec <i>netmgr</i>	300
Démarrer/arrêter le Listener	304
Vérifier la présence du Listener sous Windows	307
Contrôler le fichier log du Listener	307
Lancer le Listener Oracle Net au démarrage du serveur	307
Création de nouveaux alias	307
Où placer les fichiers tnsnames, sqlnet et listener ?	308
Tester une connexion Oracle Net	308
Tester Oracle Net en local	308
Tester Oracle Net en client-serveur	308
Oracle 10g et le Multithreading	309
Coexistence d'Oracle Net et de Net8	309
Oracle Net, un middleware intelligent	309
Comment identifier les sessions connectées ?	310
Résumé	310

CHAPITRE 16

Accéder à Oracle depuis Windows	311
Les différents accès	312
Les pilotes génériques	312
Les pilotes natifs Oracle	315
Oracle Objects for OLE (OO4O)	315
Les accès .NET	316
Oracle Call Interface (OCI)	317
L'accès à une base Oracle distante	318
Accéder à Oracle depuis un serveur Web	318
Oracle et Microsoft Transaction Server (MTS)	320
Quel accès choisir ?	320
Migrer des données provenant d'Access, Excel vers Oracle 10g	321
Oracle COM Automation	321
Résumé	324

PARTIE IV

Outils et langages de programmation

CHAPITRE 17

Programmer avec PL/SQL	327
Les avantages de PL/SQL	327
Survol de PL/SQL	329
Une structure de « bloc »	330
Les variables PL/SQL	330
Les structures de contrôle	330
Les curseurs	331
La modularité	331
Les attributs	331
L'ouverture	331
Un accès aux données relationnelles-objet	332
Le traitement des erreurs	332
Les exemples de programmes fournis par Oracle	332
Architecture de PL/SQL	333
Développer avec PL/SQL	334
Fonctionnement de PL/SQL	334
Une structure de « bloc »	335
Exécution d'un programme PL/SQL avec SQL*Plus et iSQL*Plus	337
Les variables PL/SQL	337
Les structures de contrôle	341
Les curseurs	344
Les curseurs explicites	345
Les étapes de la vie d'un curseur	345
Simplification d'écriture des curseurs	351
Simplification d'écriture des curseurs : FOR... LOOP	353
Simplification d'écriture des curseurs : FOR... IN	354
Les curseurs paramétrés	355
Mettre à jour des données avec un curseur : CURRENT-OF	355
Gestion des erreurs	356
La section EXCEPTION	356

Procédures et fonctions	358
Qu'est-ce qu'un sous-programme ?	358
Avantages d'un sous-programme	359
Les procédures	359
Les fonctions	360
Création, suppression des procédures et fonctions	361
Utilisation de SQL*Plus avec les procédures et les fonctions	361
Création d'une fonction comportant une erreur	361
Compilation d'une fonction sans erreur	362
Exécution d'une fonction compilée sans erreur	363
Affichage de variables contenues dans un bloc PL/SQL	363
Retrouver la liste des sous-programmes créés dans une base	364
Retrouver le code d'un sous-programme créé dans une base	364
Les packages	365
Qu'est-ce qu'un package ?	365
Appel d'un objet d'un package	365
Les packages PL/SQL standards	366
Comment créer ces packages ?	367
Les fonctions standards : DBMS_STANDARD	367
Émettre des messages d'un bloc PL/SQL : DBMS_OUTPUT	367
Écrire et lire dans des fichiers : UTL_FILE	368
Envoyer des e-mails depuis Oracle : UTL_MAIL	370
Soumettre des travaux : DBMS_JOB	371
Accéder aux données d'un annuaire LDAP : DBMS_LDAP	371
Travailler avec des données binaires : DBMS_LOB	372
Générer des scripts de création des objets : DBMS_METADATA	376
Construire dynamiquement des ordres SQL : DBMS_SQL	377
Communiquer entre processus : DBMS_ALERT et DBMS_PIPE	379
Oracle Advance Queuing : DBMS_AQ	380
Oracle Log Miner : DBMS_LOGMNR	380
Récupérer le schéma des objets : DBMS_METADATA	381
Un regroupement d'utilitaires : DBMS_UTILITY	381
Manipuler le XML : DBMS_XML	381
Manipuler des données géolocalisées : SDO	381
Les triggers	382
Principes d'utilisation des triggers	382
Exemple d'écriture d'un trigger	382
Comment créer et gérer un trigger ?	384

Quand se déclenchent les triggers ?	384
La clause FOR EACH ROW	385
Les conditions d'exécution	385
Accéder aux valeurs d'un enregistrement dans un trigger	385
Peut-il y avoir plusieurs triggers sur une table ?	386
Comment visualiser les triggers portant sur une table et leur code ?	386
Résumé	386

CHAPITRE 18

Les outils d'Oracle 10g	387
Présentation des outils d'Oracle 10g	387
Oracle Export / Oracle Import	388
Principes de l'Export/Import	389
L'Export/Import est-il un outil de sauvegarde ?	390
Où se situe le jeu d'essai de l'Export/Import ?	390
Configuration préalable	390
Compatibilité des Export/Import avec les précédentes versions d'Oracle ..	391
Que contient un fichier export ?	391
Comment transférer des fichiers Export ?	391
Comment lancer un Export/Import ?	392
L'Export/Import en mode incrémental	392
Les options d'un Export	392
Les possibilités de l'Export en mode USER	395
Les possibilités de l'Export en mode TABLE	395
Qui peut réaliser une exportation complète ?	395
Dans quel ordre sont exportés les objets ?	395
Qu'est-ce que l'ARRAY FETCH ?	396
Utilisation des paramètres BUFFER et RECORDLENGTH	396
Comment sont exportés les champs de type LONG ?	397
Comment utiliser un fichier de commande ?	397
Comment utiliser l'option QUERY ?	397
Comment limiter la taille d'un fichier export ?	398
Comment récupérer les erreurs rencontrées lors d'un export ?	398
Les options d'un Import	398
Quelle action réalise le paramètre DESTROY ?	400
Comment créer le clone d'une base ?	401
Peut-on remplacer des données existantes par celles issues d'une importation ?	401

Quelles sont les conséquences d'une importation pour les objets propriétés de SYS ?	401
Les mots de passe des utilisateurs sont-ils automatiquement exportés/importés ?	401
Exporter tous les utilisateurs, est-ce identique à un Export en mode FULL ?	402
Comment sont importées les contraintes d'intégrité référentielle ?	402
Peut-on changer le nom d'une table ?	402
Quelle répercussion à l'Import sur les index, contraintes et triggers d'une table ?	402
Comment sont gérés les droits et les synonymes ?	402
À quoi sert l'option COMPRESS=Yes ?	403
Comment diminuer la taille d'une table ou d'un index ?	404
Comment récupérer les ordres SQL de création de tous les objets d'une base ?	404
Si une table et son index ont des propriétaires différents, que se passe-t-il ?	404
Qu'en est-il des tablespaces OFFLINE ?	404
Les informations du tablespace SYSTEM sont-elles automatiquement exportées ?	405
Comment connaître le résultat d'un Import lancé sous at?	405
Quelles sont les erreurs rencontrées ?	405
Oracle Data Pump	405
Les nouveautés apportées par Data Pump	405
Les différences entre Data Pump et Export/Import	406
Comment relancer Data Pump Export	406
Comment relancer Data Pump Import	408
Utiliser Data Pump pour récupérer le code d'une procédure	409
Quelles sont les autres possibilités de Data Pump ?	409
Oracle SQL*Loader	409
Quelle utilisation pour SQL*Loader ?	409
Comment fonctionne SQL*Loader ?	410
Où se situe le jeu d'essai de SQL*Loader ?	410
Comment utiliser SQL*Loader ?	410
Comment générer les fichiers log, bad, discard ?	411
Les données à insérer peuvent-elles être de longueur variable ?	412
Les données à insérer peuvent-elles être de longueur fixe ?	412
SQL*Loader crée-t-il la table dans la base Oracle 10g ?	413
Quels liens existe-t-il entre la table et le fichier de contrôle ?	413
Comment préciser le type des données en entrée ?	414
Comment charger des données qui ne sont pas codées en ASCII ?	415
Peut-on modifier les données lors de leur insertion ?	415

Peut-on insérer des données dans plusieurs tables ?	416
Comment charger plus vite avec SQL*Loader ?	416
Quelle relation entre SQL*Loader et les index ?	416
Comment forcer SQL*Loader à effectuer un commit à la fin du chargement ?	417
Comment sont utilisés les rollback segments ?	417
Que contient le fichier log, résultat d'un chargement ?	417
Oracle SQL*Plus	419
Pourquoi administrer Oracle avec SQL*Plus ?	419
Les deux modes de connexion à SQL*Plus	420
Connexion à SQL*Plus en mode « administration »	420
Connexion à SQL*Plus en mode « utilisation »	421
Interactions entre SQL*Plus et son environnement	422
Les différentes commandes de SQL*Plus	423
Installation d'un jeu d'essai	423
Chargement de l'aide en ligne SQL*Plus	424
Que signifie le message « Product User Profile is not loaded » ?	424
Les ordres SQL et les commandes SQL*Plus	424
Utilisation de SQL*Plus avec PL/SQL	425
Utiliser un éditeur de texte pour modifier vos ordres SQL	425
Créer un fichier de commandes SQL	426
Exécuter un fichier de commande à partir de SQL*Plus	426
Mesurer la durée de vos ordres SQL	427
Exécuter une commande OS	427
Interrompre un ordre SQL	428
Inclure des commentaires	428
Écrire des commandes interactives avec SQL*Plus	428
Décrire la structure des tables et des vues	429
Améliorer la présentation de vos résultats	429
Création de requêtes dynamiques	433
Autre requête dynamique	435
Les outils complémentaires	437
Quelle complémentarité attendre d'Oracle Enterprise Manager ?	437
Les limites d'Oracle Export/Import ?	437
Les limites d'Oracle Data Pump ?	438
Les limites d'Oracle SQL*Loader ?	438
Quelles sont les limites d'Oracle SQL*Plus ?	438
Quels autres outils pour quels besoins ?	438
Résumé	439

CHAPITRE 19

Pilotes JDBC et SQLJ	441
Principe des pilotes JDBC	442
JDBC : les pilotes et leur utilisation	443
Comment choisir entre les différents JDBC existants	444
Pilote JavaSoft (ou autre) utilisant ODBC	444
Pilote OCI s'appuyant sur Oracle Net	445
Pilote téléchargeable Thin JDBC	446
Pilote JDBC intégré à la JVM Oracle 10g	447
SQLJ	448
SQLJ et JDBC	449
écrire un fichier SQLJ	453
Traduire et compiler un fichier .sqlj : l'utilitaire SQLJ	454
SQLJ côté serveur	454
JDBC 2.0	457
Mise à jour en batch	457
Les ResultSet parcourables	459
DataSource	461
Le pool de connexions	464
Connection Caching	466
Les transactions distribuées	468
JDBC 3.0	471
Transaction Savepoints	471
Passage entre transactions locales et globales	472
Statement caching	473
Support du JDK 1.4	474
Résumé	474

CHAPITRE 20

Java et Oracle 10g	475
L'environnement Java d'Oracle 10g	476
Évolutions d'Oracle 10g	476
Configuration Java dans Oracle 10g	477
Paramètre SHARED_POOL_SIZE	477
Paramètre JAVA_POOL_SIZE	478

Paramètre JAVA_SOFT_SESSIONSPACE_LIMIT	478
Paramètre JAVA_SOFT_SESSIONSPACE_SIZE	478
Sécurité Java dans Oracle 10g	478
Gérer la sécurité Java 2 dans Oracle 10g	479
Sécurité Oracle 10g préconfigurée	479
Gérer les permissions Java 2 dans Oracle 10g	480
Les objets Java dans Oracle 10g	482
Objets Java stockés dans la base de données	482
Déployer des programmes Java dans la base Oracle	483
Gestion des références entre objets Java	484
Contexte d'exécution d'une classe Java	484
Compiler une classe Java	485
Charger des objets Java avec l'utilitaire loadjava	485
Supprimer des objets Java avec l'utilitaire dropjava	489
Charger des objets Java en PL/SQL	489
Supprimer des objets Java en PL/SQL	490
Options de compilation par défaut	490
Consulter les objets Java stockés dans Oracle 10g	490
L'interface shell ojvmjava	491
Compilation native des classes Java dans Oracle 10g	492
Configuration de la compilation native Oracle 10g	493
Lancer la compilation native avec l'utilitaire Oracle ncomp	494
Déployer des bibliothèques dynamiques de classes Java avec l'utilitaire deploync	497
Vérifier la compilation native avec l'outil statusnc	498
Les procédures stockées Java dans Oracle 10g	499
Méthodes Java et procédures stockées Java	499
Écrire une procédure stockée Java	500
Déployer la procédure stockée Java	501
Publier manuellement une procédure stockée Java	501
Appeler une procédure stockée Java	507
Appeler une méthode Java depuis SQL*Plus	507
Appeler une méthode Java depuis un déclencheur (trigger) de base de données	507
Appeler des fonctions Java depuis des instructions de manipulation de données SQL	509
Appeler des méthodes Java depuis des blocs PL/SQL	510
Appeler des sous-programmes PL/SQL depuis des méthodes Java	512
Résumé	513

CHAPITRE 21

XML et Oracle 10g	515
À propos d'XML	515
Introduction	515
Structure d'un document XML, les DTD et XML Schéma	517
Construction d'un document XML avec une DTD ou un schéma XML ..	520
Présentation et affichage des données XML avec les feuilles de style ...	520
Utilisation d'XML	521
Interrogation de documents XML : Langage W3C XML Path (XPath) ..	523
Les produits Oracle XML	525
Stockage XML dans le référentiel Oracle XML DB	525
Installation d'Oracle XML DB	525
Configuration d'Oracle XML DB	527
Gestion d'arborescence	531
Gestion de version	533
API PL/SQL de gestion des ressources XML DB	535
API Java/JNDI de gestion des ressources XML DB	536
Sécurité des ressources XML DB	538
Accès au référentiel XML DB par, FTP, HTTP et WebDAV	540
Schéma global d'architecture XML DB	543
Stockage XML traditionnel dans Oracle 10g	543
Stocker un document XML en tant que document unique	544
Stocker un document XML en tant que données	548
Rechercher dans un document XML	553
Recherche par Oracle Text classique avec l'index Context	554
Recherche par Oracle Text avec le nouvel index CTXXPATH	557
Transformation et validation d'instances XML Type	558
XSLT Processor	558
Transformation d'instance XML Type	560
Validation d'instance XML Type	560
Traitement XML : XML Parser v2 et API XML Type	561
Interfaces DOM et SAX	561
Exemple : XML Parseur	564

Nouvelle API PLSQL de Manipulation XML	567
Nouvelle API Java de manipulation XML	570
XML Class Generator pour Java	574
Générer des classes Java à partir d'une DTD	575
Créer un document valide XML à partir de classes	578
XML SQL Utility pour Java	582
Générer un document XML à partir du résultat d'une requête	582
Écrire du code XML dans une base de données	586
XDK JavaBeans	588
DOMBuilder Bean	589
XSL Transformer Bean	589
XMLSourceView Bean	589
XMLTransformPanel Bean	589
DBViewer Bean	589
DBAccess Bean	589
XMLDiff Bean	590
XSQL Servlet	590
Résumé	591

CHAPITRE 22

Présentation d'Oracle Application Express	593
Introduction à Oracle Application Express	593
Quel est le positionnement d'Oracle Application Express ?	594
Quelles sont les applications cibles ?	595
Les sources d'information	596
L'architecture d'Oracle Application Express	597
L'architecture technique	598
Configuration requise	599
Les composants d'Oracle Application Express	599
Application Builder	600
SQL Workshop	601
La gestion des droits d'accès d'Oracle Application Express	601
Résumé	602

PARTIE V

Administration d'Oracle 10g

CHAPITRE 23

Gestion de l'espace disque et des fichiers	605
Les apports d'Oracle Enterprise Manager	606
Gestion de l'espace disque	606
Base de données physique et logique	606
Principe du tablespace	606
Le DB_BLOCK_SIZE	611
Segments, extents et blocs	611
Les différents types de fichiers	613
La gestion des tablespaces en mode LOCAL	613
Différences entre la gestion locale et dictionnaire	614
Les options des tablespaces en gestion locale	614
Comment déterminer le mode de gestion des tablespaces ?	616
La gestion des objets des tablespaces en mode DICTIONARY	616
Création d'un tablespace en mode DICTIONARY	617
La clause STORAGE	617
Que se passe-t-il lorsque aucune clause STORAGE n'est définie ?	618
Comment contrôler le nombre d'extensions d'un segment ?	619
Modifier les caractéristiques de la clause STORAGE	621
La fragmentation est-elle pénalisante ?	622
Défragmenter une table	622
Défragmenter un index	624
Gestion des tablespaces et des fichiers de données	624
Création d'un tablespace	624
Définir un tablespace par défaut	624
Les tablespaces temporaires	625
Les tablespaces UNDO ou d'annulation	626
Les tablespaces BIGFILES	626
Modification d'un tablespace	627
Renommer un tablespace	627
Ajouter un fichier de données	627
Compacter et réduire les segments en ligne	628
Extension automatique d'un fichier de données	629

Les alertes d'utilisation des tablespaces	630
Estimation de la taille d'un nouveau segment	630
Liens entre fichiers et tablespaces	631
Caractéristiques des tablespaces et fichiers associés	631
Positionner un tablespace online/offline	633
Déplacer/renommer les fichiers d'un tablespace	634
Fragmentation des tablespaces	634
Modifier la taille d'un fichier	635
Supprimer un tablespace	635
Les tablespaces transportables	636
Les tablespaces READ-ONLY	636
Gestion des tablespaces d'annulation	636
Les nouveautés des tablespaces UNDO	636
La création d'un tablespace UNDO	637
La période de rétention d'un tablespace UNDO	638
Actions possibles sur un tablespace UNDO	638
La gestion des rollback segments	638
Création d'un rollback segment	639
Visualiser les rollback segments	639
Suppression d'un rollback segment	640
Nombre de rollback segments à créer	640
Gestion des fichiers redo-log	640
Création d'un fichier redo-log	640
Caractéristiques des fichiers redo-log	641
Basculer la base en mode ARCHIVELOG	641
Basculer la base en mode NOARCHIVELOG	643
Sauvegarde base ouverte, utilisateurs actifs	643
Informations sur le mode ARCHIVELOG	643
Ajouter un fichier redo-log	644
Supprimer un fichier redo-log	644
Déplacer un fichier redo-log	645
Performances et redo-log	645
Cas d'usage de Log Miner	645
Principe d'utilisation de Log Miner	646
Gestion des fichiers de contrôle	646
Les fichiers de contrôle	646
Visualiser les fichiers de contrôle existants	647
Ajouter un fichier de contrôle	647

Sauvegarder un fichier de contrôle base « en marche »	648
Déplacer et supprimer un fichier de contrôle	648
Que faire lorsque tous les fichiers de contrôle sont perdus ?	648
Gestion automatique des fichiers par Oracle	649
Pourquoi utiliser la gestion automatique des fichiers ?	649
Quels inconvénients présente la gestion automatique des fichiers ?	650
Les paramètres d'initialisation	650
Les options utilisées par défaut	651
Exemple d'une base créée avec ces options	651
Conclusion sur la gestion automatique des fichiers	654
Gestion du fichier d'initialisation	654
Démarrage d'une base	654
Création d'un fichier persistant	655
Créer un fichier d'initialisation	655
Modifier des valeurs par l'ordre ALTER SYSTEM	655
Visualiser les valeurs d'initialisation	656
Outils pour modifier les paramètres d'initialisation	657
Résumé	657
CHAPITRE 24	
Stratégie de sécurité sous Windows	659
Sécurité : les points sensibles	660
Le danger vient de l'intérieur	660
Des droits trop étendus	660
Divulgaration des mots de passe	661
Les raisons empêchant de changer de mot de passe	661
Les objectifs	661
La sécurité d'accès physique	662
L'accès à votre machine	662
Les comptes Windows	663
Les utilisateurs Oracle	663
Les besoins d'accès de l'administrateur Oracle 10g	664
La connexion « standard » à Oracle 10g	664
Droits étendus pour démarrer/arrêter Oracle 10g	665
Configuration standard s'appuyant sur le groupe Windows ORA_DBA ..	667
Le groupe ORA_DBA est-il satisfaisant ?	668

Les privilèges SYSDBA et SYSOPER	668
Attribuer les privilèges SYSDBA ou SYSOPER à un utilisateur particulier	669
Quels sont les utilisateurs qui possèdent les privilèges SYSOPER et SYSDBA ?	670
Schémas des utilisateurs connectés sous SYSOPER et SYSDBA	670
Utilisation d'un fichier mot de passe	671
Créer un fichier mot de passe	671
Une base utilise-t-elle un fichier mot de passe ?	674
Faut-il sauvegarder le fichier mot de passe ?	674
Comment tester votre sécurité ?	675
Résumé	675

CHAPITRE 25

Oracle Enterprise Manager	677
Objectifs d'Oracle Enterprise Manager	677
Architecture d'Oracle Enterprise Manager	679
Les différentes versions d'Oracle Enterprise Manager	679
Oracle Enterprise Manager Database Control	680
Oracle Enterprise Manager Database Grid Control	681
Les nœuds administrés	683
Les agents intelligents	683
Le système de jobs	683
Les events ou événements	684
La remontée d'alertes	684
La découverte de services	684
Le repository ou référentiel	685
Les applications standard	685
Les applications optionnelles	685
Les éditions	686
Les Groups ou groupes	686
L'installation d'Oracle Enterprise Manager	686
Installer OEM Database Control	686
Paramétrer OEM pour utiliser les Jobs sous Windows	687
Vérifier le fonctionnement d'Oracle Enterprise Manager	687

Présentation d'Oracle Enterprise Manager	690
Comment naviguer dans Oracle Enterprise Manager ?	690
La connexion à Oracle Enterprise Manager et la page d'accueil	692
L'onglet Performances	695
L'onglet Administration	697
L'onglet Maintenance	698
Les faiblesses d'Oracle Enterprise Manager	699
Résumé	700

CHAPITRE 26

La sauvegarde d'une base Oracle 10g	701
Principes de sauvegarde d'une base Oracle 10g	702
Les éléments à sauvegarder	702
Autres éléments à sauvegarder	702
Les quatre types de sauvegarde d'une base Oracle 10g	703
La sauvegarde logique par Export/Import	704
Avantages et inconvénients des sauvegardes physiques	704
Sauvegarde et restauration	705
Existe-t-il d'autres options de sauvegarde ?	705
Les sauvegardes avec Oracle Enterprise Manager	705
Sauvegarde base fermée ou base ouverte	706
Sauvegarde base fermée	706
Pourquoi ne pas copier les fichiers base ouverte ?	706
Sauvegarde base ouverte	707
Le fichier d'alerte	708
Les sauvegardes par script de commandes	709
Lancement des sauvegardes	710
Compte rendu d'exécution	714
Résumé	716

CHAPITRE 27

Configurer les disques pour Oracle	717
Les différents systèmes de fichiers	718
Les différents systèmes de fichiers Windows	718
Quel système de fichiers pour Oracle 10g ?	718

Les technologies RAID	719
RAID niveau 0	719
RAID niveaux 1 et 0+1	720
RAID niveau 2	720
RAID niveau 3	720
RAID niveau 4	720
RAID niveau 5	721
Comparaison des différents niveaux RAID	721
RAID logiciel et RAID matériel	722
Coexistence RAID 1 et RAID 5	722
Sur quel type de RAID placer les fichiers Oracle ?	723
Les partitions de type Raw Devices	724
Automatic Storage Management (ASM)	725
Résumé	727

CHAPITRE 28

Optimisation et performances	729
L'optimisation de Windows	730
Le support des larges pages	730
Réduire la priorité des applications de premier plan	730
Supprimez les services non indispensables	732
Supprimez les protocoles inutiles et adaptez leur ordre	733
Utilisez le dernier Service Pack de Windows	733
Les outils disponibles sous Windows	734
Le gestionnaire des tâches Windows	734
L'analyseur de performances Windows	735
L'analyseur de performances Windows pour Oracle	735
L'utilitaire Page Fault Monitor	736
Fichier d'échange utilisé par oracle.exe	736
Fermez toutes les applications inutiles	736
Les étapes du tuning	737
Identification des contentions	740

Étape 1 : installer et configurer Oracle 10g	741
Choisir des disques durs	741
Préparer une installation respectant les règles OFA	741
Le DB_BLOCK_SIZE d'Oracle 10g	741
Étape 2 : design de l'application	742
Identifier les points sensibles	742
Des index oubliés	742
Des ordres SQL mal construits	743
Étape 3 : optimisation des traitements et ordres SQL	743
Comment détecter les problèmes ?	743
L'organisation physique de la base de données	743
Disposer de statistiques à jour	743
Les segments d'annulation	744
Les fichiers redo-log	744
L'optimisation des ordres SQL : utilisation de SQL*Plus	745
SQL Tuning Advisor	745
Le plan d'exécution des ordres SQL avec SQL*Plus	746
Étape 4 : tuning de la mémoire	748
N'utilisez que les options nécessaires d'Oracle	748
La pagination	749
Surveillance de la pagination Windows	749
L'optimisation de la mémoire réservée pour Oracle 10g	749
Étape 5 : tuning des entrées/sorties disque	751
Étape 6 : tuning de l'utilisation CPU	751
Fermer toutes les applications inutiles	751
Étape 7 : tuning des différentes contentions	752
Automatic Workload Repository (AWR)	752
Automatic Database Diagnostic Monitor (ADDM)	753
Le SQL Access Advisor	755
L'outil tkprof	755
Les scripts utlbstat/utlestat	759
Les outils complémentaires	760
Résumé	760

PARTIE VI

Annexes

ANNEXE 1

Glossaire des termes utilisés	765

ANNEXE 2

Mémento SQL	775

Langage de définition des données	775

Disponibilité d'accès	775

Impact des modifications	776

Validation des modifications	776

Liste commentée des commandes DDL	776

Langage de manipulation de données (DML)	779

Disponibilité d'accès	779

Validation des modifications	779

Liste commentée des commandes DML	779

Contrôle des transactions	780

Contrôle des sessions	780

Contrôle du système	780

ANNEXE 3

Procédures pour le DBA	781

Bannière d'accueil	782

Version Oracle utilisée	782

Base auditée	783

Taille de la SGA	783

Cache Hit Ratio	783

Paramètres de l'Init.ora	784

Liste des fichiers physiques	786

Liste des fichiers physiques auto extensibles	786

Liste des fichiers redo-log	787

Liste des fichiers de contrôle	787

Liste et propriétés des tablespaces	787

Taille totale et espace disponible dans les tablespaces	788

Espace disponible dans les tablespaces	789
Liste des segments d'annulation	790
Place utilisée par segment	790
Liste des procédures, fonctions, triggers, packages	792
Liste et profil des utilisateurs	792
Liste des utilisateurs possédant les privilèges SYSDBA et SYSOPER ..	793
Liste des rôles existants	793
Rôles affectés aux utilisateurs	794
Liste des profils existants	795
Fin du script	796
ANNEXE 4	
Extensions de fichiers	797
ANNEXE 5	
NLS : support des langues par Oracle 10g et l'euro	799
Qu'est-ce qu'un jeu de caractères ?	799
Qu'est-ce qu'un jeu de caractères pour la base Oracle ?	800
Qu'est-ce que le jeu de caractères d'un client ?	800
Sur quoi agit le LANGUAGE du jeu de caractères d'un client ?	801
Sur quoi agit le TERRITORY du jeu de caractères d'un client ?	801
Comment gérer les paramètres NLS au niveau de la base de données ?	801
Comment visualiser les paramètres NLS de la base et d'une session ? ..	802
Comment s'assurer que des paramètres autres qu'anglo-saxons peuvent être affichés ?	804
En quoi le jeu de caractères influe sur les Export/Import ?	804
En quoi le jeu de caractères influe sur SQL*Loader ?	804
Comment s'effectue la conversion entre jeux de caractères ?	805
Qu'est-ce qu'un jeu de caractères multi-byte ?	805
Quel jeu de caractères choisir pour les architectures Internet ?	805
Comment afficher les messages Oracle en français ?	805
Comment modifier la présentation par défaut des dates ?	806
Comment modifier la présentation par défaut des chiffres ?	806
Oracle et l'euro	807
Pourquoi le stockage de l'euro pose-t-il problème ?	808
L'affichage graphique de l'euro	808

Créer une base de données de test	809
Comment ne pas se perdre ?	810
Le paramétrage du poste client	811
Conclusion	812

ANNEXE 6

Les logiciels livrés	813

ANNEXE 7

Sites Internet et bibliographie	819
Le site de l'auteur	819
Le site de Digora	819
Recherches de livres techniques	820
Sites Internet créés par Oracle	820
Sites Oracle techniques	820
Sites Oracle généraux	821
Sites Oracle locaux et affiliés	821
Les sites externes techniques	821
Les sites Windows-Oracle	822
Les sites Linux-Oracle	822
Freewares, sharewares et logiciels libres	822
Les forums de discussion	823
Les listes de diffusion	823
Comparer les bases de données	824
Devenir un DBA certifié	824
Java, SQLJ et JDBC	824
Revue informatique	825

ANNEXE 8

Installation d'Oracle 10g XE	827
Les apports d'Oracle Express Edition	827
À qui s'adresse Oracle XE ?	827
Principales différences entre XE et les autres versions Oracle	828
Quelle est la stratégie d'Oracle avec XE ?	829
Parcours de lecture pour les utilisateurs d'Oracle XE	829
Les ressources Oracle	831
Les autres ressources	832

Installation d’Oracle XE sous Windows	832
L’installation	832
Le paramétrage optionnel	835
Commandes utiles	836
Développer en PHP	836
Résumé	836
Index	837

Avant-propos

Depuis quelques années, les systèmes d'exploitation proposés par Microsoft sont matures et leur présence dans les entreprises l'atteste : Windows NT, Windows 2000, 2003, Server, Advanced Server, Datacenter, XP et les nouvelles déclinaisons .NET. Quel que soit votre choix ou le parc existant, Oracle est disponible sur ces plates-formes, de même que sur les systèmes Linux et Unix présents dans votre entreprise.

Oracle, leader mondial des bases de données a trouvé en Microsoft des systèmes à la hauteur de son avancée technique. Suivez-nous pas à pas et découvrez les possibilités d'Oracle 10g sous Windows : comment l'installer, le configurer, le programmer, l'administrer et l'intégrer à Windows pour en tirer le meilleur parti.

Notre souci constant est de vous aider à distinguer l'essentiel de l'accessoire et à bien comprendre l'articulation des différents produits. C'est pourquoi nous avons choisi de traiter les chapitres sous forme de tâches à réaliser, plutôt que de vous livrer une liste indigeste de commandes, sans fil conducteur précis.

Objectifs de ce livre

Les produits utilisés

Nous avons choisi de prendre Oracle 10g comme support de ce livre. C'est la dernière version du serveur Oracle disponible sous Windows : une version d'évaluation est téléchargeable gratuitement sur le site <http://oracle.com/technology>. Elle comporte l'intégralité des possibilités d'Oracle 10g offertes sous les autres systèmes d'exploitation du marché.

La version Oracle 10g Express Edition (XE) n'a été annoncée qu'au bouclage du livre. Elle présente un avantage majeur : sa qualité ! Elle est présentée à l'annexe 8, Oracle Express Edition. Un parcours de lecture légèrement modifié par la version XE est indiqué.

Quant à la version de Windows, nous avons utilisé, par souci de simplicité, Windows XP Professionnel. Plusieurs autres versions de Windows conviennent pour utiliser Oracle 10g. Pour connaître la matrice de compatibilité des logiciels Oracle sous les systèmes Microsoft, je vous engage à consulter les sites <http://metalink.oracle.com> ou <http://otn.oracle.com>.

Pour éviter des répétitions, nous parlerons de « Windows » pour qualifier tous les systèmes d'exploitation de Microsoft : Windows NT, Windows 2000, 2003 Workstation, Server, Datacenter, Advanced Server, XP, .NET, etc.

À qui s'adresse ce livre ?

Ce livre se veut une réponse aux attentes de deux catégories de lecteurs, et ce en fonction de trois axes d'orientation.

Il s'agit tout d'abord d'une orientation stratégique destinée aux décideurs : directeurs informatiques, responsables de veille technologique, consultants, pour les aider à comprendre les enjeux d'Oracle 10g sous Windows et leur présenter les apports, les architectures, les logiciels inclus lors de l'achat d'Oracle 10g, les types d'applications qui peuvent être mises en œuvre, etc.

En second lieu, une orientation d'ordre technique est destinée aux administrateurs Windows et Unix, aux responsables réseau et aux développeurs. Leur culture informatique peut émaner de plusieurs sources : utilisateurs d'Oracle 10g sur d'autres plates-formes, développeurs connaissant des bases de données micro telles que SQL Server, Informix, MySQL, Postgres, etc., amenés à mettre en œuvre Oracle 10g.

Les hébergeurs peuvent maintenant installer la version gratuite Oracle Express Edition sur le serveur, sans surcoût pour leurs clients. Oracle se repositionne ainsi face à MySQL.

Enfin, grâce aux versions d'évaluation ou gratuites d'Oracle 10g, il est facile d'installer rapidement un environnement de test et d'apprentissage sous Windows.

Cette orientation technique permet aussi :

- à des administrateurs Oracle 10g confirmés mais novices sous Microsoft, de comprendre et mettre en pratique leurs connaissances acquises sous d'autres systèmes d'exploitation ;
- à des architectes techniques, spécialisés ni dans Oracle 10g, ni dans Microsoft, de comprendre clairement les briques logicielles mises en œuvre avec telle ou telle configuration. C'est particulièrement le cas des nouvelles technologies comme Java, JDBC ou les bases Oracle mises en architecture Cluster sur des serveurs Windows ;
- à des administrateurs Oracle confirmés de découvrir les nombreuses nouveautés apportées par Oracle 10g.

Les connaissances Windows étant très variables, nous aborderons, outre les aspects propres à Oracle 10g, les commandes et les utilitaires Microsoft indispensables pour tirer le meilleur profit d'Oracle 10g.

Fruit de nombreuses années d'expérience en la matière, cet ouvrage vous apprend à vous servir d'Oracle 10g avec toutes les subtilités imposées par les environnements Microsoft. Il représente un outil pédagogique grâce aux très nombreux exemples et remarques qu'il contient et à la clarté de son propos. Certains aspects ont été développés en annexe pour ne pas alourdir les chapitres.

Prérequis à la lecture

Les premiers chapitres ne requièrent pas de la part du lecteur une culture informatique très poussée. Ils présentent essentiellement des concepts, des architectures et « positionnent » différents produits.

Les chapitres techniques nécessitent davantage de compétences en informatique. Ils ne se contentent pas de présenter des fonctionnalités, mais expliquent comment les mettre en place. Il s'agit de chapitres concis répondant à des objectifs précis.

Toutes les opérations effectuées sous Windows et Oracle 10g sont explicitement présentées. Ainsi, tout utilisateur n'ayant pas d'expérience précise sous Windows ou Oracle 10g, mais connaissant un autre environnement (Windows NT, 2000/XP, Linux, Unix ou autre) pourra contrôler avec précision Oracle 10g sous Windows.

Parcours de lecture proposé

Pour plus de clarté, les chapitres ont été regroupés en différentes parties cohérentes. La première partie est orientée stratégie : pourquoi choisir Oracle 10g sous Windows ? Quelles architectures matérielles le couple Oracle 10g-Windows prend-t-il en charge ? Quelles briques logicielles sont fournies lorsque vous achetez Oracle 10g ? Tous ces thèmes permettent de faire le point sur ce domaine. Ils aideront les décideurs et les architectes techniques à cerner l'intérêt d'Oracle 10g et Windows pour leur stratégie actuelle ou future.

La deuxième partie est composée de chapitres principalement théoriques. Ils permettent de comprendre le fonctionnement des différents composants d'Oracle 10g sous Windows : les processus, la mémoire, les fichiers. Enfin, les étapes relatives à la création, au démarrage, à la fermeture d'une base Oracle 10g sont également décrites.

La troisième partie regroupe les chapitres consacrés à l'installation et au paramétrage d'Oracle 10g. L'ensemble des outils et logiciels associés à Oracle 10g y sont présentés ; le middleware Oracle Net y est traité.

La quatrième partie vous apprend comment accéder à Oracle 10g et à programmer. Le langage SQL, PL/SQL et les outils standards d'Oracle 10g sont abordés.

La cinquième partie traite des différents aspects de l'administration d'Oracle 10g. Elle vous apprend à tirer le meilleur profit de votre environnement et à réaliser toutes les tâches essentielles d'administration. Les chapitres de cette partie sont également consacrés à la sécurité, les sauvegardes, l'optimisation et les performances.

Enfin, la sixième partie est composée d'annexes techniques qui évitent d'alourdir les chapitres.

Vous pouvez facilement passer d'une partie à une autre, d'un chapitre à un autre, sans effectuer une lecture linéaire. S'il est souhaitable de lire au préalable tel ou tel chapitre ou si des notions ont été abordées ailleurs, cela est précisé.

Ce que ce livre n'est pas

L'objet de cet ouvrage étant l'intégration d'Oracle 10g sous Windows, il ne vous initie ni à Windows, ni à son installation, paramétrage, etc. De nombreux ouvrages spécifiques ou ressources Internet abordent ce sujet.

Nous avons focalisé notre propos sur l'intérêt offert par Oracle 10g sous Windows, sur sa mise en place et sur toutes les tâches afférentes.

Des exemples précis sont proposés dès lors que des commandes SQL intéressantes ou utiles sont abordées ou qu'un contexte technique nécessite des précisions liées à l'offre ou au positionnement d'un fournisseur. En outre, nous avons inclus à l'attention des développeurs Oracle 10g débutants des chapitres d'initiation au langage SQL et PL/SQL.

Partie I : Oracle 10g sous Windows

Chapitre 1 : Pourquoi choisir Oracle 10g sous Windows ?

Ce chapitre revêt une importance stratégique. Il aborde le positionnement de Windows 2000/XP par rapport à leurs grands rivaux Unix et Linux. Il précise les enjeux majeurs lors du choix d'un système d'exploitation. Il vous fournit les arguments pour justifier le choix d'une base Oracle sous Windows auprès de votre direction. De nombreux exemples d'utilisation d'Oracle 10g et d'architectures techniques mises en place y sont présentés.

Chapitre 2 : Les nouveautés d'Oracle 10g

Oracle 10g apporte un grand nombre de nouveautés qui sont présentées et commentées dans presque tous les chapitres de ce livre. *Les nouveautés d'Oracle 10g* permettent à un administrateur Oracle confirmé de découvrir rapidement les nouveautés et les chapitres qui s'y rapportent.

Chapitre 3 : Comparatif technique Windows/Linux

On peut comparer Windows 2000/XP et Linux durant de longues heures sans obtenir de résultat probant en raison de la constante partialité de jugements. Nous avons choisi pour notre comparaison d'observer différentes étapes que chaque administrateur rencontre en utilisant une base de données. Nous n'évaluons pas les aspects système d'exploitation ou performances, mais les tâches rencontrées au quotidien sur les deux systèmes par un administrateur Oracle 10g.

Chapitre 4 : Environnement Windows : l'indispensable

Nous énumérerons les outils et commandes Windows standard mais aussi d'autres moins courants qui peuvent s'avérer utiles. Nous utilisons ces outils et ces commandes tout au long de ce livre.

Partie II : Principes de fonctionnement d'Oracle 10g

Chapitre 5 : Fonctionnement d'Oracle 10g sous Windows

Ce chapitre constitue une introduction indispensable pour comprendre le fonctionnement d'Oracle sous Windows. Il donne une vision d'ensemble des interactions entre Oracle et Windows. Il précède d'autres chapitres qui décrivent plus précisément à la fois le fonctionnement d'Oracle et certains liens avec Windows, mais cette présentation générale est une introduction indispensable.

Chapitre 6 : Les clusters Oracle

Ce chapitre fait le point sur les solutions cluster Windows et Oracle existantes. Des solutions cluster Windows arrivent sur un marché jusque là réservé à d'autres systèmes d'exploitation. C'est un domaine nouveau à la fois pour Windows et pour les constructeurs de matériel micro qui devront intégrer cette technologie ou plutôt ces technologies. Car sous le nom de Microsoft Cluster Server (MSCS), Microsoft introduit étape par étape des technologies allant des plus simples aux plus rigides.

Chapitre 7 : Les processus d'Oracle 10g

Ce chapitre est consacré aux processus qui assurent le fonctionnement d'une instance Oracle 10g. Les principaux composants (fichiers, processus et espaces mémoire) sont présentés puis nous entrons dans le détail du fonctionnement interne des processus, pour vous permettre de mieux comprendre le rôle de chacun, son impact sur le fonctionnement d'une instance et les performances des bases Oracle 10g.

Chapitre 8 : L'utilisation de la mémoire par Oracle 10g

Dans tout système informatique, l'utilisation de mémoire est synonyme de performance. L'accès et l'utilisation des données sont beaucoup plus rapides en mémoire que sur disque. C'est pourquoi Oracle 10g, utilisant des processus efficaces, a souvent recours à la mémoire physique des ordinateurs.

Ce chapitre traite des différents types de mémoire présents sous Windows et comment Oracle 10g les utilise. Il est important de bien comprendre ces éléments, car ils interviennent dans les opérations d'amélioration des performances.

Chapitre 9 : Les fichiers d'une base Oracle 10g

Oracle 10g a spécialisé certains fichiers pour conserver les données, d'autres sont uniquement dédiés au fonctionnement interne de la base ou à son paramétrage. Dans ce chapitre, nous ne nous limitons pas aux fichiers qui composent la base de données et qui doivent être sauvegardés, mais nous abordons aussi les fichiers de paramétrage, trace, log, etc. Ils sont méconnus mais pourtant d'une grande utilité pour rechercher la source de problèmes.

Nous décrivons l'architecture OFA (*Optimal Flexible Architecture*) qui propose une méthode rigoureuse et structurée simplifiant l'administration Oracle.

Chapitre 10 : Démarrer et arrêter une base Oracle 10g

Ce chapitre présente toutes les étapes utiles à la gestion d'une base Oracle 10g : sa création, son ouverture/fermeture et sa suppression. De nombreux cas particuliers sont évoqués et une démarche progressive révèle les mécanismes internes d'Oracle 10g sous Windows. Nous vous présentons les grandes étapes de la création d'une nouvelle base de données sous Windows.

Chapitre 11 : Transactions et accès concurrents

Toute base de données a pour objectif de fournir aux utilisateurs un accès simultané aux données. La notion de concurrence d'accès et de verrouillage des données intervient lorsque plusieurs utilisateurs essaient d'accéder simultanément aux mêmes données. Le concept de transaction est différent mais il n'en reste pas moins à la base de la gestion des accès concurrents : les données modifiées lui sont réservées jusqu'à sa fin. Comme ces mécanismes sont assurés par les segments d'annulation, ils sont également abordés dans ce chapitre.

Chapitre 12 : Le dictionnaire de données Oracle 10g

Le dictionnaire de données Oracle 10g est souvent obscur, même pour de nombreux administrateurs Oracle 10g. Ce chapitre traite à la fois de sa création « minimale », des différents scripts SQL additionnels et des vues qui permettent d'y accéder. Les vues DBA, ALL, USER, V\$ sont développées car leur connaissance générale est indispensable à tous les administrateurs Oracle 10g. En outre, nous fournissons une liste complète de ces vues.

Partie III : Installer et paramétrer Oracle 10g sous Windows

Chapitre 13 : L'installation d'Oracle 10g sous Windows

Oracle 10g est un produit très complet, qui intègre la base de données mais aussi tout un assortiment d'utilitaires et de logiciels. Nous détaillons l'ensemble de ces logiciels, en précisant leur utilisation. Nous installons Oracle 10g et une base initiale utilisée lors des tests.

Chapitre 14 : Création d'une base Oracle 10g

Le logiciel *Database Configuration Assistant* (DBCA, écrit en Java) offre une interface graphique conviviale destinée à vous assister lors de la création d'une nouvelle base de données. Si l'interface graphique constitue un plus, la création d'une base demeure néanmoins complexe. C'est pourquoi toutes les étapes sont détaillées.

Chapitre 15 : Oracle Net, le middleware Oracle

Ce chapitre explique comment installer, configurer et mettre en œuvre Oracle Net, le middleware de communication Oracle. Des exemples précis sont fournis pour vous seconder pas à pas.

Chapitre 16 : Accéder à Oracle depuis Windows

Ce chapitre constitue la suite logique des précédents : une fois le logiciel Oracle installé et Oracle Net configuré, quelles sont les possibilités d'accès aux bases de données depuis les applications développées en environnement Windows ?

Nous décrivons les accès ODBC, Oracle OLE DB, Oracle Objects for OLE (OO4O), ActiveX Data Objects (ADO), Oracle Call Interface (OCI) ainsi que les nouvelles interfaces .NET : ODBC .NET, Oracle OLE DB .NET, Oracle Data Provider for .NET.

Partie IV : Outils et langages de programmation

Chapitre 17 : Programmer avec PL/SQL

PL/SQL est le langage procédural de troisième génération (L3G), extension du SQL, intelligemment intégré au serveur Oracle 10g. Il permet de manipuler toutes les données Oracle 10g : relationnelles, objet-relationnel, Java. C'est un langage de programmation à la fois puissant, simple et moderne, destiné à exprimer facilement vos règles de gestion complexes sous forme de procédures et de triggers stockés.

Ce chapitre vous propose une prise en main progressive du PL/SQL, adaptée aux besoins des développeurs. De nombreux exemples sont proposés, les différentes techniques exposées et des règles de bonne programmation établies.

Chapitre 18 : Les outils d'Oracle 10g

Depuis des années, chaque version d'Oracle 10g est livrée avec un ensemble d'outils standards. On retrouve ces outils sur toutes les machines où Oracle 10g est installé. Il est très intéressant de les découvrir car ils sont puissants et performants. Livrés gratuitement avec Oracle 10g, ils rendent l'utilisation de produits complémentaires facultative.

Ce chapitre aborde les outils suivants :

- Export/Import : il s'agit de deux utilitaires qui fonctionnent l'un avec l'autre. Export permet d'exporter la totalité ou une partie du contenu d'une base Oracle 10g, Import autorise l'importation de la partie exportée dans une base de données Oracle 10g. Export/Import sont principalement utilisés lors de sauvegardes, de réorganisations et de déplacement du contenu d'une base Oracle 10g d'une machine à une autre machine. La portabilité des fichiers exportés permet, par exemple, à un Export réalisé sur une base Oracle 10g fonctionnant sous Windows 2000/XP d'être importé dans une base Oracle 10g sous Windows et inversement.
- Data Pump Import et Data Pump Export, les nouvelles versions d'Import et Export apparues avec Oracle 10g ;
- SQL*Loader est un utilitaire très puissant qui permet de charger des bases de données Oracle 10g à partir de fichiers contenant les données.

- SQL*Plus est l'interface mode caractère d'accès à Oracle 10g la plus répandue. Cet outil est indispensable à tout développeur ou tout administrateur Oracle 10g, car il représente l'interface SQL privilégiée pour accéder et administrer les bases Oracle 10g.

Chapitre 19 : Pilotes JDBC et SQLJ

Dans les architectures d'applications distribuées les middlewares jouent un rôle fondamental. Ils servent à l'échange de données entre des applications et des bases de données. Les standards JDBC et SQLJ sont des middlewares destinés aux programmes Java qui permettent aux applications Java autonomes ou aux applets Java de communiquer avec des bases de données.

Ce chapitre vous explique quand et comment utiliser les pilotes JDBC. D'autre part, nous abordons SQLJ pour exposer comment incorporer facilement des instructions SQL dans du code Java.

Chapitre 20 : Java et Oracle 10g

Ce chapitre aborde dans le détail les fonctionnalités Java intégrées dans le serveur de base de données Oracle 10g. Ce n'est donc pas un guide d'apprentissage du langage Java et sa connaissance est un prérequis à la lecture de ce chapitre.

Oracle 10g fournit une plate forme de déploiement destinée aux applications réseaux (intranet ou Internet) qui intègre la machine virtuelle Java Oracle JServer dans son serveur de base de données. Elle vient en complément du moteur PL/SQL. Avec Java et Oracle 10g, il devient possible de réaliser simplement des applications par composants sous forme d'Enterprise JavaBeans, d'objets CORBA ou plus simplement de procédures stockées.

Chapitre 21 : XML et Oracle 10g

XML, langage universel pour l'échange de données, est destiné à être utilisé de plus en plus fréquemment. C'est pourquoi il est important d'y faire référence dans cet ouvrage, d'autant plus qu'Oracle gère nativement le XML depuis plusieurs versions, avec un spectre technique de plus en plus large. Ce format de données peut être stocké dans une base et facilite la manipulation des données et la communication entre applications.

Chapitre 22 : Oracle Application Express

Ce chapitre est consacré à la présentation d'un nouveau produit Oracle apparu récemment : Oracle Application Express. Successeur de l'antique WebDB, Oracle Application Express propose un environnement de développement 100 % accessible depuis un navigateur Internet pour créer des applications Web centrées autour d'une base de données Oracle.

Partie V : Administration d'Oracle 10g

Chapitre 23 : Gestion de l'espace disque et des fichiers

Une base Oracle 10g vous permet de stocker des données, de les modifier, de les supprimer, etc. Toutes ces actions sont conservées dans les fichiers qui composent la base de données.

Cependant, comment Oracle 10g gère-t-il l'espace disque qui lui est alloué au cours de toutes ces opérations ? Comment dimensionner la taille d'une base de données ? Comment contrôler l'évolution des différents constituants et anticiper les problèmes ? Quels peuvent être ces problèmes ? Ce chapitre répond à toutes ces questions par une approche théorique et pratique : il vous aide à comprendre et agir au cœur de la gestion de l'espace disque de vos bases Oracle 10g.

Chapitre 24 : Stratégie de sécurité sous Windows

Ce chapitre aborde le thème de la sécurité, les différentes techniques pour démarrer/arrêter une base Oracle 10g et s'y connecter avec ou sans mot de passe. Ces techniques sont intéressantes car elles permettent de réaliser des scripts de commande, sans y faire figurer de mot de passe : la sécurité du système n'est pas compromise. Il est ainsi possible de changer tous les mots de passe sans risque de rendre des scripts inopérants.

Chapitre 25 : Oracle Enterprise Manager

Oracle Enterprise Manager est un environnement d'administration et de supervision de nouvelle génération, développé d'une manière modulaire et ouverte. Très complet, il permet d'envisager des tâches complexes avec beaucoup de sérénité. Nous décrivons ses différents composants, leur mise en œuvre et leur utilisation.

Chapitre 26 : La sauvegarde d'une base Oracle 10g

Une base de données contient souvent des informations vitales pour votre société. La mise en place de sauvegardes simples et efficaces est essentielle à votre sécurité et tranquillité d'esprit. C'est la tâche majeure dévolue à tout administrateur Oracle 10g.

Ce chapitre a pour objectif de vous aider à choisir le meilleur système de sauvegarde. Nous aborderons les différentes techniques de sauvegarde d'une base Oracle 10g, nous évoquerons Enterprise Manager avant de proposer une méthode simple et efficace de scripts pour automatiser les sauvegardes.

Chapitre 27 : Configurer les disques pour Oracle

Ce chapitre commente un aspect important de votre système : le stockage des données. Différentes techniques existent sous Microsoft. Sous ces technologies, vous pouvez mettre en œuvre des techniques RAID ou encore *Automatic Storage Management* (ASM), une nouveauté d'Oracle 10g. Tous ces points sont traités dans le cadre de l'installation d'une base de données, qui présente des contraintes spécifiques.

Chapitre 28 : Optimisation et performances

Comment tirer le meilleur parti d'Oracle 10g sous Windows ? Le sujet est vaste : l'optimisation ou « tuning » consiste à établir le meilleur niveau de performances d'un système en tenant compte de contraintes souvent contradictoires. Notre objectif est de vous présenter les éléments fondamentaux pour assurer des performances aussi bonnes que possible.

Partie VI : Annexes

Annexe 1 : Glossaire des termes utilisés

Liste et définitions des termes techniques utilisés dans ce livre.

Annexe 2 : Mémento SQL

Mémento d'ordres SQL, classés par thème. Ce mémento répertorie les commandes SQL recensées. Nous expliquons brièvement l'utilité de chacune d'entre elles.

Annexe 3 : Procédures pour le DBA

Ordres SQL utiles à un administrateur de bases de données Oracle 10g.

Annexe 4 : Extensions de fichiers

Cette annexe précise la signification des différentes extensions de fichiers utilisés dans une installation Oracle 10g. Certaines peuvent être modifiées, mais il est toujours bon de conserver un standard de nommage.

Annexe 5 : NLS, le support des langues par Oracle 10g et l'euro

Cette annexe traite du NLS (*National Language Support*) ou support des langues nationales par Oracle 10g. Les spécificités du stockage du caractère euro dans une base Oracle 10g sont abordées.

Annexe 6 : Les logiciels livrés

Cette annexe présente les logiciels livrés avec Oracle 10g.

Annexe 7 : Sites Internet et bibliographie

Cette annexe fournit un ensemble d'adresses Internet pour obtenir des informations sur Oracle 10g. Elles constituent un excellent point de départ pour découvrir la richesse des sites proposés, qu'ils soient d'origine Oracle ou non.

Annexe 8 : Installation d'Oracle 10g XE

Cette annexe décrit la procédure d'installation d'Oracle 10g XE. Enfin, n'hésitez pas à consulter mon site Web sur lequel vous trouverez des compléments et mises à jour du livre.

Gilles Briard

<http://gbriard.club.fr>

Partie I

Introduction

Chapitre 1 : Pourquoi choisir Oracle 10g sous Windows ?

Chapitre 2 : Les nouveautés d'Oracle 10g

Chapitre 3 : Comparatif technique Windows/Linux

Chapitre 4 : Environnement Windows : l'indispensable

1

Pourquoi choisir Oracle 10g sous Windows ?

Dans ce chapitre :

- pourquoi choisir Oracle 10g sous Windows ?
- le positionnement de Microsoft et de Linux dans l'environnement informatique actuel ;
- les modes de commercialisation d'Oracle ;
- les utilisations possibles d'Oracle 10g pour Windows.

On oppose souvent Windows à Linux/Unix et de nombreuses questions se posent lors du choix d'une base de données. Chaque système a ses avantages et ses inconvénients et Oracle a pour principal avantage d'être parfaitement intégré à Windows. Cette intégration complète exploite les caractéristiques les plus poussées de Windows.

La force principale d'Oracle est la portabilité de sa base de données. Aujourd'hui, il y a un groupe de quelques machines auxquelles Oracle accorde une priorité. Windows en fait partie, au même titre que Linux, HP/UX (Hewlett Packard), Solaris (Sun) et Aix (IBM). C'est-à-dire que toute sortie d'une nouvelle version d'Oracle sera synchronisée de manière prioritaire pour l'ensemble de ces plates-formes.

L'engagement envers Linux pousse actuellement Oracle à livrer prioritairement cette version. Les autres suivent avec un léger décalage.

Opposer Linux/Unix et Windows pour les applications d'entreprise : vraie guerre ou faux débat

On parle beaucoup de la « guerre » entre Windows et Linux/Unix. Il est vrai que les enjeux économiques et stratégiques sont capitaux pour les entreprises dont les choix doivent allier performance, pérennité et meilleur coût. Mais on ne peut pas choisir entre Windows et Linux/Unix sans se poser les questions suivantes :

- pour quelle utilisation ?
- quel est le niveau de performance requis ?
- quelle est la disponibilité demandée ?
- quelles sont les compétences de mes équipes ?
- quel en sera le coût réel ?

Rappelez-vous, il y a quelques années, lors de la montée en puissance d'Unix. À cette époque, on opposait Unix (le « petit ») aux mainframes (les « gros »). Chacun y allait de son avis : qui allait manger qui ? L'un allait écraser l'autre ! L'un était mûr, l'autre immature ! Ça ne prendra jamais ! Ça ne peut pas marcher, etc. Cela évoque fortement l'affrontement actuel entre Windows et Linux/Unix.

Unix n'a pas tué le mainframe, il l'a remplacé pour certaines applications. De même, Windows vient concurrencer Unix et peut le suppléer pour certaines applications. C'est une bonne raison pour accorder toute son attention à Windows, mais aussi pour apprécier les limites de ce type de serveur.

Il ne faut pas oublier que si Windows monte en puissance, dans le même temps, Linux/Unix progresse et les mainframes tout autant ! Évitions l'écueil qui consiste à comparer des systèmes de générations différentes.

Pour de petites applications avec quelques utilisateurs connectés, des volumes de données de quelques giga-octets, des postes de développement autonomes, des projets du type groupes de travail, Windows présente des avantages certains face à Unix. Ce type de projet et d'environnement est de plus en plus répandu.

À l'inverse, si vous devez utiliser des terminaux en mode caractère, si vous avez beaucoup d'utilisateurs connectés, si la puissance et la disponibilité sont importantes, si vous devez exploiter votre système sans avoir à le relancer à chaque modification, c'est Linux/Unix qui reprend l'avantage.

Les choix sont plus difficiles lorsque la charge que vous devez soumettre à votre serveur se situe dans une zone intermédiaire. Voici quelques aspects qui peuvent vous aider à choisir :

Une large bibliothèque de logiciels a été développée autour d'Oracle. Pour de petits groupes de travail, la solution la plus rapidement opérationnelle consistait à placer un serveur Windows équipé d'Oracle et des postes clients équipés de la solution logicielle.

Unix se positionnait mal pour ce style de projet, mais Linux vient concurrencer Windows sur ce segment d'application.

Windows n'est pas réputé pour tirer le meilleur parti des processeurs qu'on peut lui ajouter, comparé à Unix. Ce qui veut dire que vous n'êtes pas assuré qu'une application qui fonctionne bien sur une machine Windows avec un processeur verra sa puissance grimper linéairement (ou presque) par l'adjonction d'autres processeurs. Mais avez-vous réellement besoin d'une machine à 16 processeurs ?

D'un point de vue administration, Unix dispose de langages de scripts plus sophistiqués (script Shell versus ligne de commande), lui offrant de plus grandes possibilités. Le langage de scripts ne vient d'arriver qu'avec les dernières versions de Windows. De même, des outils complémentaires doivent être ajoutés à Windows pour effectuer nombre de tâches, alors que la bibliothèque standard d'Unix offre un riche éventail de fonction.

Windows est avantageux pour les réseaux réduits qui n'ont pas besoin d'une administration sophistiquée, alors qu'Unix propose des choix d'architectures plus larges. Comme vous le voyez, c'est ce que l'on souhaite faire qui détermine les choix. L'autre critère important est votre « sensibilité » informatique : c'est le règne des critères affectifs et irrationnels, et là, il y a souvent une partialité proche des guerres de religion ! En tout état de cause, une configuration hybride avec à la fois des serveurs Windows et d'autres Unix permet, le cas échéant, de tirer bénéfice des avantages de chaque système.

Enfin, le choix du type de serveur peut être imposé par l'application (le progiciel) que l'entreprise achète. Certaines applications valident un système d'exploitation et une version Oracle, ce qui réduit la latitude de choix.

Vers une nouvelle stratégie de distribution

Depuis des années, le coût des logiciels baisse. Cette baisse est souvent compensée par l'introduction des nouveautés dans le logiciel de base. Il y a quelques années, le protocole TCP/IP était une option facturée des systèmes d'exploitation. Aujourd'hui, même les mainframes (les plus réticents à l'ouverture vers les standards) proposent TCP/IP dans leurs logiciels de base. Le système d'exploitation se « banalise ». Le message est clair : ce n'est plus le système d'exploitation qui procure un avantage concurrentiel à votre entreprise.

Ce sont maintenant les architectures Web qui dictent le choix du système d'exploitation des serveurs. C'est pour contrer cela que Microsoft lance sa grande offensive .NET. Il s'oppose à Apache, Java et tous les logiciels libres de ce domaine.

Les adresses incontournables d'Oracle sont www.oracle.com et <http://www.oracle.com/technology>. Depuis ces adresses, vous pouvez télécharger de nombreux produits Oracle, dont la base de données en version Express ou XE (gratuite), Standard ou Entreprise.

Dans une précédente version de ce livre, j'avais indiqué : « Pour contrer l'arrivée des bases de données sous forme de logiciels libres, Oracle dispose d'un argument de poids : agir sur son prix ! Surveillez la politique tarifaire d'Oracle avant d'envisager une acquisition. » C'est maintenant chose faite, il existe même la version Oracle Express gratuite, présentée à l'annexe 8.

Une couverture de l'ensemble des besoins de l'entreprise

La micro-informatique est désormais prédominante dans l'environnement des entreprises. De nombreuses applications nécessitent un stockage de données. Pourquoi ne pas mettre à la disposition des utilisateurs familiers du PC, des outils leur permettant de stocker et de manipuler des volumes importants de données, d'autant que des mémoires de masse considérables équipent désormais tous les micro-ordinateurs ? Comment éviter de recourir à des solutions hétérogènes, « micro », « mini » et « mainframe » qui communiquent peu ou mal ? *Oracle Database 10g Standard* et *Oracle Database 10g Enterprise* et Oracle Database Express Edition répondent à ces besoins, en offrant des caractéristiques techniques strictement identiques à celles de leurs aînés qui fonctionnent sur des machines plus puissantes. Ils vous permettent ainsi de prendre pied dans l'univers d'Oracle, avec un investissement réduit et en utilisant des machines que vous maîtrisez déjà.

Prendre en charge avec une même technologie des projets d'envergure différente

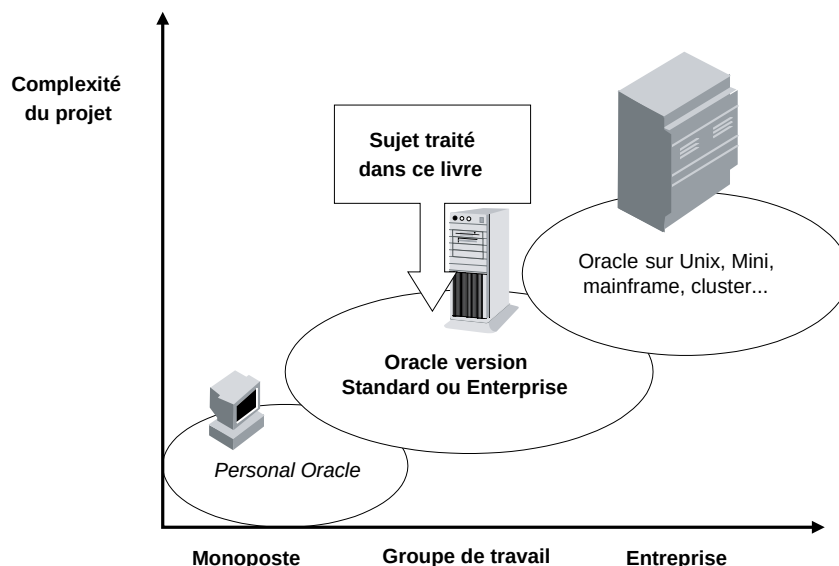


Figure 1-1

Positionnement d'Oracle 10g pour Windows

Oracle 10g est porté sur différentes machines et systèmes d'exploitation. Tout d'abord, *Personal Oracle 10g* est une base de données monoposte, ne permettant pas à des utilisateurs distants de se connecter. *Oracle 10g pour Windows*, version *Standard* ou *Enterprise* est une base de données multi-utilisateur autorisant des clients locaux, intranet ou distants à se connecter. Enfin, Oracle 10g est utilisé sur de très nombreuses machines puissantes, de type Unix, mini ou mainframe.

Oracle 10g : une seule souche technologique

Ce qui est primordial, c'est que toutes ces versions sont issues de la même souche technologique. Toutes les fonctions accessibles aux développeurs pour l'une de ces versions se retrouvent dans les autres. Il convient de parler du *portage* d'Oracle sur différentes machines, car, à numéro de version équivalent, ce sont les mêmes produits.

Cette situation offre de grands avantages, car elle vous permet :

- de disposer d'une même base de données pour des développements personnels, un groupe de travail ou l'ensemble de l'entreprise ;
- de prendre en charge des projets allant du plus simple, nécessitant peu de puissance, au plus complexe, qui en requiert beaucoup ;
- d'utiliser la bonne machine au bon endroit, selon vos besoins, votre budget, avec une puissance adaptée ;
- d'utiliser Oracle avec le système d'exploitation que vous maîtrisez ;
- d'autoriser une souplesse d'architecture, incorporant différentes technologies (client-serveur, Internet, Java, etc.), fédérées par une seule base de données ;
- de respecter les choix technologiques des entreprises en assurant une cohérence autour de la base de données ;
- de changer le système d'exploitation qui héberge les bases de données sans remettre en cause les applications.

Permettre une réelle concurrence

Comment choisir une machine, un système d'exploitation, une base de données, sans se tromper ? Le choix de l'un des composants détermine-t-il obligatoirement celui des autres ? Comment concilier la durée de vie des applications avec les choix technologiques ? Comment faire évoluer un tel parc ?

Oracle apporte une réponse en transformant l'informatique moderne en un meccano, un assemblage de composants. Le point invariant d'un système d'informations n'est plus le système d'exploitation ou un type de machine, mais les données et les applications. C'est un recentrage nécessaire de l'informatique : mettre la technique au service des besoins et non l'inverse.

Ce recentrage et cet assemblage de composants qui banalisent les machines et systèmes d'exploitation au profit de la base de données permettent une diversité de fournisseurs. Cela renforce la concurrence pour chaque composant. La base de données n'est pas l'élément le plus coûteux d'une configuration, mais c'est à la fois le plus structurant et celui qui vous procure la plus grande liberté de choix pour l'ensemble des autres composants.

Windows est un système d'exploitation comme un autre, ce qui signifie qu'il présente des avantages et des inconvénients : à vous de les apprécier et d'en tirer le meilleur parti.

Quelles utilisations possibles d'Oracle 10g ?

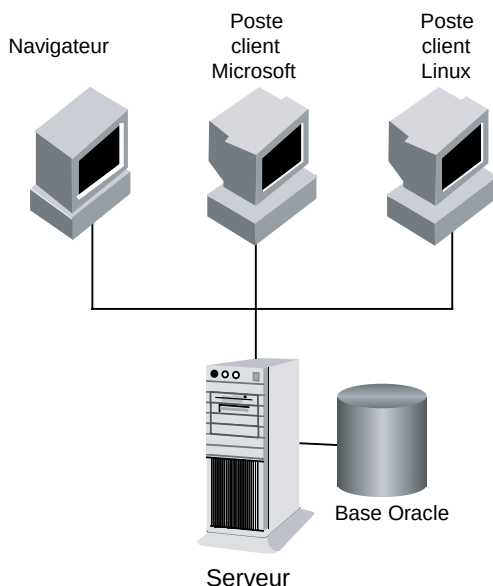
Tout projet nécessitant un stockage de données peut être réalisé par ces logiciels. Il peut s'agir d'un progiciel acheté, d'une application que vous développez, de la réalisation d'un infocentre ou d'une messagerie, bref, tout ce qui était envisageable avec les bases de données connues depuis l'apparition de la micro-informatique, plus ce que seule la puissance des mini-ordinateurs permettait.

Oracle 10g : serveur de données

Oracle a établi sa réputation comme serveur de données, sur de multiples plates-formes : Windows est l'une d'entre elles. Toutes les applications « classiques » nécessitant une base de données peuvent être mises en œuvre avec Oracle 10g sous Windows.

Figure 1-2

Oracle 10g, serveur de données sous Windows



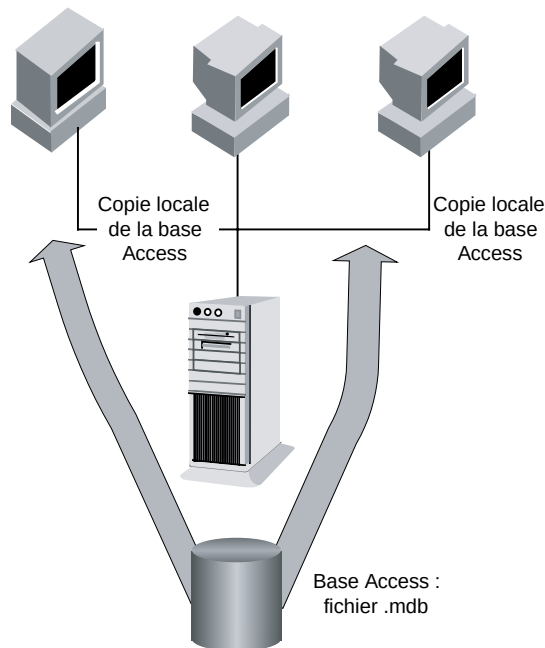
Par sa facilité d'installation et de paramétrage, un serveur Windows relié à des postes Windows 98/2000/XP, Linux, ou encore à des navigateurs Internet, constitue facilement une configuration client-serveur à faible coût. Dans cette architecture, les programmes fonctionnent sur les clients et interagissent avec la base de données. Seules les données utiles aux programmes sont échangées entre le client et le serveur.

Pourquoi pas une base de données type Microsoft Access ?

La question est légitime : pourquoi ne pas préférer une base de données de type Microsoft Access à Oracle ?

Figure 1-3

*Fonctionnement
d'une base Access*



Le principal inconvénient de Microsoft Access (ou plus généralement de ce type de bases) est qu'à chaque utilisation, c'est l'ensemble de la base de données qui est transféré sur le poste client. De plus, il y a verrouillage des données à chaque mise à jour, ce qui bloque fréquemment les autres utilisateurs. Le point fort d'Access, c'est à son intégration : gestion de données, outils de développement, création d'éditeurs. Facile d'utilisation pour de petites configurations (peu d'utilisateurs simultanés, faible volume, peu de transactions), ce type de bases doit être limité à cet usage.

Pourquoi pas une base de données « libre » ?

D'excellentes bases de données « libres » existent. La plate-forme LAMP (Linux, Apache, MySQL, PHP) a popularisé MySQL. D'autres bases, comme SQL Lite (qui accompagne maintenant PHP), Postgres, etc., peuvent concurrencer Oracle sur certains points.

Pour autant, je suis souvent frappé par des comparatifs qui placent au même niveau technique des bases libres et Oracle. C'est partiellement vrai pour Postgres, mais les autres bases sont techniquement loin derrière Oracle. Faut-il pour autant les négliger ? Ce sont souvent d'excellents choix pour stocker les données d'un frontal Web et les coûts pour disposer d'une base MySQL chez un hébergeur n'ont rien à voir avec ceux nécessaires pour faire héberger une base Oracle.

Vous pouvez ainsi tirer parti du « meilleur composant » au « meilleur endroit » : Oracle pour sa puissance, sa fiabilité, ses possibilités techniques impressionnantes et une base libre pour un frontal ou une application Web hébergée. Il ne vous reste qu'à synchroniser les deux environnements, ce qui est fort simple.

Autre limite des logiciels libres : les éditeurs de progiciels ont développé leurs applications avec une base stable, mature, largement répandue dans les entreprises : Oracle. Il faudra attendre une refonte complète de leur logiciel pour qu'ils changent de base de données. Là encore, ils ne choisiront une base libre que si celle-ci offre toutes les garanties (performance, palette fonctionnelle, support technique, etc.) que les clients attendent.

Les applications client-serveur

Oracle 10g permet un réel fonctionnement client-serveur, ce qui fait de Windows une plate-forme privilégiée pour le déploiement d'applications de type départemental.

La puissance du serveur doit être adaptée aux besoins. Windows fonctionne sur des serveurs mono ou multiprocesseur offrant une plage de puissance intéressante. Le haut de gamme est à réserver à Unix, qui dispose de très nombreuses références en ce domaine.

Oracle 10g permet de disposer d'une même technologie sur des machines de puissance très différentes : PC monoposte, serveur Windows, serveur Unix. Il y a, bien entendu, des segments de marché qui mettent ces machines en concurrence.

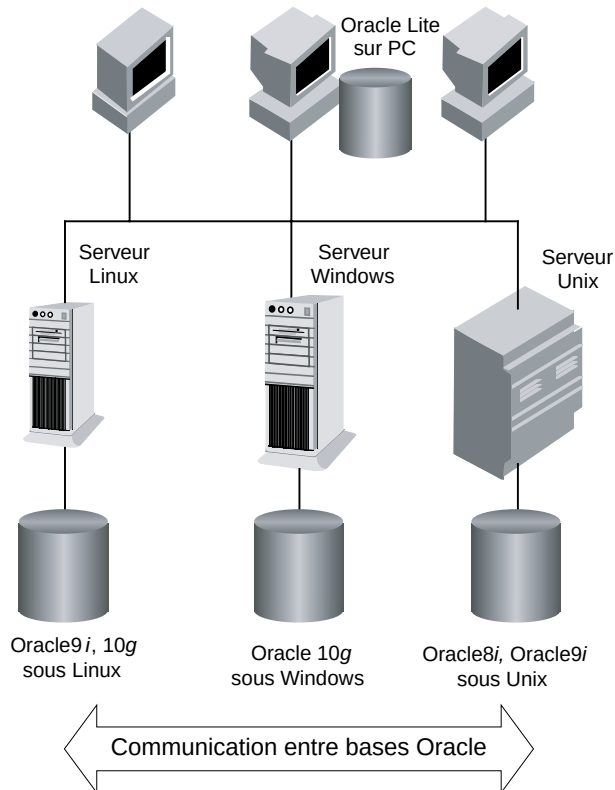
Le fait que de petites machines suffisent pour programmer des applications susceptibles d'être déployées sur de plus grosses offre de grands avantages. Par exemple, les sociétés de service qui développent des progiciels n'ont plus à se soucier de l'environnement cible, d'où les très nombreux logiciels proposés par des partenaires d'Oracle.

L'intégration de Windows dans le système d'information

Que les bases Oracle soient situées sur un poste autonome, sur un serveur 2000/XP ou sur un serveur Windows, il est toujours facile de les faire communiquer.

Figure 1-4

Communication entre bases Oracle



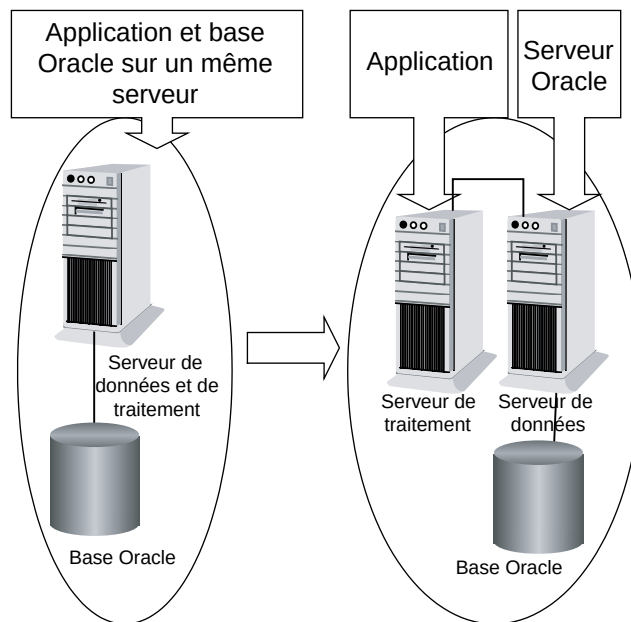
Dans cette architecture, tout le monde communique avec tout le monde : le serveur Windows peut être à la fois serveur pour les bases de données qu'il contient et client des bases situées sur Linux ou Unix. Il en va de même pour le serveur Linux/Unix qui peut se transformer en client du serveur Windows.

Partition des applications

Cette souplesse d'architecture apportée par le fonctionnement en client-serveur permet de répartir les applications comme bon vous semble. Par applications, nous entendons toutes sortes de programmes exécutables (écrans, éditions, traitements « batch »...) qui accèdent aux données de la base.

Figure 1-5

Partitionnement
des applications



Exemple caractéristique : vous utilisez une application de gestion de paie et, lors des éditions des bulletins, le serveur arrive à saturation, victime de surexploitation. Que faire ? Augmenter sa puissance ? Acheter une autre machine ? Changer de système d'exploitation ? Il suffit de répartir les applications sur différentes machines qui accèdent toutes à la même base de données. C'est la solution idéale si une machine tierce est disponible.

La séparation entre serveur de données et serveur de traitement se rencontre très souvent dans les architectures Internet. On multiplie les serveurs « frontaux » reliés à Internet et on centralise l'accès aux données. Plusieurs serveurs de traitement accèdent alors à un serveur de données.

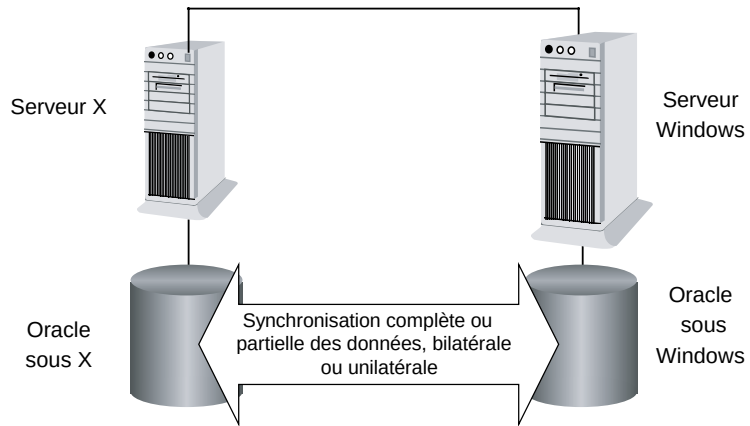
La réplication des données

Répartir des informations dans plusieurs bases de données ne va pas sans poser des problèmes techniques. Pour les résoudre et assurer une cohérence entre des données synchronisées, la *réplication* intègre l'ensemble des techniques à mettre en œuvre.

Oracle 10g sous Windows permet une synchronisation aisée des données entre plusieurs bases situées sous Windows ou sous d'autres systèmes d'exploitation. Utiliser Oracle 10g apporte la garantie de ne pas rester « isolé » et de disposer d'un système d'informations ouvert et communicant.

Figure 1-6

Principe de la réplication de données

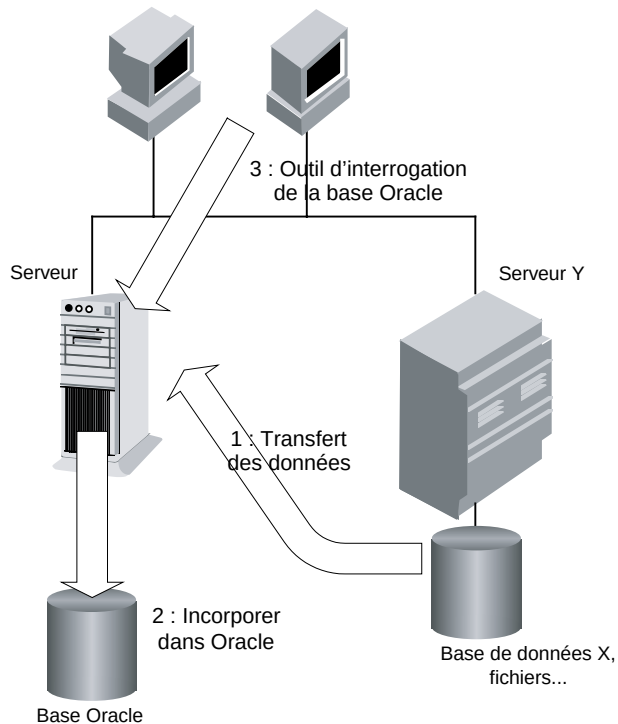


Infocentres, datawarehouses et data marts

Les données d'entreprise sont de véritables mines d'informations. Les solutions de type infocentre, datawarehouse ou data marts consistent à placer les données dans une base Oracle 10g, à en rafraîchir régulièrement le contenu et à l'interroger à des fins d'analyse, de consolidation et de décision.

Figure 1-7

Oracle 10g en utilisation décisionnelle



Il est préférable d'éviter de travailler directement sur la machine hébergeant les applications de production, car des interrogations libres peuvent affecter les performances globales de votre activité courante. De plus, les données ne se trouvent pas forcément dans une base relationnelle comme Oracle 10g, dont l'accès est facilité par des outils d'interrogation du marché : Business Objects, Impromptu de Cognos, etc.

Des outils d'ETL : Extraction Transformation Loader (chargement) servent à automatiser ces actions entre différents systèmes.

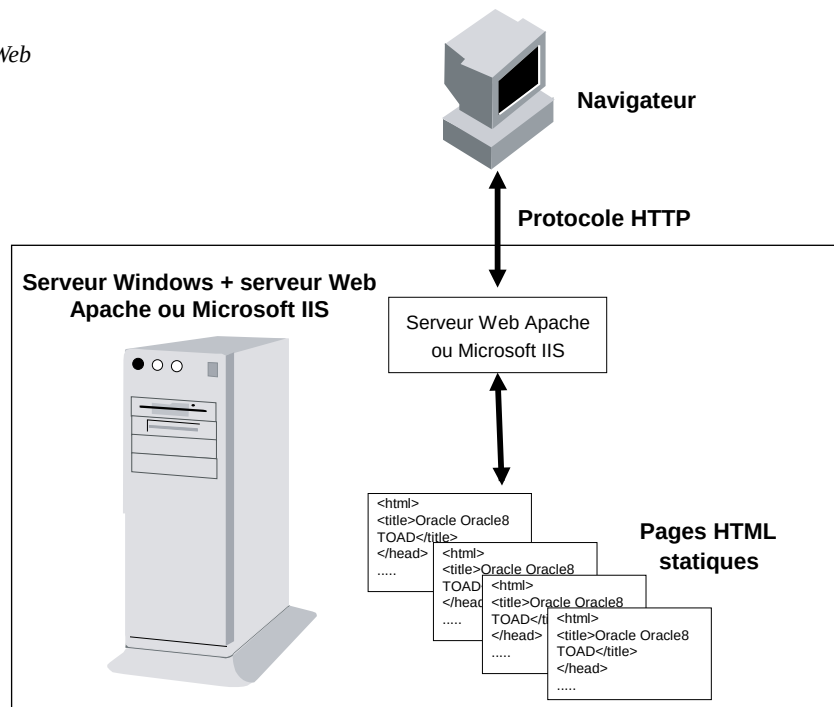
Oracle 10g sous Windows constitue alors une solution simple pour bâtir rapidement et à coût réduit une solution de type infocentre, datawarehouse ou data marts. Les outils décisionnels modernes accèdent à la base Oracle 10g en client-serveur ou par interface Web. Le serveur Windows peut voir ses performances amoindries par la charge des interrogations sans que cela n'affecte pour autant le serveur de données de production.

Serveur Internet/intranet

Les sites Internet et intranet se multiplient. Le couple Windows-Internet Information Server (IIS) ou Apache est l'un des vecteurs du succès de Windows. Nombre d'applications ont été développées sous cet environnement. Vous pouvez proposer des pages Web statiques, dont le contenu ne change pas, ou dynamiques, dont le contenu extrait d'une base de données à l'instant de l'interrogation reflète fidèlement les modifications subies par celle-ci.

Figure 1-8

Consultation de pages Web
statiques



Leur finalité est d'apporter information et connaissances à ceux qui les consultent. Vous pouvez extraire des données de la base Oracle 10g et les mettre à disposition sur un intranet ou sur Internet. Nous revenons sur les techniques de connexion d'IIS avec Oracle au chapitre 16, *Accéder à Oracle depuis Windows*.

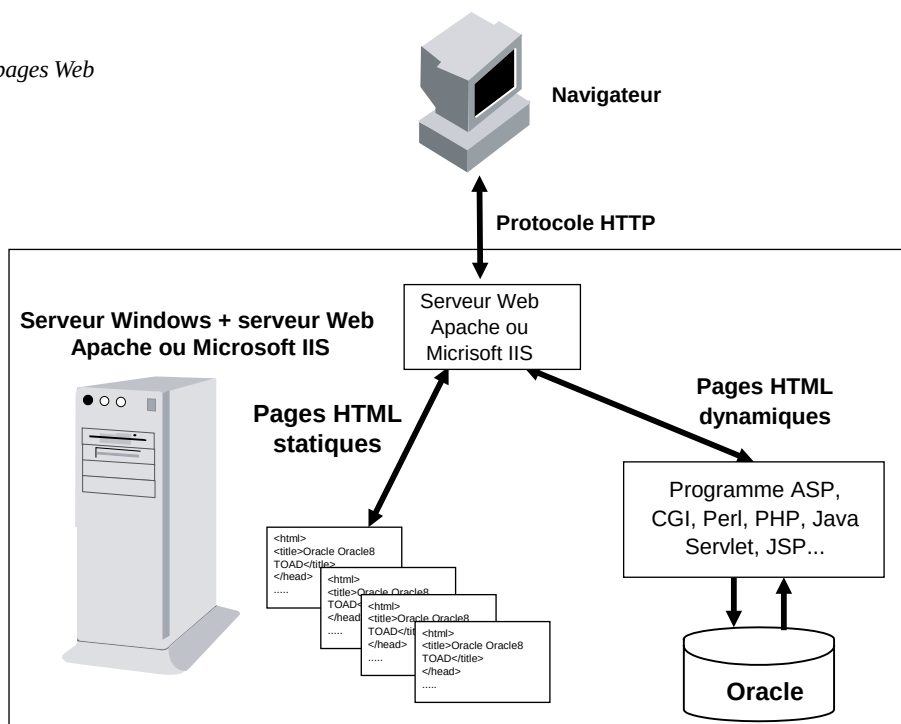
Cependant, comment se contenter de pages affichées « statiques », qui ne reflètent pas dynamiquement le contenu de la base de données ?

L'accès depuis Internet à des données issues de bases de données a été un élément décisif dans l'engouement autour du Web. Windows n'est pas en reste : de nombreux logiciels permettent d'accéder à une base de données. Leur principe est toujours le même, des programmes ou des modules s'intercalent entre le serveur Web IIS ou Apache et la base Oracle 10g, cette base peut être locale ou distante. En ce qui concerne les programmes, il peut s'agir de scripts ASP, Perl, CGI, PHP ou encore des Servlet et JSP Java. Le choix est large et la stabilité prouvée. Oracle propose ces techniques via son propre serveur Web, iAS (Oracle Internet Application Server).

Cette combinaison de techniques permet d'extraire les données d'Oracle 10g et de les publier de manière professionnelle et sécurisée sur le Web. Ces techniques présentent l'avantage majeur d'utiliser un outil du marché, un « navigateur » comme Internet Explorer de Microsoft pour visualiser des pages contenant des données extraites d'Oracle 10g.

Figure 1-9

Consultation de pages Web dynamiques



Là encore, l'alliance d'Oracle 10g et de Windows constitue une solution simple pour évaluer, réaliser une maquette ou mettre en production les données sur le Web. Cette architecture extensible est presque sans limite.

Une voie plus simple, Oracle HTML DB est présentée au chapitre 22, *Oracle HTML DB*.

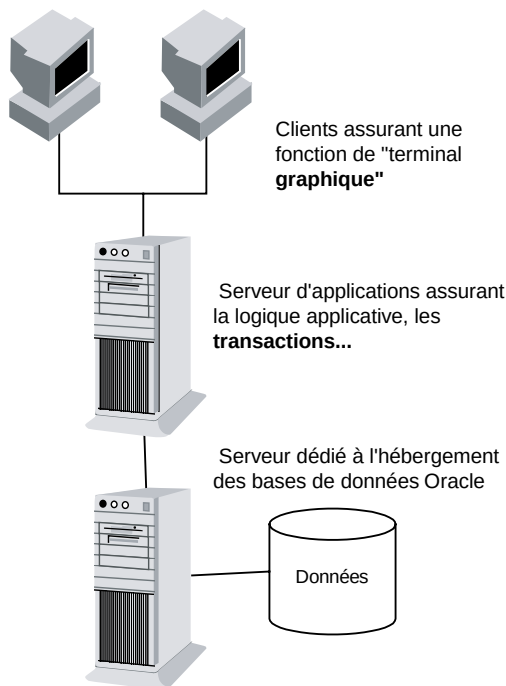
Architecture à trois niveaux

Cette architecture est l'une des plus intéressantes. En comparant le client-serveur et les pages HTML, nous constatons que :

- le client-serveur nécessite des clients puissants, son installation et son déploiement sont lourds ;
- l'interface cliente rendue possible techniquement par les pages HTML est limitée ;
- le standard HTML connaît de nombreuses évolutions, toutes ne sont pas compatibles entre elles. C'est pourquoi certains sites Internet ont des développements spécialisés en fonction des types de navigateurs Internet qui y accèdent.

Figure 1-10

Principe d'une architecture à trois niveaux



Pour supprimer ces inconvénients, il convient de redéfinir le rôle de chaque composant d'une architecture informatique :

- le poste utilisateur graphique doit se comporter comme un « terminal », incorporant ou non des capacités de traitement ;

- un serveur intermédiaire assurera la réalisation de tous les traitements, la gestion des connexions des utilisateurs, leur contexte d'exécution, etc. ;
- un serveur de données hébergera la base de données.

C'est la démarche suivie par Microsoft et Oracle, avec des objectifs différents. Pour Microsoft, il s'agit de se focaliser sur des technologies propriétaires : serveurs 2000/XP, Cluster Server, moniteur transactionnel, postes client Windows 2000/XP ou CE et... base de données SQL Server ! Du fait de son absence dans des domaines comme les systèmes d'exploitation ou le poste utilisateur, Oracle propose un assemblage de composants provenant de divers éditeurs. Le couple Linux-Apache avec sa multitude d'interface de programmation vient s'intercaler comme serveur de traitement, notamment grâce à la réelle portabilité des applications développées en Java.

Et la technologie Java ?

Face à Microsoft, quelles sont les ressources des autres acteurs du monde du logiciel pour survivre ? Comment se retrouver autour d'un projet commun tout en restant en situation de concurrence (le barbarisme « coopétiteurs » résume bien la situation) ? Comment réduire l'emprise de Microsoft sur les interfaces client ? Comment exister face au marketing technologique de Microsoft qui, avec ses énormes moyens, « truste » les premières places ? Windows est bien entendu l'une des réponses, mais elle n'est pas unique. Java, avec des objectifs et des ambitions différents, tente de redistribuer les cartes.

La technologie Java peut être pratiquement utilisée à tous les niveaux des architectures évoquées dans ce chapitre. Au niveau du serveur, le langage Java permet de réaliser plus simplement qu'avec un langage C ou C++ des applications Web les plus complexes. Tous les grands serveurs d'application Web (WebSphere d'IBM, WebLogic de BEA, iAS d'Oracle) et de nombreux serveurs libres (Tomcat Jakarta, Jboss) utilisent Java. Enfin, le serveur de données peut intégrer des procédures et fonctions programmées en Java. C'est une possibilité prise en compte dans Oracle 10g.

Java et PHP, quel est le choix d'Oracle ?

En fait, Oracle supporte ces deux langages ! Le serveur d'application d'Oracle est clairement orienté Java, avec toutes ses déclinaisons actuelles. De même, l'outil JDeveloper (très largement concurrencé par le logiciel libre Eclipse) vise lui aussi les développements Java.

Pour autant, Oracle ne peut négliger PHP. Depuis plusieurs années, les serveurs Web Apache livrés par Oracle intègrent PHP. Oracle va maintenant plus loin, en fournissant sur son site <http://otn.oracle.com> de nombreuses ressources PHP. Enfin, courant 2005, Oracle a conclu un accord avec Zend, la société des créateurs PHP, pour optimiser le moteur de PHP et offrir des très hautes performances d'accès à Oracle depuis PHP.

Fidèle à son habitude : choix du système d'exploitation, capacités techniques du serveur, du stockage, des modes d'accès et outils de développement, Oracle ne vous enferme pas dans un cadre strict et vous permet un vaste choix.

Quelle version d'Oracle choisir pour Windows ?

Oracle est votre choix pour Windows. Mais quel Oracle ? Les utilisateurs sont souvent décontenancés devant la multitude des options et des produits proposés par Oracle. Le contenu des différents packages varie suivant les versions et il est parfois difficile de s'y retrouver. Voici quelques éléments d'informations pour vous guider.

La documentation *Oracle Database Licensing Information* disponible dans le pack de documentation d'Oracle 10g détaille le contenu de chaque licence.

Il existe deux grandes familles de produits sur Windows, chacune d'elles ayant des possibilités de plus en plus élaborées, à la manière des poupées russes qui s'imbriquent les unes dans les autres.

C'est la version la plus complète, Oracle Database 10g Enterprise Edition qui sert de support à ce livre. Comme nous n'avons pas abordé les produits supplémentaires proposés par cette version, le contenu du livre est intégralement utilisable même pour la version de base.

Les différentes versions d'Oracle 10g proposées sont :

- Oracle Database Personal Edition;
- Oracle Database Standard Edition One ;
- Oracle Database Standard Edition ;
- Oracle Database Enterprise Edition ;
- Oracle Database Express Edition, la version gratuite.

Une liste commentée de tous les modules et options figure au chapitre 13, *L'installation d'Oracle 10g sous Windows* et à l'annexe 8, Oracle Express Edition.

Plus vous montez dans la gamme de produits, plus vous disposez de fonctionnalités et d'options, mais le prix du produit suit la même progression. Si vous ne voulez rien dépenser, Oracle Express Edition est pour vous, si vous souhaitez obtenir le maximum de fonctionnalités au moindre coût, c'est *Oracle Database Standard Edition One* pour Windows qu'il vous faut. Ce sont son installation, sa configuration, ses fonctionnalités, ses possibilités et ses produits que nous allons aborder et détailler dans ce livre.

Que contiennent les versions d'Oracle 10g pour Windows ?

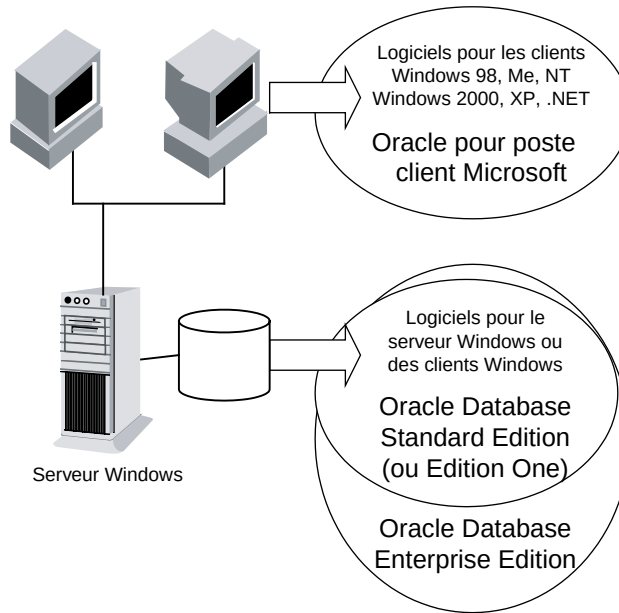
Nous entrons dans le vif du sujet. Avant de nous intéresser à l'installation d'Oracle 10g Server, nous allons en détailler les composants.

Le package d'*Oracle 10g Database* pour Windows ne comprend pas uniquement un moteur de base de données et de logiciels d'installation pour Windows. Il s'accompagne également de tout un ensemble de logiciels pour permettre à des postes clients Windows de se connecter à une base Oracle sur Windows ou sur un autre système. Si vous souhaitez que des postes

Windows 98/Me, 2000/XP ou .NET se connectent en client-serveur à la base Oracle 10g sous Windows, vous devez utiliser les logiciels Oracle 10g client pour l'environnement Microsoft. Vous pouvez télécharger ces logiciels à partir du site <http://otn.oracle.com>.

Figure 1-11

*Logiciels Oracle
nécessaires suivant les
systèmes d'exploitation*



Ces clients peuvent être d'autres postes Windows NT, Windows 2000/XP ou .NET. D'autres types de clients (Linux, Unix, Apple...) peuvent également se connecter à la base Oracle sur Windows, mais leurs logiciels ne sont pas fournis dans ce package. Il faudra donc les acquérir ou les télécharger séparément.

Résumé

Ce chapitre a permis de positionner Oracle 10g et Windows dans leur contexte :

- sous quelles conditions comparer Windows et les autres systèmes d'exploitation, les enjeux et objectifs respectifs ;
- les besoins couverts par Oracle 10g sous Windows ;
- l'intégration entre Oracle 10g et Windows ;
- les différentes versions d'Oracle 10g.

Les différentes utilisations possibles d'Oracle 10g ont été largement abordées et commentées.

2

Les nouveautés d'Oracle 10g

Dans ce chapitre :

- Les nouveautés apportées par Oracle9i et Oracle 10g ;
- Les chapitres qui abordent ces nouveautés.

Chaque nouvelle version d'Oracle apporte un grand nombre de nouveautés. Elles sont présentées et commentées dans ce chapitre. Pour chaque nouveauté, la version concernée Oracle9i ou Oracle 10g est indiquée, ce qui vous permet d'identifier rapidement les nouveautés majeures, la version concernée et les chapitres qui s'y rapportent.

Les apports d'Oracle 10g

Comme pour beaucoup d'autres logiciels, la sortie d'une nouvelle version d'Oracle s'accompagne d'un fort battage médiatique. Il convient de repérer les véritables nouveautés des apports mineurs.

D'une part, Oracle 10g est **plus complexe, plus puissante et plus robuste** que les versions précédentes. D'autre part, c'est la version **la plus simple à installer, à configurer et à administrer** de toutes les précédentes versions. Vous avez le choix de ne pas utiliser toute la flexibilité (et la complexité) d'Oracle 10g en créant des bases faciles à maintenir et nécessitant peu d'interventions manuelles.

Cette dichotomie est en elle-même une nouveauté majeure : elle permet à Oracle 10g d'être à la fois plus robuste et plus simple. La plupart des nouveautés d'Oracle 10g vont dans cette voie.

Oracle 10g apporte de nombreuses nouveautés qui **permettent d'automatiser presque totalement** l'administration d'une base. Il est important de noter que ces nouveautés sont optionnelles et qu'elles ne sont pas censées remplacer un administrateur de données. Les caractéristiques d'automatisation d'Oracle 10g permettent plutôt à des sociétés ne possédant pas d'administrateur de base de données d'utiliser sereinement des bases Oracle.

Les petites sociétés ne possédant pas d'administrateur se satisferont d'une installation et des caractéristiques d'automatisation standard. Les autres sociétés possédant un administrateur en bénéficieront aussi lorsque le nombre de bases de données à gérer ne permet pas d'optimiser correctement chacune d'entre elles. Quel DBA peut en effet prétendre que toutes ses bases (celles de test, de développement...) sont finement optimisées ? Oracle 10g apporte une réponse globale à toutes ces situations.

Les objectifs annoncés d'Oracle 10g :

- réduire le coût de maintenance ;
- assurer une administration la plus automatique possible ;
- augmenter les performances nécessaires aux applications critiques.

Voyons maintenant les principales nouveautés apportées respectivement par Oracle 10g et Oracle9i, en commençant par le Grid Computing.

Le Grid Computing

Après Oracle8i et 9i (le *i* signifiait Internet), le *g* d'Oracle 10g signifie *Grid*. Vous pouvez regarder dans la documentation technique d'Oracle, vous trouverez très peu d'informations sur le Grid.

Un nouveau concept et très peu de documentation : quelle signification cela a-t-il pour un administrateur de base de données ? Rassurez-vous, c'est l'objet même de cet ouvrage. Le Grid recouvre toutes les synergies qui résultent de l'association d'Oracle et de matériels actuels ou futurs. Une définition simple a été fournie par Larry Ellison, le bouillant PDG d'Oracle, qui a indiqué que le Grid permettait de fournir « ...de la puissance à la demande à l'aide de composants matériels bon marché ».

La tendance de fond arrive : les matériels se standardisent, les coûts baissent. Les architectures Internet sont construites pour la plupart par un ensemble de serveurs PC bon marché. Pourquoi alors ne pas associer plusieurs machines pour obtenir plus de disponibilité, de puissance, gérer de plus gros volumes à l'aide de baies de disques peu coûteuses comme des NAS ? Pour accompagner cette évolution, l'orientation Grid d'Oracle 10g permet :

- de supporter la virtualisation des capacités de traitement ;
- de tirer parti des matériels à faible coût, grâce à des caractéristiques telles qu'Oracle RAC (*Real Application Cluster*) ou ASM (*Automatic Storage Management*) ;

- d'accompagner l'évolution des matériels : Ultra Large Datafiles, les tables externes, Oracle Streams, les tablespaces transportables ;
- une architecture distribuée : Oracle RAC (*Real Application Cluster*), Oracle Streams et les options de distribution des données.

Les technologies nécessaires à la réalisation du Grid ne sont pas toutes disponibles avec Oracle Database 10g, qui constitue une étape d'une orientation annoncée. Au-delà du concept marketing du Grid, Oracle Database 10g contient suffisamment de nouveautés pour faire de cette nouvelle version une mise à jour plus qu'intéressante.

Les options d'automatisation

Vers une maintenance simplifiée (Oracle 10g)

Parmi les nouveautés simplifiant la maintenance d'une base, les principales sont :

- la surveillance automatique de l'utilisation : des informations sont collectées automatiquement pour une utilisation future éventuelle ;
- le traçage automatique des durées des opérations internes : permet l'auto-ajustement des caractéristiques de la base ;
- le tablespace SYSAUX : sépare la structure des données d'utilitaires Oracle (Enterprise Manager, Spatial, etc.) du dictionnaire de données contenu dans SYSTEM (voir plus loin dans ce chapitre) ;
- l'automatisation des tâches d'administration : un ordonnanceur (*scheduler*) interne permet de programmer des tâches d'administration dans des « fenêtres de maintenance ». Par exemple, la collecte automatique des statistiques, pour un fonctionnement optimum de l'optimiseur statistique. Voir AWR et ADDM dans ce chapitre ;
- les alertes automatiques : le serveur envoie directement un e-mail lorsqu'un incident survient ou qu'un seuil prédéfini est franchi.

ASM : Automatic Storage Management (Oracle 10g)

L'Automatic Storage Management est un apport majeur. En allouant directement des disques (et non plus des fichiers) à Oracle, ASM autorise pratiquement n'importe quelle approche combinant le mirroring de disques, le stripping, la répartition automatique des données et la tolérance aux pannes. Cette possibilité libère le DBA de tâches fastidieuses généralement dévolues à un administrateur système expérimenté.

Ces points sont détaillés au chapitre 27, *Configurer les disques pour Oracle*

AWR : Automatic Workload Repository (Oracle 10g)

L'Automatic Workload Repository (AWR) est un ensemble de tables associées à des traitements qui capturent automatiquement (à un intervalle d'une heure par défaut) des

informations sur l'activité et les performances : les ordres SQL exécutés, le nombre de lectures disque, d'exécutions, etc. Ces données sont utilisées pour les activités d'auto-maintenance des bases.

Ces points sont détaillés au chapitre 28, *Optimisation et performances*.

AMT : Automatic Maintenance Task (Oracle 10g)

L'Automatic Maintenance Task automatise le rafraîchissement des statistiques lorsqu'elles ne sont plus pertinentes, ainsi que la reconstruction des index lorsque beaucoup de mises à jour les ont désorganisés. Ce sont deux aspects importants pour garantir de bonnes performances car Oracle s'appuie sur les statistiques et les index pour traiter les ordres SQL. Ces tâches courantes pour tout DBA sont planifiées par le nouvel ordonnanceur Oracle 10g Unified Scheduler (US).

Ces points sont détaillés au chapitre 28, *Optimisation et performances*.

ADDM : Automatic Database Diagnostic Monitor (Oracle 10g)

L'Automatic Database Diagnostic Monitor analyse les données provenant de l'AWR, d'une façon identique à celle d'un humain analysant les statistiques issues de STATS-PACK. Tout comme un DBA, ADDM recherche les verrous de contention, les goulots d'étranglement d'entrées-sorties disque, les limites mémoire de la SGA. Mais il l'effectue plusieurs fois par jour et d'une façon exhaustive ! Une fois l'analyse effectuée, ADDM vous propose des recommandations d'optimisation que vous pouvez faire exécuter automatiquement.

Ces points sont détaillés au chapitre 28, *Optimisation et performances*.

AMM : Automatic Memory Management (Oracle 10g)

L'Automatic Memory Management modifie les différentes zones mémoire contenues dans la System Global Area (SGA). Contrairement à une allocation statique, AMM privilégie l'allocation dynamique des zones mémoire les plus utilisées, dans la limite de la mémoire totale allouée à la SGA. En exploitant les données de l'AWR, AMM adapte la taille de la Shared Pool et des buffers de données suivant la charge du système.

Ces points sont détaillés au chapitre 28, *Optimisation et performances*.

STA : SQL Tuning Adviser (Oracle 10g)

Le SQL Tuning Adviser travaille en liaison avec l'Automatic Tuning Optimiser (ATO) pour analyser l'historique de la charge des ordres SQL, issus des données de l'AWR. Il fournit des recommandations pour construire de nouveaux index, des materialized views qui réduiront les entrées-sorties disque des ordres SQL concernés.

Ces points sont détaillés au chapitre 28, *Optimisation et performances*.

Server Generated Alerts (Oracle 10g)

Le Server Generated Alerts, en liaison avec l'Unified Scheduler (US), adresse des e-mails lorsqu'un problème perturbe les performances ou le fonctionnement d'Oracle. Par exemple, un système de fichier plein ou trop peu de mémoire allouée en SGA.

Ces points sont détaillés au chapitre 25, *Oracle Enterprise Manager*.

Installation d'Oracle 10g

Oracle 10g : une installation plus facile (Oracle 10g)

Comparée à toutes les versions précédentes d'Oracle, la version 10g est la plus facile à installer. Par exemple, les produits nécessaires à l'installation d'une base, toutes versions confondues, ont été regroupés sur un seul CD-Rom, évitant ainsi les attentes et les changements laborieux en cours d'installation.

Pour les habitués d'Oracle sous Unix, il n'est plus possible d'installer les logiciels Oracle à partir de l'ancien outil « mode caractère ». Depuis Oracle8i, un logiciel d'aspect identique pour l'ensemble des plates-formes (Linux, Unix, Windows...) est utilisé. Il s'agit de l'OUI (*Oracle Universal Installer*). Oracle 10g apporte ainsi une installation plus simple, plus rapide et plus fiable.

Ces points sont détaillés au chapitre 13, *L'installation d'Oracle 10g sous Windows*.

Nouvelles règles OFA (Oracle 10g)

L'Optimal Flexible Architecture (OFA) propose un ensemble de recommandations pour structurer les répertoires d'installation des logiciels, gérer les versions, l'emplacement des bases et le nom des fichiers et répertoires. Les règles ont évolué avec l'arrivée de la 10g.

Ces points sont détaillés aux chapitres 9, *Les fichiers d'une base Oracle 10g* et 13, *L'installation d'Oracle 10g sous Windows*.

Le moteur Oracle 10g

Nous ne sommes plus à l'époque où Oracle avait besoin d'expliquer son fonctionnement interne pour convaincre et entraîner l'achat d'une licence. Oracle est un leader qui veut convaincre les entreprises par l'utilisation d'arguments stratégiques.

C'est pourquoi Oracle a peu communiqué sur les nouveautés techniques apportées au moteur Oracle 10g. Pourtant, les équipes de développement ne se sont pas endormies et les nouveautés sont nombreuses. Elles vont dans de multiples directions et concernent tous les aspects de la base de données. Pour ne pas nous éparpiller, nous ne citons que les plus importantes, celles qui sont suffisamment généralistes pour intéresser l'ensemble des administrateurs Oracle. Pour chacune d'elles, nous indiquons le chapitre où elle est présentée en détail.

Par souci de compatibilité avec les versions précédentes, peu de caractéristiques techniques disparaissent : seuls des ajouts sont effectués. Pourtant, Oracle insiste fortement sur l'utilisation de certaines nouveautés.

C'est presque l'intégralité de la gestion interne d'Oracle qui a été améliorée avec ces nouvelles versions.

Nouveau tablespace SYSAUX (Oracle 10g)

SYSAUX est un nouveau type de tablespace nécessaire à chaque base Oracle 10g. Précédemment, le dictionnaire de données hébergé dans le tablespace SYSTEM contenait aussi toutes les structures des différents utilitaires Oracle (RMAN, Text, Ultra Search, Data Mining, etc.). Le dictionnaire de la base est maintenant dans SYSTEM et tous les utilitaires Oracle dans SYSAUX.

Ces points sont détaillés au chapitre 23, *Gestion de l'espace disque et des fichiers*.

Nouveau mode de gestion des tablespaces (Oracle 10g)

Pour simplifier et améliorer les performances d'accès aux tablespaces, deux modes de gestion existent depuis Oracle9i : le mode LOCALY MANAGED et le mode DICTIONARY.

Le nouveau mode LOCALY MANAGED stocke tous les aspects d'allocation de segments à l'intérieur de chaque tablespace et les informations sont codées au format *bitmap* dans chaque en-tête de tablespace.

L'ancien mode DICTIONARY centralise l'allocation d'espace dans le dictionnaire de données de chaque base.

Le mode LOCALY MANAGED facilite l'échange de données entre bases en « décrochant » un tablespace pour le « raccrocher » à une autre base. Ceci est très utile pour, par exemple, mettre rapidement à disposition d'une base de test la réplique exacte des données d'une base de production.

Ces points sont détaillés au chapitre 23, *Gestion de l'espace disque et des fichiers*.

Allocation automatique des segments de stockage (Oracle9i)

Le mode LOCAL apporte de nombreuses options d'allocation des espaces disques. Les fastidieuses clauses STORAGE disparaissent au profit d'option comme AUTOALLOCATE.

Ces points sont détaillés au chapitre 23, *Gestion de l'espace disque et des fichiers*.

Les tablespaces temporaires (Oracle9i)

Oracle effectue de nombreux tris à la demande des utilisateurs ou pour son fonctionnement interne. Ils sont principalement effectués dans une zone de tri en mémoire. Si cette zone s'avère trop petite, un espace disque est réservé à cet effet : c'est le tablespace temporaire ou TEMPORARY TABLESPACE.

Ce nouveau type de tablespace TEMPORARY est détaillé au chapitre 23, *Gestion de l'espace disque et des fichiers*.

Les tablespaces UNDO ou d'annulation (Oracle9i)

Un tablespace de type UNDO est un nouveau tablespace hébergeant les segments d'annulation, destiné à remplacer les rollback segments. C'est dans cet espace qu'Oracle gère les données en attente de validation ou d'annulation. L'utilisation d'un tablespace UNDO est beaucoup plus simple que celle de rollback segments.

Ils sont tous deux abordés au chapitre 23, *Gestion de l'espace disque et des fichiers*.

La période de rétention d'un tablespace UNDO (Oracle9i)

Le *Flashback Query* permet d'interroger les données dans l'état où elles étaient plusieurs heures auparavant. Pour cela, on indique au tablespace UNDO une durée de rétention des informations. Il est ainsi possible d'interroger les données de la base telles qu'elles étaient plusieurs heures auparavant.

Ces points sont détaillés au chapitre 23, *Gestion de l'espace disque et des fichiers*.

Réorganisation des objets base ouverte (Oracle9i)

De nombreux objets peuvent maintenant être réorganisés base ouverte, utilisateurs connectés. La réorganisation en ligne des tables, index et tablespaces augmente ainsi la disponibilité des bases. Des assistants permettent d'automatiser ces actions.

Ces points sont détaillés aux chapitres 23, *Gestion de l'espace disque et des fichiers* et 25, *Oracle Enterprise Manager*.

Extension automatique d'un fichier de données (Oracle9i)

Auparavant, l'un des problèmes les plus difficiles à résoudre pour les administrateurs Oracle était la gestion de la taille des tablespaces. Que faire lorsqu'un segment (de données, d'index, de tri, d'annulation) cherche à s'étendre dans un tablespace et ne dispose plus de place dans les fichiers pour y créer un nouvel extent ? L'utilisateur qui déclenchait sans le savoir l'opération se heurtait alors à un blocage.

Oracle permet l'augmentation automatique de la taille des fichiers pour tous les types de tablespaces (données, index, temporaire, UNDO, SYSTEM) qu'ils soient en mode de gestion locale ou dictionnaire.

Ces points sont détaillés au chapitre 23, *Gestion de l'espace disque et des fichiers*.

Les tablespaces transportables (Oracle9i)

Cette caractéristique permet de transférer un tablespace d'une base de données Oracle9i vers une autre base de données Oracle9i. Les utilisations sont multiples : déplacer l'ensemble

des données d'une application d'une base à une autre, créer rapidement un environnement de test identique à celui de production, transférer des tablespaces d'une base de production vers une base d'infocentre, etc.

Oracle 10g apporte encore plus de flexibilité en permettant l'échange de tablespace entre des systèmes d'exploitation différents. Par exemple, une base de production sous Linux et une base de test sous Windows.

Ces points sont détaillés au chapitre 23, *Gestion de l'espace disque et des fichiers*.

Une base peut comporter plusieurs DB_BLOCK_SIZE (Oracle9i)

Une base est toujours créée avec un DB_BLOCK_SIZE unique pour le tablespace SYSTEM. Comme les tablespaces transportables peuvent provenir de bases configurées avec des DB_BLOCK_SIZE différents, Oracle accepte maintenant des tablespaces possédant une valeur différente de celle définie pour son tablespace SYSTEM. L'utilisation des tablespaces transportables est limitée entre bases Oracle9i.

Ces points sont détaillés au chapitre 23, *Gestion de l'espace disque et des fichiers*.

L'exploration des fichiers redo-log : le Log Miner (Oracle9i)

Oracle propose un outil d'analyse du contenu des fichiers redo-log d'une base : le Log Miner. Cet utilitaire analyse de manière séquentielle l'ensemble des opérations réalisées sur une base. Vous pouvez ainsi répondre à des questions telles que :

- Comment visualiser sous forme d'ordres SQL le contenu des fichiers redo-log, c'est-à-dire des actions qui ont modifié les données de la base ?
- Quelles sont les modifications de structure qui ont été apportées à une base ?
- Quels sont les derniers ordres SQL enregistrés dans le fichier redo-log juste avant de rencontrer un problème disque ?
- Quelles sont les modifications des données d'une table ?
- Quelles sont les modifications apportées par un utilisateur, sur plusieurs tables ou sur une table particulière ?

On se rend compte que son usage appartient à deux grandes familles : aller le plus loin possible dans la récupération des actions effectuées sur une base suite à un problème technique d'une part et la surveillance et le contrôle d'accès aux données d'autre part.

L'avantage procuré par Log Miner est que rien ne doit être prévu a priori, tout est analysé *a posteriori*.

Ces points sont détaillés aux chapitres 23, *Gestion de l'espace disque et des fichiers* et 25, *Oracle Enterprise Manager*.

La gestion automatique des fichiers par Oracle (Oracle9i)

La gestion des fichiers proposée par Oracle offre de multiples options pour en faciliter l'administration : accroissement automatique de leur taille, gestion interne, etc.

Oracle9i va plus loin en vous proposant d'assurer à 100 % la gestion de l'ensemble des fichiers composant une base, y compris leur création, suppression, nom, etc. Les paramètres d'initialisation `DB_CREATE_FILE_DEST` et `DB_CREATE_ONLINE_LOG_DEST_n` gèrent la création des fichiers :

- les fichiers liés aux tablespaces ;
- les fichiers redo-log ;
- les fichiers de contrôle.

Ces points sont détaillés au chapitre 23, *Gestion de l'espace disque et des fichiers*.

La suppression automatique des fichiers (Oracle9i)

L'ordre `DROP TABLESPACE` possède maintenant une option permettant de supprimer les fichiers liés au tablespace en même temps que l'ordre de suppression du tablespace. Une option de l'ordre `ALTER DATABASE TEMPFILE` effectue la même opération pour les fichiers liés aux tablespaces de tri.

Ce point est détaillé au chapitre 23, *Gestion de l'espace disque et des fichiers*.

Gestion du fichier d'initialisation (Oracle9i)

Historiquement, toutes les versions d'Oracle ont utilisé un simple fichier texte pour y conserver les paramètres d'initialisation. Avec Oracle9i, ces valeurs peuvent maintenant être gérées automatiquement dans un fichier binaire persistant, le *pfile*. Ce fichier permet (entre autres) d'administrer vos bases à distance via l'interface Web d'Oracle Enterprise Manager (OEM), sans avoir besoin de connaître l'emplacement physique du fichier d'initialisation.

Ce point est détaillé au chapitre 23, *Gestion de l'espace disque et des fichiers*.

Valeurs dynamiques du fichier d'initialisation (Oracle9i)

De plus en plus de paramètres d'initialisation sont modifiables dynamiquement, ce qui signifie que leur modification est immédiatement prise en compte sans nécessiter l'arrêt puis le démarrage de la base.

Cette nouveauté d'Oracle9i prépare l'optimisation automatique des paramètres mémoire d'Oracle (présentée en début de chapitre), apportée par Oracle 10g.

Ces points sont détaillés au chapitre 23, *Gestion de l'espace disque et des fichiers*.

Contrôle dynamique de la mémoire SGA (Oracle9i)

Les paramètres d'initialisation qui dimensionnent l'allocation mémoire d'une instance sont dynamiques. Le paramètre dynamique DB_CACHE_SIZE remplace l'ancien DB_BLOCK_BUFFER qui était statique.

Ces points sont détaillés aux chapitres 23, *Gestion de l'espace disque et des fichiers*, 8, *L'utilisation de la mémoire par Oracle 10g* et 28, *Optimisation et performances*.

Nouvelle syntaxe pour démarrer une instance (Oracle9i)

Une nouvelle syntaxe doit être utilisée pour créer, démarrer, arrêter une instance. Elle s'effectue à partir de SQL*Plus qui devient l'interface privilégiée pour créer une base.

```
sqlplus /nolog
connect system/manager as sysdba
startup pfile=/u01/app/oracle/admin/TEST/scripts/init.ora
```

Ces points sont détaillés aux chapitres 10, *Démarrer et arrêter une base Oracle 10g* et 24, *Stratégie de sécurité sous Windows*.

Disparition de Server Manager (Oracle9i)

L'utilitaire Server Manager a disparu. Il est remplacé par SQL*Plus qui devient l'interface privilégiée d'administration d'une base.

Ces points sont détaillés aux chapitres 10, *Démarrer et arrêter une base Oracle 10g* et 18, *Les outils d'Oracle 10g*.

Disparition du CONNECT INTERNAL (Oracle9i)

Il n'est plus possible d'utiliser un CONNECT INTERNAL pour se connecter à une base. Les privilèges SYSOPER et SYSDBA qui existent depuis plusieurs versions le remplacent.

```
sqlplus /nolog
connect system/manager as sysdba
startup pfile=/u01/app/oracle/admin/TEST/scripts/init.ora
```

Ces points sont détaillés au chapitre 24, *Stratégie de sécurité sous Windows*.

Utilisation d'un fichier mot de passe (Oracle9i)

Pour démarrer une base en mode client-serveur ou à partir d'Oracle Enterprise Manager, Oracle recommande l'utilisation d'un fichier mot de passe. Il autorise les utilisateurs possédant les privilèges SYSOPER et SYSDBA à administrer une base à distance.

Ces points sont détaillés aux chapitres 24, *Stratégie de sécurité sous Windows* et 25, *Oracle Enterprise Manager*.

Modification du moteur Java (Oracle9i)

Le moteur JServer interne à la base Oracle a considérablement été revu et de très nombreuses possibilités concernant Java et XML sont apparues.

Ces points sont détaillés au chapitre 20, *Java et Oracle 10g*.

Les langages de développement

XML et Java forment un monde en pleine effervescence. Oracle 10g propose de nombreuses évolutions en ce sens. PL/SQL, le langage historique d'Oracle n'est pas oublié.

Nouveautés XML d'Oracle 10g

Oracle a considérablement étendu les fonctionnalités du type XMLType et un nouveau référentiel de stockage XML avec accès par WebDAV, HTTP, FTP a été intégré.

Ces points sont détaillés au chapitre 21, *XML et Oracle 10g*.

Nouveautés PL/SQL (Oracle 10g)

Comme pour chaque nouvelle version, Oracle 10g apporte de nouvelles caractéristiques ainsi que de nouveaux packages (procédures « toutes prêtes ») au langage PL/SQL.

Tout d'abord, le compilateur PL/SQL est **beaucoup** plus rapide qu'auparavant. Oracle revendique une vitesse deux fois plus grande que celui d'Oracle9i R2 et trois fois plus grande que celui d'Oracle8i.

Ces points sont détaillés au chapitre 17, *Programmer avec PL/SQL*.

Nouveautés SQL : les expressions régulières (Oracle 10g)

Pour les habitués d'Unix, les expressions régulières sont accessibles depuis le SQL, directement sur les données d'une base. Ce sont les fonctions REGEXP_*. Elles sont mentionnées pour information, leur utilisation dépasse le cadre de cet ouvrage.

Les nouveaux packages PL/SQL (Oracle 10g)

Les packages PL/SQL sont des « pépites » trop souvent ignorées. Savez-vous que, depuis plusieurs années, vous pouvez effectuer, au moyen du code PL/SQL s'exécutant dans la base, des actions telles que : lire et écrire dans des fichiers texte, réaliser un workflow, exécuter dynamiquement des ordres SQL, piloter (sous Windows) des applications COM, etc. Oracle 10g apporte parmi ces nouveautés :

- **Utl_mail.** Vous pouvez envoyer des e-mails directement depuis un programme PL/SQL, avec ou sans pièce jointe, plus facilement qu'avec utl_smtp présenté dans mes précédents ouvrages (tout le monde s'exclame : « Enfin !!! »).

- **Utl_compress.** Ce package permet de zipper/dézipper des données. Cela permet d'optimiser le stockage de données binaires de type BLOB, RAW ou CLOB.
- **Dbms_warning.** Permet de filtrer les messages d'erreur pour obtenir un traitement plus affiné au sein d'un programme PL/SQL.

Ces points sont détaillés au chapitre 17, *Programmer avec PL/SQL*.

Les outils et assistants

Pour exploiter les nouvelles possibilités d'Oracle, de nombreux assistants sont apparus. Ils facilitent les actions de configuration et d'administration.

Oracle Application Express (Oracle 10g)

Grande nouveauté apportée par Oracle 10g, Oracle Application Express est une suite intégrée de développements d'applications en ligne qui permet de construire des applications Web dynamiques. Il tire pleinement parti de la puissance de la base de données Oracle, et ce à partir de votre navigateur favori.

Oracle Application Express est présenté au chapitre 22, *Oracle Application Express*.

Data Pump Export et Import (Oracle 10g)

Deux nouveaux utilitaires offrent un transfert plus rapide entre bases Oracle. Par rapport aux anciennes versions, Import et Export qui sont maintenues par compatibilité, Data Pump Export et Data Pump Import permettent l'arrêt et la reprise d'un traitement en cours sans nécessairement le reprendre à zéro. Ce pont est très important lorsque l'on travaille sur de très gros volumes.

Data Pump Import et Export sont détaillés au chapitre 18, *Les outils d'Oracle 10g*.

Création d'une base Oracle (Oracle 10g)

Le logiciel DBCA (*Oracle Database Configuration Assistant*) offre une interface graphique conviviale pour vous guider. Cet utilitaire réclamé par tous les administrateurs Oracle depuis des années est disponible depuis Oracle8. Il s'avère très bien conçu et sa dernière version prend en compte toutes les nouveautés apportées par Oracle 10g.

DBCA est détaillé au chapitre 14, *Création d'une base Oracle 10g*.

Paramétrage d'Oracle Net (Oracle9i)

Oracle Net correspond à l'ancien SQL*Net ou Net8. L'assistant NETCA (*Oracle Net Configuration Assistant*) permet de créer rapidement une configuration client-serveur

opérationnelle. L'assistant NETMGR (Oracle Net Manager) accède à toutes les possibilités de paramétrage d'Oracle Net.

Ils sont détaillés au chapitre 15, *Oracle Net, le middleware Oracle*.

OEM : Administrer votre environnement Oracle (Oracle 10g)

Oracle Enterprise Manager, la console d'administration graphique d'Oracle, a considérablement évolué : Oracle 10g lui apporte un fonctionnement 100 % à distance par interface Web, via votre navigateur favori. Elle possède de nombreuses nouvelles possibilités que je vous conseille d'explorer.

Ces points sont détaillés au chapitre 25, *Oracle Enterprise Manager*.

Résumé

Ce chapitre a abordé succinctement les nouveautés d'Oracle 10g. Plus qu'une simple évolution, la gestion interne de la base a été revue à 100 %. Pour simplifier l'installation et l'administration de toutes ces nouvelles options, de nombreux assistants sont apparus.

Ce survol vous permet de vous diriger rapidement vers le ou les chapitres où sont détaillées ces nouveautés.

Comparatif technique Windows/Unix

Dans ce chapitre :

- un comparatif technique entre Windows et Unix ;
- l'exploration de plusieurs facettes techniques.

On peut comparer Windows et les systèmes Unix ou Linux durant de longues heures sans obtenir de résultat probant en raison de la constante partialité de jugements. Nous avons choisi pour notre comparaison d'observer différentes étapes que chaque administrateur rencontre en utilisant une base de données. Nous n'évaluons pas les aspects système d'exploitation ou performance, mais les tâches rencontrées au quotidien sur les deux systèmes par un administrateur Oracle 10g.

Mise en place d'Oracle 10g

Préparation du serveur pour installer les logiciels Oracle

Oracle pour Windows	Oracle pour Unix/Linux
Les prérequis à vérifier avant l'installation sont peu nombreux : création d'utilisateur, espace disque disponible et mémoire principalement.	Les prérequis à vérifier avant l'installation sont nombreux et divers : version exacte du système d'exploitation et des bibliothèques système, création d'utilisateurs et de groupes, espace disque disponible, validations de certains paramètres du système d'exploitation, compilation si nécessaire d'un nouveau noyau, etc.

Préparation des disques pour installer les logiciels Oracle et les bases de données

Oracle pour Windows	Oracle pour Unix/Linux
Les possibilités offertes par Oracle sont identiques quelle que soit la plate-forme, y compris ASM. Windows a comme atout sa simplicité, au détriment de l'ensemble des possibilités offertes par Linux.	Les possibilités offertes par Oracle sont identiques quelle que soit la plate-forme, y compris ASM. Linux possède plus d'options en ce domaine, au prix d'une plus grande complexité de configuration.

Utilisateur du système d'exploitation servant à l'installation et à la configuration d'un serveur Oracle

Oracle pour Windows	Oracle pour Unix/Linux
L'utilisateur Windows qui installe Oracle sur une machine Windows Serveur ou Windows Workstation doit nécessairement appartenir au groupe « Administrateur Local » de la machine. C'est à cette seule condition que des services Windows peuvent être créés.	Par convention, le compte utilisateur Unix servant à installer Oracle se nomme « oracle ». Ce nom n'est pas obligatoire, mais il doit impérativement appartenir au groupe Unix OSOPER ou OSDBA. Ces paramètres figurent dans les fichiers /etc/passwd et /etc/group.

Moteur Java nécessaire aux utilitaires d'installation d'Oracle

Oracle pour Windows	Oracle pour Unix/Linux
Livré sur le CD Oracle 10g pour Windows. Installé automatiquement.	Livré sur le CD Oracle 10g pour Unix. Installé automatiquement.

Ressources mémoire

Oracle pour Windows	Oracle pour Unix/Linux
Les communications inter processus sont gérées par des threads et non des processus. Ces ressources, y compris la mémoire partagée et les sémaphores ne sont pas modifiables par les utilisateurs.	Les ressources permises par le système sont parfois inadéquates pour une base de données importante. Il faut alors modifier les paramètres de mémoire partagée (SHMMAX) et de sémaphores (SEMMNS) puis recompiler le noyau Unix.

ORACLE_HOME : nom donné par convention à l'arborescence par défaut contenant les fichiers et exécutables Oracle

Oracle pour Windows	Oracle pour Unix/Linux
Le fonctionnement est identique, mais les variables ORACLE_BASE et ORACLE_HOME sont dans la base de registre Windows. La gestion sur un même serveur de différentes versions d'Oracle est moins aisée sous Windows que sous Unix. Un utilitaire permet de basculer les variables du registre d'une version à l'autre.	Plusieurs répertoires ORACLE_HOME peuvent être situés sous un répertoire ORACLE_BASE unique. Chaque répertoire ORACLE_HOME contient une version Oracle (8.1.x, 9.x...). Il est possible de modifier le contenu de la variable d'environnement ORACLE_HOME pour utiliser une version précise des logiciels. ORACLE_HOME et ORACLE_HOME sont des variables d'environnement Unix.

Emplacement des VARIABLES ou VALEURS de référence nécessaires au fonctionnement des produits Oracle

Oracle pour Windows	Oracle pour Unix/Linux
Ce sont soit des variables d'environnement positionnées au niveau du système d'exploitation, soit des valeurs stockées dans la base de registre dans l'arborescence HKEY_LOCAL_MACHINE/SOFTWARE/ORACLE. On peut vérifier les variables d'environnement par la commande « set » et consulter les valeurs existant dans le registre avec un éditeur de registre.	Il s'agit de variables d'environnement, positionnées soit dynamiquement à l'invite Unix, soit dans les fichiers d'initialisation de session Unix (<i>.profile</i> , <i>.bash</i> , etc.). La commande <i>env</i> permet de vérifier la valeur de ces variables.

Paramétrage des variables NLS (National Language Support)

Oracle pour Windows	Oracle pour Unix/Linux
On précise l'environnement NLS dans lequel on souhaite travailler en positionnant des variables spécialisées comme NLS_LANG dans la base de registre.	On précise l'environnement NLS dans lequel on souhaite travailler en positionnant des variables spécialisées comme NLS_LANG, comme variable d'environnement ou dans la base de registres.

Installation d'un noyau Oracle

Oracle pour Windows	Oracle pour Unix/Linux
Vérifiez les prérequis comme le niveau de Service Pack. Insérez le CD-Rom d'installation ; l'installateur graphique écrit en Java se lance automatiquement, une fois les classes Java nécessaires à son fonctionnement installées.	Vérifiez l'installation et les versions de certains composants : version du noyau Unix, librairies (exemple : <i>glibc</i>). Insérez le CD-Rom d'installation puis montez-le sous Unix. Lancez l'installateur graphique écrit en Java.

Link obligatoire des exécutables lors de l'installation ou de la modification des options installées

Oracle pour Windows	Oracle pour Unix/Linux
Cette notion n'existe pas sous Windows, puisque tout le système est conçu autour de bibliothèques livrées compilées.	Les librairies partagées sont similaires aux Dynamic Link Libraries (DLL) de Windows. Durant l'installation, les fichiers objet ainsi que les librairies sont linkées pour générer les exécutables Oracle. Un utilitaire graphique automatise cette option. Elle peut toutefois être exécutée manuellement.

Emplacement par défaut des traces concernant le noyau Oracle

Oracle pour Windows	Oracle pour Unix/Linux
Elles sont situées par défaut dans un répertoire respectant les règles de l'architecture OFA.	Elles sont situées par défaut dans le répertoire \$ORACLE_HOME/rdbms/log. L'architecture OFA (<i>Optimal Flexible Architecture</i>) préconise de les placer à l'endroit où tous les éléments concernant l'instance sont regroupés.

Création d'une base Oracle

Emplacement par défaut des fichiers de paramètres des instances

Oracle pour Windows	Oracle pour Unix/Linux
Ils sont situés par défaut dans un répertoire respectant les règles de l'architecture OFA. Les liens symboliques n'existent pas sous Windows.	Ces fichiers se situent par défaut dans le répertoire \$ORACLE_HOME/database. L'architecture OFA préconise de les placer à l'endroit où tous les éléments concernant l'instance sont regroupés. Les liens symboliques sous Unix permettent de faire figurer les fichiers dans un répertoire bien qu'ils soient physiquement placés dans un autre.

Emplacement par défaut des fichiers de données de la base

Oracle pour Windows	Oracle pour Unix/Linux
Ils sont situés par défaut dans un répertoire respectant les règles de l'architecture OFA.	Ils se situent par défaut dans le répertoire \$ORACLE_HOME/database. L'architecture OFA préconise de les placer à un l'endroit où tous les éléments concernant l'instance sont regroupés.

Création d'une nouvelle instance

Oracle pour Windows	Oracle pour Unix/Linux
Le nom de l'instance est limité à 4 caractères.	Il est préférable que le nom de l'instance soit de 4 caractères écrits en majuscules (il est ainsi plus facile de les repérer parmi les processus).

Création des services associés

Oracle pour Windows	Oracle pour Unix/Linux
Chaque base de données doit être associée à un service Windows, créé simultanément à la base de données ou au moyen de l'utilitaire ORADIM. Une fois le service créé, il doit être lancé avant de pouvoir démarrer la base associée. Il peut fonctionner en tâche de fond et être lancé automatiquement si aucun utilisateur n'est connecté au serveur. Par défaut, les services fonctionnent sous le compte SYSTEM qui possède le droit « Administrateur Local ».	Cette fonction est assurée par les démons Unix, qui sont à la base du fonctionnement de ce système d'exploitation. Ils sont lancés automatiquement lors du démarrage de la base de données.

Création d'un lien symbolique

Oracle pour Windows	Oracle pour Unix/Linux
Lacune du système, la notion n'existe pas sous Windows. Microsoft envisage de l'ajouter dans ses futurs systèmes.	Utilisez la commande <code>ln</code> .

Positionnement des variables d'environnement

Oracle pour Windows	Oracle pour Unix/Linux
Les variables d'environnement peuvent être placées dans la base de registre. Pour fixer une variable d'environnement (hors base de registre), il faut saisir la commande suivante :	La notion de base de registre n'existe pas. Pour fixer une variable d'environnement, il faut saisir la commande suivante :
C:\> set ORACLE_SID=TEST	\$ ORACLE_SID=TEST \$ export ORACLE_SID

Création d'une nouvelle base de données

Oracle pour Windows	Oracle pour Unix/Linux
Créez le service associé avec l'utilitaire ORADIM.	Pas d'étape préliminaire.
\$ sqlplus /nolog	\$ sqlplus /nolog
SQL> connect / as SYSDBA	SQL> connect / as SYSDBA
SQL> startup nomount	SQL> startup nomount
SQL> create database TEST ...;	SQL> create database TEST ...;
L'utilitaire graphique Database Assistant (<i>dbassist</i>) existe et automatise toutes ces actions.	L'utilitaire graphique Database Assistant (<i>dbassist</i>) existe et automatise toutes ces actions.

Passage des scripts nécessaires au bon fonctionnement de la base

Oracle pour Windows	Oracle pour Unix/Linux
\$ sqlplus /nolog	\$ sqlplus /nolog
SQL> connect / as SYSDBA	SQL> connect / as SYSDBA
SQL>\$ORACLE_HOME\rdbms\admin\catalog.sql	SQL>\$ORACLE_HOME/rdbms/admin/catalog.sql
SQL>\$ORACLE_HOME\rdbms\admin\catproc.sql	SQL>\$ORACLE_HOME/rdbms/admin/catproc.sql
Etc.	Etc.

Sauvegarde des fichiers d'une base ouverte

Oracle pour Windows	Oracle pour Unix/Linux
Mettre le tablespace OFFLINE.	Mettre le tablespace OFFLINE.
Sauvegarder le tablespace avec l'utilitaire Oracle OCOPI.	Sauvegarder le tablespace avec l'utilitaire Unix de votre choix.
Mettre le tablespace ONLINE.	Mettre le tablespace ONLINE.

Contrôle des instances par le système d'exploitation

Contrôle des Processus et des Threads par un administrateur système

Oracle pour Windows	Oracle pour Unix/Linux
Le mode de fonctionnement de Windows impose que tous les processus soient des fils (ou threads) d'un processus maître. On peut visualiser : - le processus maître <i>Oracle.exe</i> (un par base) et le listener à l'aide du gestionnaire de tâches, - les processus Oracle (comme threads d'<i>Oracle.exe</i>) à l'aide des outils d'administration fournis par Oracle ou des outils système de la console Windows.	Tous les processus sont indépendants. On peut visualiser à l'aide de commandes standard du système d'exploitation, sans recourir à d'autres outils : - la liste des processus <code>ps -ef grep ora</code> , - le listener Oracle <code>ps -ef grep lsnrctl</code> , - la liste des segments de mémoire partagés par <code>ipcs -b</code>. De très nombreux outils (graphiques et mode caractère) permettent de contrôler Unix.

Démarrage automatique des instances Oracle

Oracle pour Windows	Oracle pour Unix/Linux
Il est assuré par le démarrage automatique du service Windows associé à l'instance. Ce point est plus simple sous Windows que sous Unix.	Il est assuré par le script <i>dbstart</i> situé dans le répertoire de démarrage automatique d'Unix qui lance les instances déclarées dans le fichier <i>/etc/oratab</i> (peut dépendre de l'OS utilisé).

Arrêt automatique des instances

Oracle pour Windows	Oracle pour Unix/Linux
L'instance s'arrête automatiquement lors de l'arrêt du service Windows associé, suivant les valeurs des les clés de registre <i>ORA_SID_SHUTDOWNTYPE</i>. Ce point est plus simple sous Windows que sous Unix.	Il est assuré par le script <i>dbshut</i> situé dans le répertoire d'arrêt automatique d'Unix qui interrompt les instances déclarées dans le fichier <i>/etc/oratab</i> (peut dépendre de l'OS utilisé).

Démarrage du listener Oracle Net

Oracle pour Windows	Oracle pour Unix/Linux
Il se lance par la commande : <pre>C :\> lsnrctl start</pre> ou par l'appel du service Windows associé : <pre>C :\> net start OracleTNSListener</pre> Le démarrage du service peut-être synchronisé avec le démarrage du serveur.	Il se lance par la commande : <pre>\$ lsnrctl start</pre> Pour synchroniser le démarrage du listener avec celui du serveur, il faut modifier les scripts de démarrage du serveur pour y incorporer la commande précédente.

Emplacement par défaut des fichiers de configuration Oracle Net

Oracle pour Windows	Oracle pour Unix/Linux
Ils se situent dans <i>ORACLE_HOME\network\admin</i>.	Ils peuvent être placés dans <i>\$ORACLE_HOME/network/admin</i> ou dans l'arborescence Unix (généralement <i>/etc</i>).

Liste des protocoles réseau installés

Oracle pour Windows	Oracle pour Unix/Linux
Vous pouvez lire dans l'outil d'installation Oracle la liste des produits installés. Ce point est peu important car TCP/IP est le protocole réseau qui domine largement les autres.	Pour récupérer la liste des protocoles réseau installés, utilisez la commande <i>adapters</i> suivie du produit considéré : \$ adapters sqlplus

Affichage des messages d'erreur

Oracle pour Windows	Oracle pour Unix/Linux
Lancez le fichier d'aide ou l'aide en ligne au format PDF (Adobe) ou HTML.	Utilisez la commande <i>oerr</i> (<i>oerr</i> = oracle error), suivie du préfixe et du numéro de l'erreur rencontrée : \$ oerr ora 4030 Lancez le fichier d'aide en ligne au format PDF (Adobe) ou HTML.

Résumé

Pour un administrateur système, les connaissances requises et le travail à effectuer sont assez différents selon qu'il opère sur un poste doté de Windows/XP ou d'Unix/Linux.

Pour un administrateur Oracle, les différences rencontrées entre Oracle 10g pour Windows et Oracle 10g pour Linux sont minimales, il utilisera indifféremment les deux systèmes. Les outils graphiques fournis par Oracle, qui étaient l'apanage de Windows, sont maintenant identiques pour les deux systèmes car écrits en Java. De plus, certains outils spécifiques à Windows ont été ajoutés. Ils offrent une intégration maximale à l'environnement Windows XP.

4

Environnement Windows : l'indispensable

Dans ce chapitre :

- un rappel concernant la base de registre et les services Windows ;
- les outils Windows fréquemment utilisés dans le livre ;
- les outils complémentaires.

Dans ce chapitre, nous effectuons quelques rappels concernant la base de registre ainsi que les services Windows. Ensuite, nous passons rapidement en revue l'ensemble des outils standards de Windows qui seront utilisés tout au long de ce livre. Cette liste est loin d'être exhaustive et nous n'avons pas la prétention de vous apprendre à les exploiter. Nous nous contenterons de préciser dans quel contexte ils sont utiles pour administrer une base Oracle.

Puis, nous évoquons quelques utilitaires pratiques pour certaines tâches d'administration, ainsi que leur provenance.

La base de registre

La base de registre utilisée par les différents systèmes d'exploitation de Microsoft est précédée d'une réputation sulfureuse : la documentation Microsoft et Oracle regorge d'invitations à la prudence dès qu'on la manipule. Il est vrai qu'une utilisation malencontreuse du registre peut aller jusqu'à empêcher votre ordinateur de redémarrer. Pourtant, comprendre les informations contenues dans la base de registre et savoir les modifier est

une tâche essentielle pour tout administrateur d'Oracle sous Windows. Le sujet étant très vaste, nous nous limitons à indiquer quelles sont les clés de registre utiles pour Oracle et nous vous indiquons quel est le moyen le plus simple pour les modifier.

Principes de la base de registre

La base de registre de Windows 2000, 2003, XP joue un rôle capital dans le fonctionnement du système d'exploitation. C'est le lieu central où sont conservées toutes les informations touchant la configuration du système comme la séquence d'initialisation, la configuration matérielle, les stratégies de sécurité, les options propres à un utilisateur particulier, le lancement d'applications ou de services.

Précédemment, toutes ces informations étaient stockées dans une multitude de fichiers de configuration se terminant par « .ini ». La base de registre centralise ces informations dans des fichiers lisibles uniquement par certains outils et les interfaces de programmation fournies par Microsoft.

Le registre est organisé de façon hiérarchique. Il se présente sous la forme d'un arbre comprenant cinq branches principales. Suivant l'emplacement de la branche dans l'arbre, on parlera de *clé* ou de *sous-clé*. Aux extrémités, on parlera de *valeurs*.

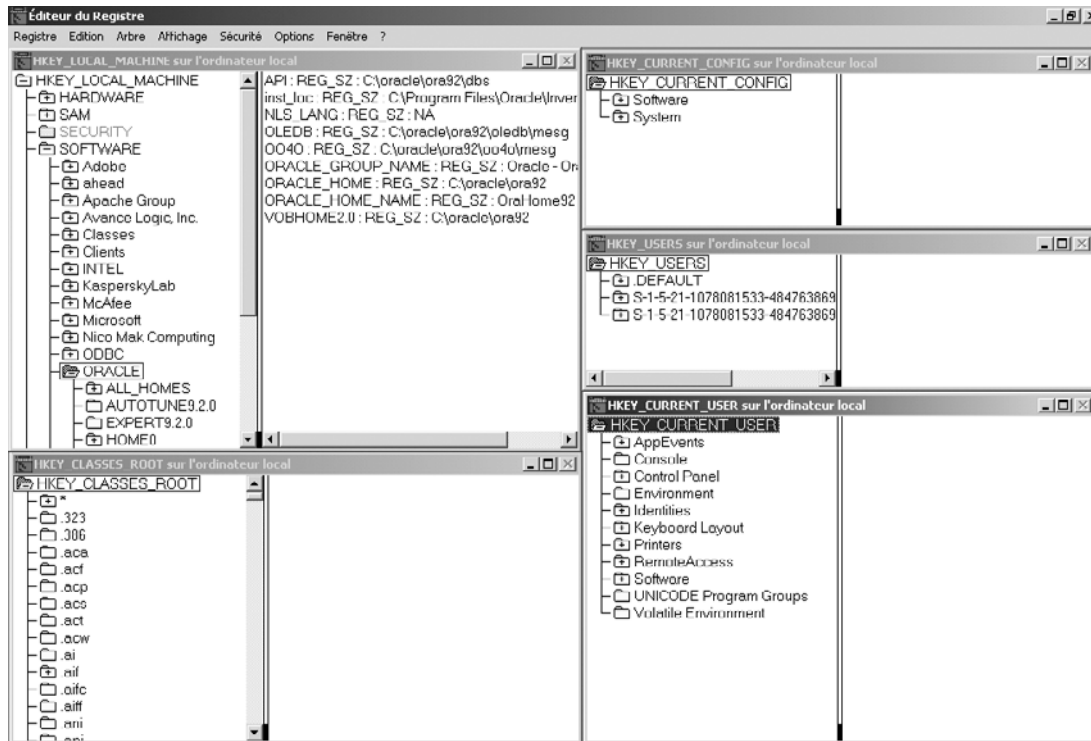


Figure 4-1

Le registre de Windows

Les principales branches du registre sont :

1. **HKEY_LOCAL_MACHINE** ou **HKLM** : contient les sous-clés **HARDWARE** (gestion de la configuration matérielle), **SAM** (gestion des comptes de sécurité), **SECURITY**, **SOFTWARE** (gestion de la configuration des logiciels installés) et **SYSTEM**.
2. **HKEY_CLASSES_ROOT** ou **HKCR** : effectue les associations entre les fichiers et les applications. Contient les informations relatives à **OLE**, **DCOM** et **.NET**.
3. **HKEY_CURRENT_CONFIG** ou **HKCC** : informations générales concernant l'ensemble du système.
4. **HKEY_USERS** ou **HKU** : clé principale contenant une sous-clé attribuée à chaque compte utilisateur sur la machine locale.
5. **HKEY_CURRENT_USERS** ou **hcku** : contient le profil de l'utilisateur connecté. C'est un lien qui pointe vers la clé parente **HKEY_USERS**.

Les paramètres de configuration d'Oracle sont placés en : *HKEY_LOCAL_MACHINE\SOFTWARE\ORACLE*.

Modifier une valeur de la base de registre

Oracle propose des interfaces graphiques pour modifier certaines de ses clés de registre. Pour modifier n'importe quelle autre clé de registre, nous vous conseillons d'utiliser **Regedit**, fourni avec Windows. Ces interfaces ne sont pas « dangereuses », soyez juste conscient des modifications que vous effectuez !

Placez-vous sur une valeur de clé à modifier, par exemple en HKEY_LOCAL_MACHINE\SOFTWARE\ORACLE\KEY_OraDb10g_home1. Sélectionnez la clé dans le panneau droit par un double-clic puis modifiez sa valeur dans l'éditeur de chaînes.

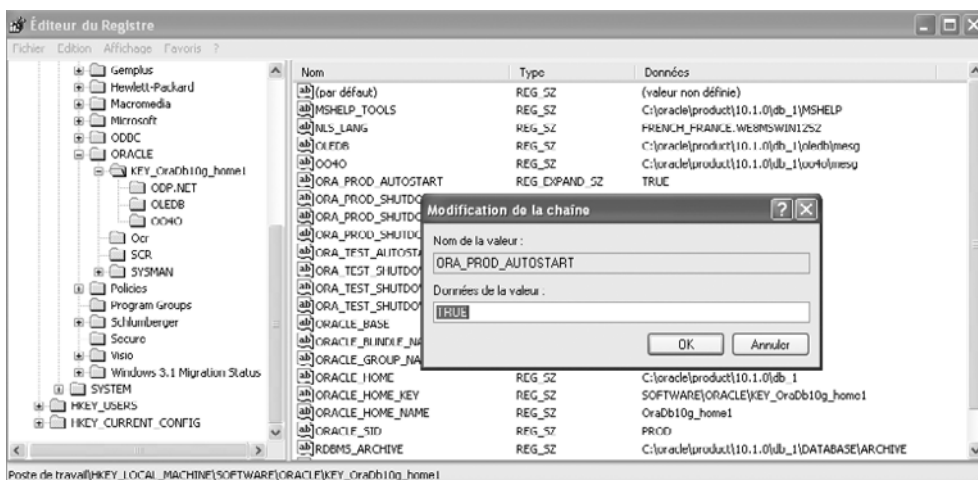


Figure 4-2
Éditer une clé du registre

Les services

Oracle utilise les services Windows pour démarrer automatiquement certains programmes.

Principes des services Windows

Les composants des systèmes d'exploitation Windows s'exécutent sous forme de *services*. Ce sont des programmes spéciaux invisibles pour l'utilisateur. Ils s'exécutent en « tâche de fond » dès la mise en route de l'ordinateur : on peut les comparer à des daemons sous Unix. Les services sont des pilotes de périphérique, des serveurs d'application, le démarrage automatique d'une base de données ou toute autre tâche d'arrière-plan. Ils sont lancés dans le contexte du système local ou sous un compte utilisateur particulier.

Une fois démarré, un service est indépendant des connexions utilisateur : leur fonctionnement ne s'arrête qu'à la demande de l'administrateur ou lors de l'arrêt de l'ordinateur. Certains services, qui ne sont pas indispensables au fonctionnement de l'ordinateur, peuvent être arrêtés et relancés. C'est le cas de la majorité des services utilisés par Oracle.

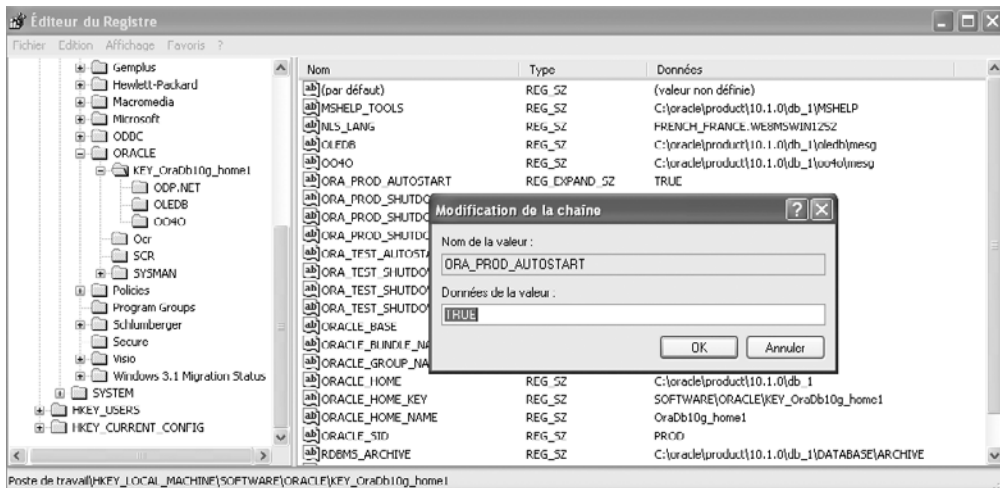


Figure 4-2

Le gestionnaire de services

Microsoft impose qu'un service soit enregistré par le sous-système *service* du système d'exploitation au moyen d'une interface de programmation (API) particulière. Par convention, les paramètres des services sont stockés dans la base de registre en `HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Services` et leurs noms commencent tous par « Oracle ».

L'interface habituelle de Windows pour visualiser, démarrer, arrêter, configurer les services est accessible depuis le menu Paramètres > Panneau de configuration > Outils

d'administration > Services ou, plus directement, à partir d'une console Microsoft Management Console (MMC).

Oracle propose des utilitaires graphiques pour créer, modifier, supprimer les services nécessaires aux logiciels Oracle. Ils limitent les accès directs à la base de registre et sont présentés au chapitre 5, *Oracle 10g sous Windows*.

Démarrer et arrêter un service Windows

Un service peut être lancé automatiquement (en même temps que l'ordinateur) ou manuellement. Dans ce cas, un utilisateur doit se connecter pour lancer manuellement le service. Parfois, il est nécessaire d'arrêter puis de relancer des services.

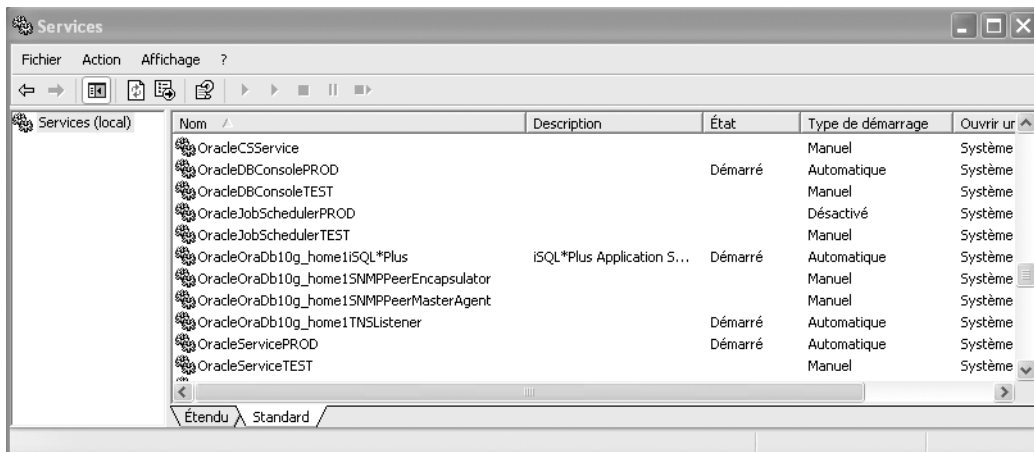


Figure 4-3

Options de démarrage d'un service

Par défaut, lors du démarrage de l'ordinateur, les programmes s'exécutent sous le compte Windows SYSTEM. On peut modifier ce fonctionnement en précisant un compte utilisateur particulier que le gestionnaire de service utilisera pour lancer le service. Nous vous déconseillons d'utiliser cette option qui complique l'administration courante.

Les outils standards de Windows

Explorateur Windows

Nous attirons votre attention sur la possibilité qu'offre l'Explorateur de visualiser les fichiers cachés du système d'exploitation.

Comme Oracle « cache » certains fichiers de cette façon, nous vous conseillons de paramétrer l'Explorateur pour les rendre systématiquement visibles. L'écran suivant présente les options à choisir.

Figure 4-4

Visualisation des fichiers
cachés d'une base Oracle

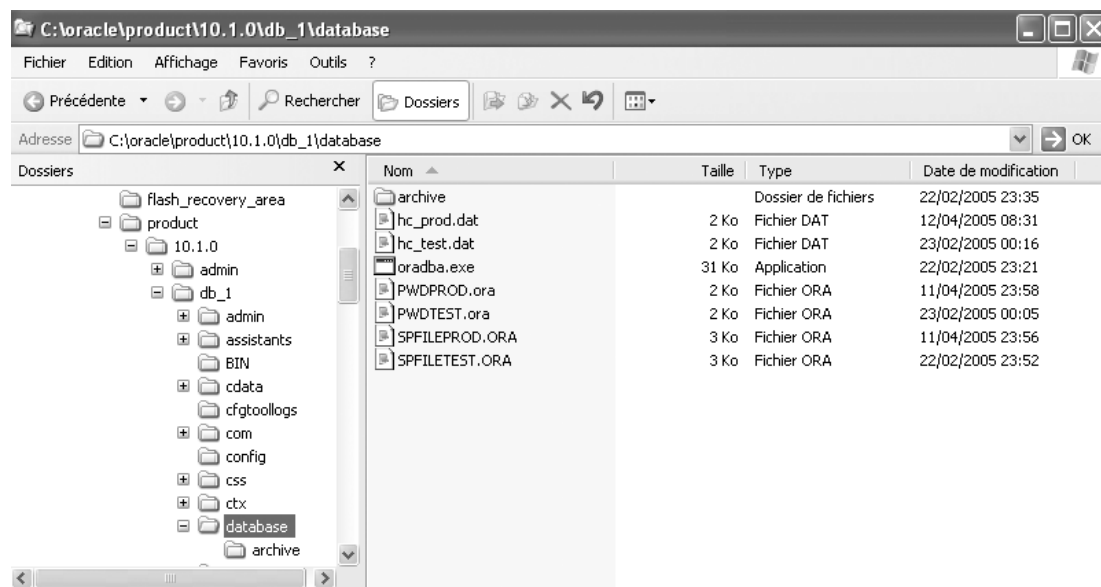
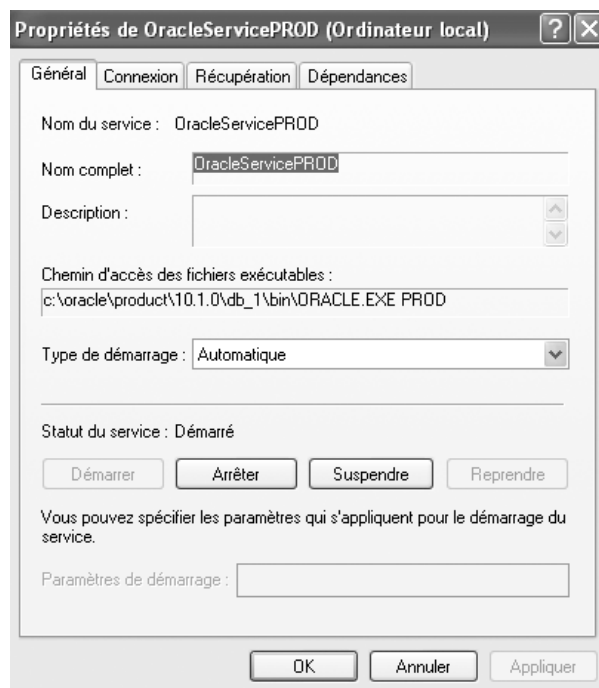


Figure 4-5

Afficher les fichiers cachés

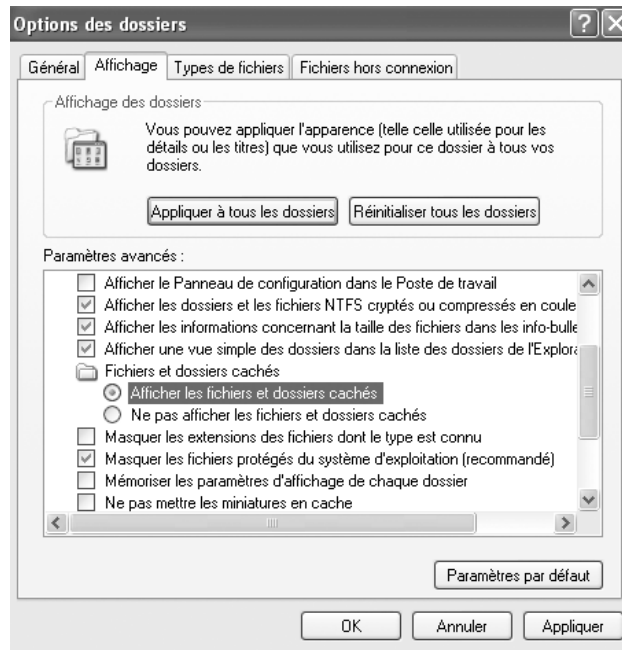
Le Gestionnaire des tâches

Le Gestionnaire des tâches de Windows permet de suivre l'activité de votre ordinateur. Contrairement à l'Analyseur de performances, qui peut se connecter à un poste distant pour en étudier les performances, le Gestionnaire des tâches ne surveille que l'ordinateur local.

Le Gestionnaire des tâches affiche un résumé de l'ensemble des activités de votre machine : les programmes lancés, les processus, ainsi que l'activité processeur, mémoire, pagination, etc.

Lancez le Gestionnaire des tâches avec un « clic droit » de la souris sur la barre d'outils de l'interface.

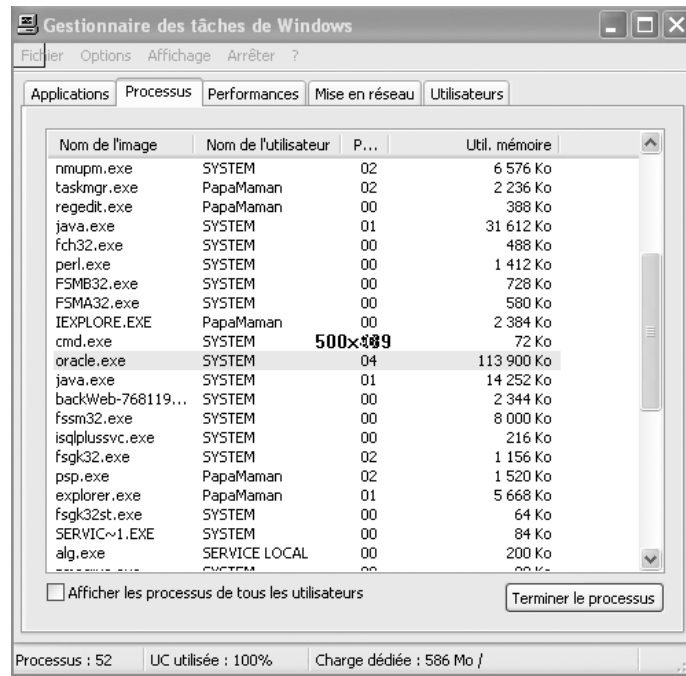
Figure 4-6
Options des dossiers



L'onglet Processus visualise les programmes lancés sur votre machine. Vous pouvez les stopper ou encore en modifier les priorités. Nous vous conseillons d'éviter ces deux options car elles ne respectent ni la procédure normale d'arrêt d'une base Oracle, ni son mode de fonctionnement.

L'onglet Performances est très intéressant. Il vous offre un résumé des consommations processeur et mémoire. Vous avez ainsi des indications précises sur la quantité de mémoire disponible sur votre serveur pour entrer dans les fichiers de pagination, préjudiciables aux performances.

Figure 4-7
Onglet Performances



Microsoft Management Console

Microsoft Management Console (MMC) offre aux administrateurs système une console unique, intégrée, destinée à héberger de nombreux outils d'administration. MMC affiche des consoles que les programmes hôtes appellent des *snap-ins*. Les prochains outils présentés utilisent une console MMC.

Oracle a lui aussi développé des *snap-ins* pour afficher dans une console MMC des compteurs de performances propres à ses bases de données.

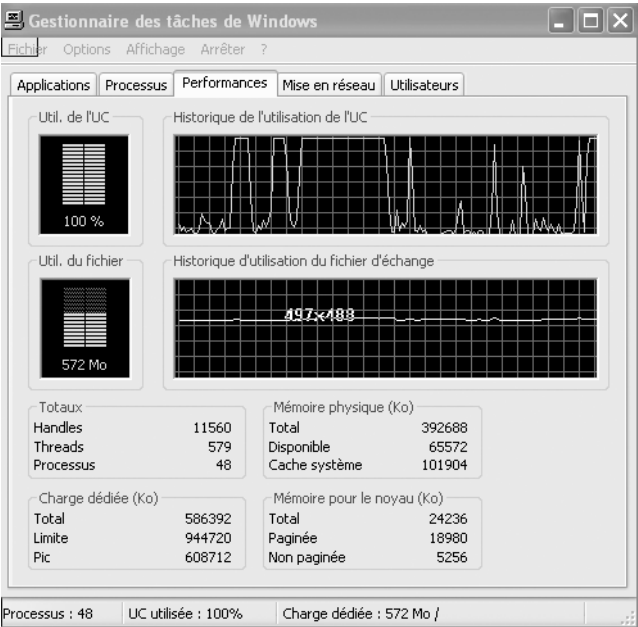
L'Analyseur de performances

L'Analyseur de performances est une console MMC d'administration pour des stations de travail, des ordinateurs locaux ou distants. Des séries de compteurs mesurent de nombreux aspects de l'activité du système : les accès disques, le temps processeur consommé, le nombre de threads actifs, etc.

Les données peuvent être examinées en temps réel sous forme de graphe, enregistrés dans un fichier, et des seuils de consommation au-delà desquels vous serez averti peuvent être définis.

Une autre caractéristique de l'Analyseur de performances est la possibilité de définir de nouveaux compteurs. C'est ce qu'a fait Oracle en ajoutant des compteurs spécifiques à la surveillance d'une instance Oracle.

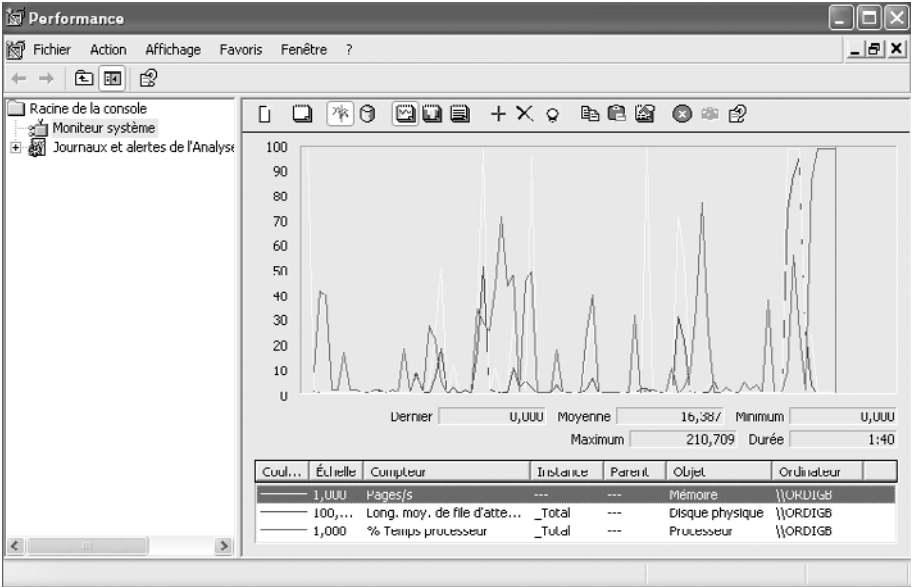
Figure 4-8
Analyseur de performances



La gestion de l'ordinateur

L'outil de gestion de l'ordinateur accessible depuis les outils d'administration de Windows offre une vue synthétique sur de nombreux paramètres.

Figure 4-9
Gestion de l'ordinateur



La possibilité qu'il offre de visualiser un très grand nombre de paramètres au sein d'un même utilitaire le rend très pratique. Il convient de l'examiner en profondeur.

Administrateur de disques

La gestion des disques est particulièrement intéressante dans le cadre d'une base de données, qui nécessite à la fois de l'espace et des performances d'accès optimales. Le chapitre 27, *Configurer les disques pour Oracle* est consacré aux avantages et inconvénients respectifs des différentes architectures.

Comme le montre la figure précédente, la gestion des disques est accessible depuis la gestion de l'ordinateur. C'est un outil très convivial, toutes les commandes étant organisées en menus. Leur résultat est affiché sous forme graphique.

Gérer des disques ou créer des partitions sont des opérations majeures qui doivent être effectuées par un administrateur Windows averti. Même si l'outil prévient du caractère irréversible des actions réalisées, de gros dégâts peuvent être occasionnés par cet utilitaire. Il est indispensable de disposer d'une sauvegarde complète du système avant toute action.

L'Observateur d'événements

Windows centralise dans l'Observateur d'événements toutes les informations « remontées » par votre ordinateur. Il s'agit des messages d'information, des erreurs et de la sécurité. L'importance de chaque message est indiquée.

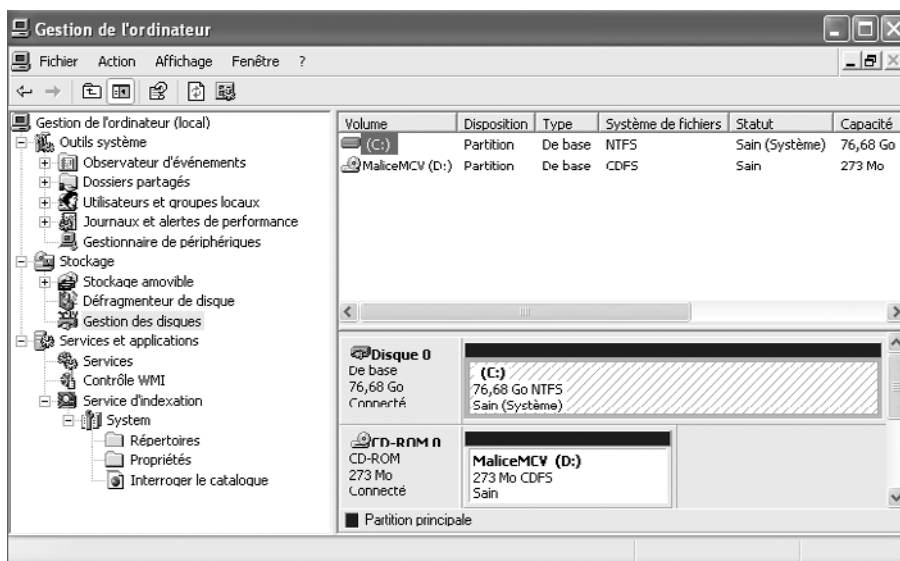


Figure 4-10

Observateur d'événements

Oracle est interfacé à l'Observateur d'événements et lui adresse automatiquement des messages.

Terminal Server

Terminal Server est une caractéristique Windows très intéressante pour un administrateur Oracle. Les services Terminal Server fournissent un accès distant au bureau d'un serveur par un logiciel d'émulation d'écran. Cette option n'est disponible que sur les *serveurs* Windows, l'émulateur étant disponible pour pratiquement toutes les versions de Windows.

Ainsi, il est enfin possible d'administrer un serveur Windows à distance. Vous pouvez installer des serveurs Windows distants et vous y connecter comme si vous étiez derrière l'écran du serveur.

Regedit

Regedit est une interface Windows permettant de visualiser et d'éditer la base de registres de l'ordinateur. Il permet d'effectuer des recherches sur le nom de la clé de registre, en plus de la valeur de la clé et des sous-clés.

Après avoir lancé manuellement les éditeurs de registre par Démarrer > Exécuter, saisir Regedit.

Paramètres régionaux, date et heure

Chaque pays possède ses propres caractéristiques. Un certain nombre de paramètres régionaux sont accessibles à partir du programme *Propriétés des Paramètres régionaux*. Quel impact pour Oracle ? L'heure système ainsi que les fuseaux horaires (changement de l'heure d'été, etc.) définissent le jour et l'heure récupérés par Oracle. La langue d'installation d'Oracle est elle aussi récupérée à partir de ces valeurs.

Les outils complémentaires

Il est utile de connaître l'existence d'utilitaires fournis avec certains outils Microsoft tels que le *Windows XP Service Pack 2 for Advanced Users*, le *Windows Support Tools* ou encore le *Kit de Ressources techniques de Windows*.

Ces outils sont généralement téléchargeables depuis le site de Microsoft. Sa structure étant en constante évolution, je vous engage à rechercher avec un ou plusieurs mots-clés mentionnés précédemment.

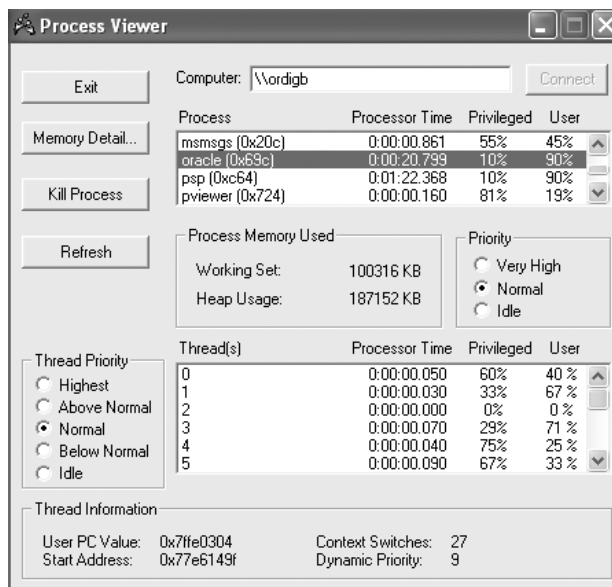
Un site d'exception est à mentionner : www.sysinternals.com. Il propose des ressources techniques très pointues.

Process Viewer

L'utilitaire Pviewer.exe permet d'analyser votre mémoire et vos processus.

Pour Windows XP Professionnel, cet outil accompagne le *Windows XP Service Pack 2 for Advanced Users*. Lors de son installation, la version la plus complète doit être sélectionnée pour qu'il soit installé.

Figure 4-11
Le ProcessViewer



Nous l'utiliserons pour visualiser les threads lancés par une base Oracle dans les différentes phases de son fonctionnement.

Les autres utilitaires

De très nombreux utilitaires ont été développés pour les environnements Windows. Vous les trouverez chez des éditeurs spécialisés ou sous forme de shareware/freeware. Un moteur de recherche comme www.google.com vous aidera à dénicher l'utilitaire de vos rêves.

Résumé

Ce chapitre a permis d'effectuer quelques rappels concernant la base de registre ainsi que les services Windows. Ensuite, nous avons passé en revue l'ensemble des outils standards de Windows qui seront utilisés dans ce livre.

Enfin, nous avons évoqué quelques utilitaires pratiques pour des tâches d'administration. Ils proviennent d'outils Microsoft tels que le *Windows XP Service Pack 2 for Advanced Users*, le *Windows Support Tools* ou encore le *Kit de Ressources techniques de Windows*.

Partie II

Principes de fonctionnement d'Oracle 10g

Chapitre 5 : Fonctionnement d'Oracle 10g sous Windows

Chapitre 6 : Les clusters Oracle

Chapitre 7 : Les processus d'Oracle 10g

Chapitre 8 : L'utilisation de la mémoire par Oracle 10g

Chapitre 9 : Les fichiers d'une base Oracle 10g

Chapitre 10 : Démarrer et arrêter une base Oracle 10g

Chapitre 11 : Transactions et accès concurrents

Chapitre 12 : Le dictionnaire de données Oracle 10g

Fonctionnement d'Oracle 10g sous Windows

Dans ce chapitre :

- une présentation rapide du fonctionnement d'Oracle : les fichiers, les programmes, la mémoire ;
- la notion de base de données et d'instance ;
- l'architecture d'Oracle sous Windows ;
- l'utilisation de la base de registres ;
- démarrer et arrêter Oracle sous Windows ;
- l'utilitaire ORADIM ;
- l'assistant d'administration Oracle pour Windows.

Ce chapitre constitue une introduction indispensable pour comprendre les bases du fonctionnement d'Oracle sous Windows, en vous donnant une vision d'ensemble des interactions entre Oracle et Windows. Il précède d'autres chapitres décrivant plus précisément à la fois le fonctionnement d'Oracle et certains liens avec Windows, mais cette présentation générale demeure essentielle.

La première partie de ce chapitre présente les grands composants d'une base de données Oracle et introduit des concepts et les termes techniques utilisés dans ce livre.

La seconde partie traite des spécificités d'Oracle sous Windows. Nous abordons dans le détail les liens qui unissent le fonctionnement de Windows et Oracle. Des exemples pratiques et des cas concrets sont exposés.

L'utilisation d'ORADIM et de l'assistant d'administration Oracle pour Windows sont détaillées.

Oracle 10g : une seule version

Lorsqu'il fonctionne sous Windows, Oracle 10g possède des caractéristiques identiques à celles d'Oracle sous d'autres systèmes d'exploitation comme Unix ou Linux. Cependant, Oracle adapte son logiciel pour exploiter au mieux les possibilités offertes par chaque système d'exploitation.

L'interface entre Oracle et Windows a été modifiée pour s'adapter et tirer des avantages de l'architecture et des évolutions de Windows, y compris sur les environnements 64 bits. Entre les versions d'Oracle sous Windows et sous Unix, les différences d'architecture ne concernent essentiellement que des aspects très proches des « couches basses » des systèmes d'exploitation. Les ingénieurs d'Oracle ont adapté la base de données sous Windows pour permettre une intégration parfaite autorisant un fonctionnement performant et stable.

Des outils d'administration complémentaires propres à Windows sont venus enrichir la palette des outils Oracle que l'on retrouve sur tous les systèmes d'exploitation. Ainsi, un administrateur Windows ne sera pas dépaycé : il contrôlera tous les aspects liés à Windows en utilisant des outils familiers. De même, un administrateur Unix retrouvera sous Windows des outils connus.

Les composants d'Oracle 10g sous Windows

Avant de les détailler dans les chapitres suivants, nous vous présentons succinctement les différents composants d'une base Oracle 10g, à savoir :

- les composants « fichiers » ;
- les composants « programmes » ;
- les composants « mémoire ».

Les composants « fichiers »

Nous ne présentons que les fichiers qui composent votre ou vos bases de données. Les autres fichiers, mis en place lors de l'installation Oracle, sont décrits au chapitre 13, *L'installation d'Oracle 10g sous Windows*.

Toute base de données se réduit en définitive à des fichiers qui contiennent les données et à d'autres qui interviennent dans le fonctionnement. Chaque base de données possède des fichiers spécifiques, qu'il faut pouvoir identifier et dont il s'agit de connaître les objectifs. Les quatre types de fichiers indispensables au fonctionnement d'une base Oracle sont :

- les fichiers de données ;
- les fichiers redo-log ;
- les fichiers de contrôle ;
- les fichiers d'administration.

Les fichiers de données contiennent les tables, index, procédures, fonctions, etc., ainsi que le dictionnaire de données (conçu lors de la création d'une base de données). Le dictionnaire, géré à 100 % par Oracle, est le cœur de la gestion interne de la base. Les fichiers de données constituent plus de 90 % du volume global d'une base Oracle. Comme se sont les fichiers les plus volumineux, leur taille, leur emplacement et leur paramétrage sont de la responsabilité de l'administrateur de la base de données.

Les fichiers redo-log mémorisent l'historique de tous les ordres modifiant des données ou la structure de la base de données. Ils sont au minimum trois, chaînés circulairement ; ce qui signifie que le premier se remplit, puis le deuxième, le troisième, puis le premier, etc. Ils assurent une mission de sécurité. Ils garantissent que dès que vous validez la modification d'une donnée, elle sera réalisée quoi qu'il arrive (arrêt brutal, panne électrique, etc.). Leur taille est fixée lors de la création de la base de données (modifiable ultérieurement). Leur fonctionnement est géré à 100 % par Oracle.

Les fichiers de contrôle renseignent la base de données sur son état précédent lors du démarrage, de l'arrêt et lors de certaines phases de fonctionnement. Par exemple, lors de son démarrage, leur consultation permet de savoir quel type d'arrêt a eu lieu précédemment : normal, brutal, etc. Ils contiennent par ailleurs les chemins d'accès et les noms de tous les fichiers (base de données et redo-log) qui composent la base de données. Pour démarrer une base de données, il suffit que le logiciel Oracle sache où se trouvent les fichiers de contrôle. Il y puise un ensemble de renseignements qui lui serviront à réaliser l'opération. Leur taille, quelques centaines de kilo-octets maximum et leur fonctionnement sont gérés à 100 % par Oracle.

Les fichiers d'administration servent à l'administration courante d'Oracle. Ce sont par exemple les fichiers d'initialisation utilisés lors du démarrage de la base, les fichiers servant à paramétrer certains outils et enfin les différents fichiers de trace et d'alerte générés par Oracle.

Tous ces fichiers sont utilisés par les composants « programmes ».

Les composants « programmes »

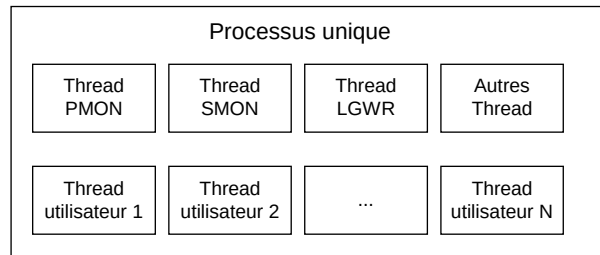
Ces composants « programmes » sont les exécutables qui assurent le fonctionnement de la base de données. Ils réalisent toutes les actions sur les fichiers (base, contrôle, redo-log). Sous Unix, on parlera des « processus » ; sous Windows, d'« exécutables », de « processus » et de « threads ».

L'architecture du système d'exploitation Windows permet à l'exécutable d'Oracle d'être multitâche et multithread. Ce sont ces différents threads qui interagissent avec les composants mémoires, les fichiers et les programmes utilisateurs.

Figure 5-1

Architecture d'Oracle
sous Windows

Architecture d'Oracle sous Windows :
des Threads communicant au sein d'un Processus



Les composants « mémoires »

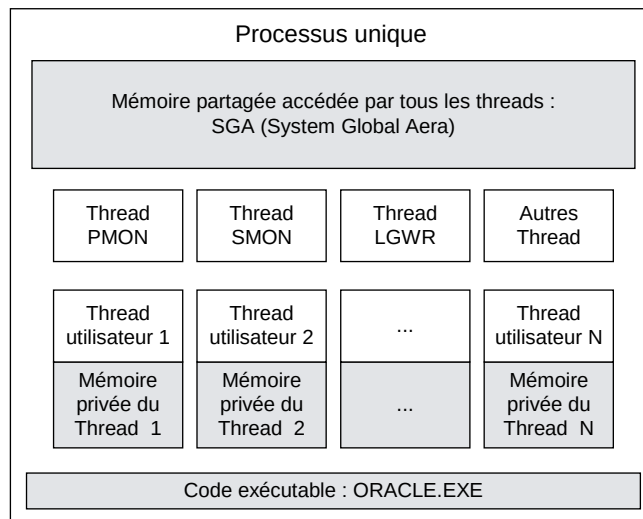
Pour assurer son fonctionnement et ses performances, Oracle a besoin de mémoire pour plusieurs usages :

- une zone mémoire partagée par tous les utilisateurs Oracle ;
- une zone mémoire privée pour chaque utilisateur connecté ;
- une zone mémoire pour exécuter les programmes.

Une zone mémoire partagée (paramétrable suivant votre configuration) est primordiale pour les performances de la base de données : c'est la System Global Area (SGA). Elle est divisée en plusieurs parties qui assurent : le cache des données manipulées, des zones de tris, l'espace pour analyser et réutiliser les ordres SQL déjà exécutés, l'exécution des programmes PL/SQL et Java au sein de la base, etc.

Figure 5-2

Mémoire utilisée
par une instance Oracle



Les autres zones mémoires sont automatiquement allouées pour le fonctionnement de la base de données (la mémoire prise par les exécutables) et par les connexions utilisateur (chaque connexion nécessite un minimum de mémoire).

Base de données et instance

Une base de données Oracle se compose donc de :

- fichiers : base de données, redo-log, contrôle, administration ;
- exécutables : pour faire fonctionner la base ;
- mémoire.

On parlera de *base de données* pour qualifier l'ensemble des fichiers qui la composent.

Une *instance* regroupe la base de données (les fichiers), la zone mémoire allouée (SGA) et les exécutables assurant le fonctionnement de la base. On peut simplifier en disant qu'une instance est une base de données « en action ».

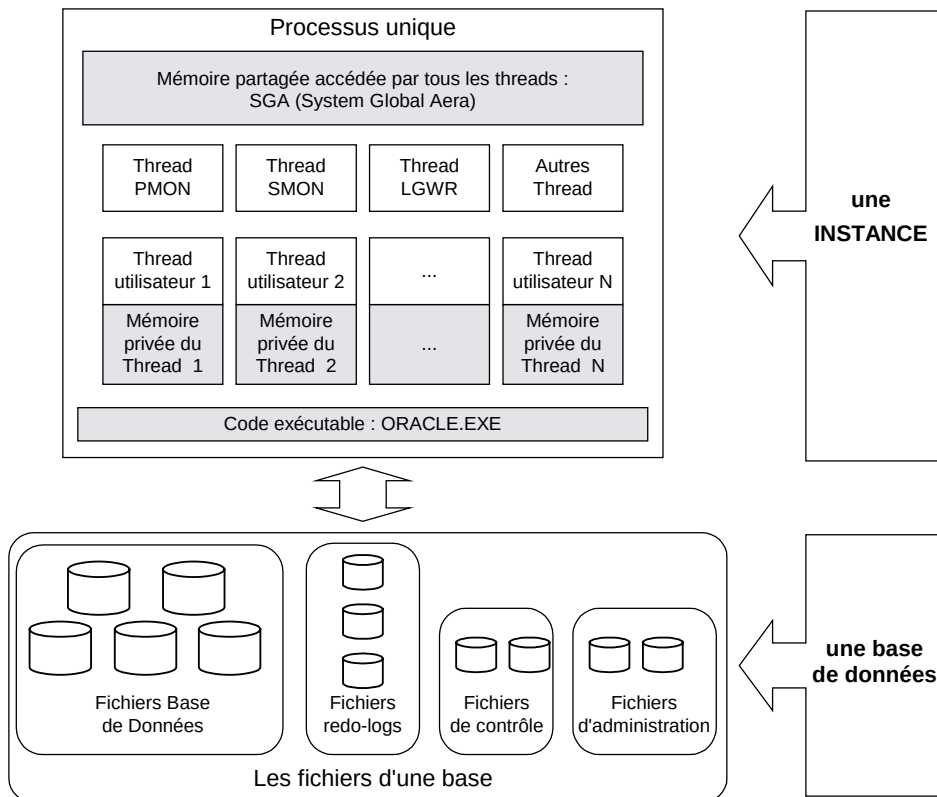


Figure 5-3

Base de données et instance

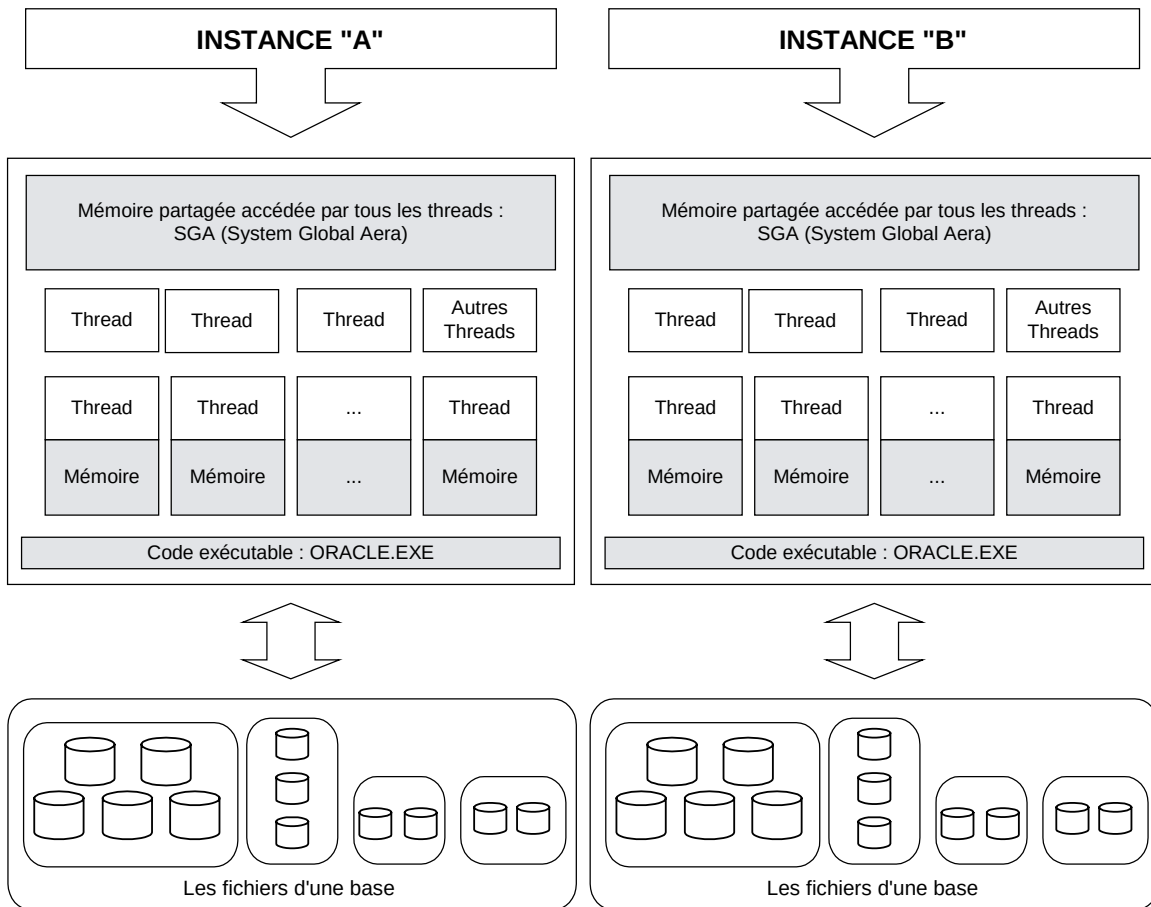


Figure 5-4

Fonctionnement simultané de deux instances sous Windows

Nous aborderons les avantages et inconvénients d'avoir une ou plusieurs bases de données sur un seul serveur Windows au chapitre 14, *Création d'une base Oracle 10g*.

Architecture interne d'Oracle 10g sous Windows

Le serveur Oracle 10g fonctionne sous Windows comme un exécutable unique ayant de multiples threads. Chaque thread a ses propres tâches à réaliser. Certains threads sont dus aux connexions des utilisateurs, d'autres réalisent le fonctionnement de la base de données en tâche de fond.

Les threads internes au fonctionnement d'Oracle 10g sont les équivalents des processus sous Unix. On retrouve des threads équivalents à PMON, DBWR, ARCH, etc., c'est-à-dire

à toutes les grandes fonctions internes du moteur Oracle 10g. Le nombre de threads lancés dépend à la fois des fonctions internes du moteur Oracle 10g (certaines, comme la réplication, sont optionnelles) et du nombre d'utilisateurs connectés.

Cet exécutable Oracle 10g fonctionne sous la forme d'un *service*, car c'est le mode de fonctionnement sur les systèmes d'exploitation Microsoft. Les services associés à une base de données sont abordés ultérieurement dans ce chapitre.

Réservée exclusivement aux environnements Microsoft, l'utilisation des services est l'une des caractéristiques principales d'Oracle sous Windows. La notion de service n'existe pas sous les systèmes Unix et Linux.

L'exécutable oracle.exe

Chaque instance d'Oracle 10g consiste donc en un programme exécutable unique ayant de multiples threads. Pour que chaque nouvelle instance soit identifiée par Windows, son nom figure dans la base de registre sous la forme d'un *service*. Le nom du service sera toujours OracleServiceSID, où *SID* représente l'identifiant de l'instance Oracle. Ce service est associé à un exécutable oracle.exe.

Le service OracleServiceSID se compose initialement de quelques threads, qui ont pour tâche de gérer l'ensemble des threads ultérieurs. C'est sous le contrôle de ce service initial que s'effectuent des opérations telles que le démarrage ou l'arrêt de la base de données.

S'il y a plusieurs instances Oracle sur un serveur, il y aura autant de services OracleServiceSID et d'exécutables oracle.exe lancés que d'instances.

Fonctionnement de l'exécutable oracle.exe sous Windows

Le démarrage d'oracle.exe sous la forme d'un service permet d'être indépendant d'une session utilisateur Windows. Le service et son instance associée peuvent être lancés automatiquement lors du démarrage de Windows.

Il est indispensable que le service soit lancé **avant** de pouvoir démarrer l'instance Oracle. Ces deux opérations peuvent être synchronisées.

Par défaut, les services sont lancés sous le compte Windows Système. Système n'est pas un utilisateur capable de créer une session Windows, il sert à assurer la gestion des services.

La création d'un service associé à une instance est assurée par l'utilitaire ORADIM, présenté ultérieurement dans ce chapitre.

Les différents threads d'oracle.exe

Les administrateurs d'Oracle sur d'autres systèmes d'exploitation connaissent bien les différents processus d'Oracle (PMON, SMON, etc.). Sous Windows, ce sont des threads du processus oracle.exe. Les premiers threads sont utilisés pour la gestion des threads suivants et ne sont pas documentés par Oracle. Les autres threads sont PMON, DBWR,

LGWR, SMON, RECO. Après ceux-ci, il en existe d'autres qui sont optionnels (ARCH, CKPT, etc.). Suivent ensuite les threads créés par les connexions des utilisateurs. Nous commentons plus largement les différents threads au chapitre 7, *Le processus Oracle 10g*.

Thread	Abréviation	Description
Visibles base fermée		Threads gestionnaires des autres threads d'Oracle 10g
Process Monitor	PMON	Il administre les différents threads et processus d'Oracle 10g. Il est chargé de la libération des ressources occupées si une session est anormalement tuée (exemple : un client distant éteint son PC sans se déconnecter d'Oracle).
Database Writer	DBWR	Il est chargé de l'écriture dans la base de données. Il écrit les blocs de données modifiés en SGA (mémoire centrale) dans les fichiers base.
Log Writer	LGWR	Il écrit séquentiellement depuis la SGA dans le fichier redo-log courant. LGWR écrit un enregistrement de petite taille dans le fichier redo-log courant. Cela garantit la possibilité de restaurer la base de données suite à un incident (exemple : coupure électrique).
System Monitor	SMON	C'est le grand « nettoyeur » : il est responsable de la validation d'une base lors de son démarrage. Il vérifie si la base a été arrêtée correctement. Si ce n'est pas le cas, il récupère dans les fichiers redo-log les enregistrements validés (par un COMMIT ou un ROLLBACK) qui n'ont pas encore été écrits en base par Oracle afin de les y écrire. Les autres enregistrements (non validés) sont annulés.
Recover	RECO	Résout les pannes lorsque l'option distribuée est utilisée.
Archivage	ARCH	C'est un thread optionnel. Par défaut, il n'est pas actif. Si le mode ARCHIVELOG est activé, sa mission consiste à archiver (copier) chaque redo-log plein dans un endroit particulier (répertoire ou bande). Les fichiers log peuvent être utilisés pour reconstruire une base de données lorsqu'elle a été perdue.
Checkpoint Dédié	CKPT	Prévient le DBWR de réaliser une opération de <i>checkpoint</i> (littéralement : point de contrôle) sur tous les composants de la base de données. S'il n'est pas présent, ce rôle est tenu par le LGWR.
Autres threads		Se sont les threads dédiés aux utilisateurs connectés à la base de données ou à certaines options d'Oracle.

La liste complète des autres threads est présentée dans ce chapitre. Elle est accessible par l'ordre SQL :

```
select name, description from v$bgprocess.
```

Les « sessions utilisateurs » d'Oracle

Chaque connexion à la base Oracle 10g crée un thread dédié dans l'exécutable oracle.exe qui gère votre instance. Toutes les actions de cet utilisateur connecté seront gérées par ce thread. C'est lui qui dialoguera avec la SGA (pour exécuter les ordres SQL et accéder aux données), et il s'allouera automatiquement de la mémoire (par exemple pour assurer le rôle de « buffer » et vous retourner vos données à l'issue d'un ordre SQL).

Comment identifier facilement la session d'un utilisateur ?

Avec le Process Viewer (présenté au chapitre 4, *Environnement Windows : l'indispensable*), il n'est pas possible d'identifier précisément quel est l'utilisateur Oracle connecté à un thread particulier. C'est grâce aux outils fournis par Oracle que vous pouvez identifier quel thread est associé à une connexion à la base Oracle 10g. Il existe plusieurs outils qui utilisent tout le SQL pour interroger le dictionnaire de données Oracle 10g. Le résultat peut être retourné sous forme textuelle ou graphique, suivant l'outil utilisé :

- ordre SQL lancé « manuellement » depuis SQL*Plus ;
- liste des processus graphiques depuis l'*Oracle Administration Assistant for Windows* ;
- liste des processus depuis *Oracle Enterprise Manager*.

Un exemple d'ordre SQL particulièrement complet est donné plus loin dans ce chapitre.

Oracle et la base de registre

Pour conserver toutes les informations concernant l'installation, le paramétrage d'Oracle et d'outils sur Windows, Oracle les conserve dans la base de registres (ou *registry*) de Windows. Vous pouvez consulter son contenu avec l'utilitaire Windows Regedit ou plus simplement avec l'*Oracle Administration Assistant for Windows* fourni par Oracle. Ce dernier outil n'offre l'accès qu'à un nombre limité de clés de registres utilisées par Oracle.

La façon dont Oracle structure ses clés de registre est détaillée au chapitre 9, *Les fichiers d'une base Oracle 10g*.

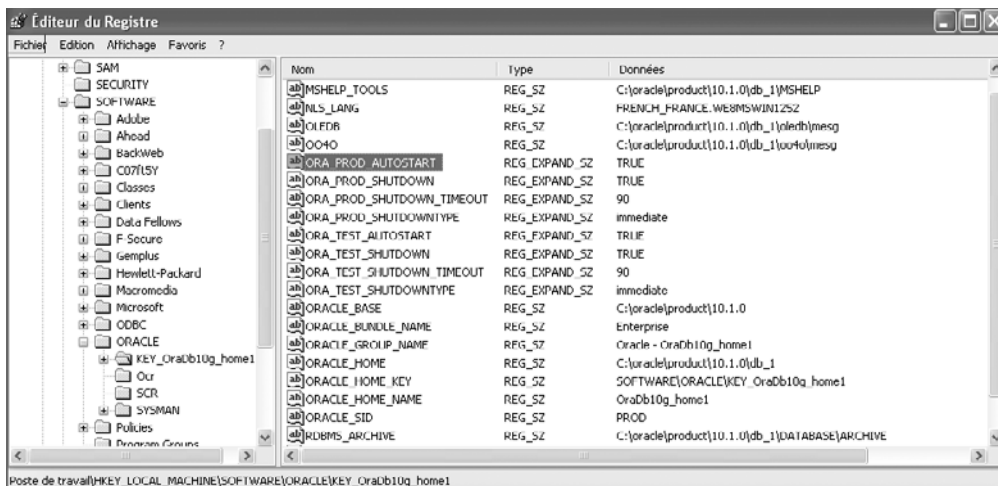


Figure 5-5

La base de registre

L'utilisation du registre par Oracle

Oracle utilise pleinement la base de registre et de nombreux paramètres y figurent. Il est important de comprendre comment Oracle a structuré ses clés de registre. Hormis les services, la majorité des clés de registre utilisées par Oracle sont regroupées sous HKEY_LOCAL_MACHINE\SOFTWARE\ORACLE. Le paramétrage des services est conservé en HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet001\Services.

Ces points sont détaillés au chapitre 9, *Les fichiers d'une base Oracle 10g*.

Plusieurs versions d'Oracle peuvent être installées simultanément sur un serveur. Chacune d'elle sera identifiée par un nom d'Oracle Home différent. On parlera alors du HOMEID associé à l'Oracle Home. Cette sous-clé du registre Oracle contiendra tous les paramètres utilisés par cette version d'Oracle.

Pour chaque version Oracle installée, le numéro ID sera incrémenté en partant de zéro. Par exemple, si les versions Oracle 10.1 et 10.2 ont été installées, leurs paramètres respectifs seront placés sous les clés :

```
HKEY_LOCAL_MACHINE\SOFTWARE\ORACLE\KEY_OraDb10g_home1  
HKEY_LOCAL_MACHINE\SOFTWARE\ORACLE\KEY_OraDb10g_home2
```

Les différentes versions d'Oracle peuvent fonctionner simultanément. Certains programmes ont un nom identique quelle que soit la version, par exemple SQLPLUS, EXP ou IMP. Pour travailler avec l'une ou l'autre de ces versions, Oracle propose un outil graphique, l'*Oracle Administration Assistant for Windows*, qui positionne toutes les variables nécessaires pour aiguiller vers la version à utiliser.

Plusieurs bases de données peuvent fonctionner simultanément. Chacune d'elles possède des caractéristiques qui peuvent être différentes de celles d'une autre base, comme la langue utilisée ou les propriétés de démarrage et d'arrêt. Là aussi, des paramètres propres à chaque base de données sont enregistrés dans le registre.

Pour l'instant, les valeurs essentielles de la base de registre que nous allons utiliser sont :

ORACLE_HOME

Cette variable placée en HKEY_LOCAL_MACHINE\SOFTWARE\ORACLE\KEY_OraDb10g_home1 indique le répertoire racine où sont installés les logiciels Oracle. Ce point est abordé au chapitre 9, *Les fichiers d'une base Oracle 10g*.

ORACLE_SID

ORACLE_SID permet de définir une instance par défaut. C'est à elle que vous vous connecterez par défaut, par exemple avec SQL*Plus. Il est possible de changer cette valeur ou de se connecter à une autre instance :

- en modifiant ORACLE_SID dans le registre avec regedit ;
- depuis l'*Oracle Administration Assistant for Windows* ;

- en utilisant un « alias » Oracle Net pour se connecter en indiquant précisément la base cible (voir le chapitre 15, *Oracle .NET, le middleware Oracle*).

Le service OracleServiceSID

Chaque instance Oracle (identifiée par un *SID*) nécessite la présence d'un service Oracle-ServiceSID, situé en HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet001\Services. Il peut être paramétré (par ordre de préférence décroissant) :

- depuis l'*Oracle Administration Assistant for Windows* ;
- depuis le Gestionnaire de Services Windows ;
- directement, en agissant sur le registre avec Regedit.

Démarrer et arrêter Oracle sous Windows

Dans ce chapitre, nous abordons les *services* Windows, l'instance Oracle, la base de registre. Leur fonctionnement est très fortement imbriqué. Pour vous présenter de manière pratique ces éléments théoriques, nous allons utiliser plusieurs scénarios :

- le démarrage du service Windows ;
- le démarrage de l'instance Oracle ;
- l'arrêt de l'instance Oracle ;
- l'arrêt du service Windows.

Quels utilitaires pour quelle tâche ?

Les utilitaires sont nombreux et parfois l'utilisateur néophyte ne sait lequel choisir par rapport à un besoin bien précis.

Avant d'entrer dans le détail de chacun de ces utilitaires, voyons à quoi ils servent, quel est celui que nous vous recommandons en premier lieu, et ceux qui peuvent vous être utiles en second lieu.

Tâche à effectuer	Choix prioritaire	Choix complémentaire
Créer une base de données	Oracle Database Configuration Assistant (DBCA)	ORADIM et SQL*Plus
Démarrer une base	Gestionnaire de services de Windows	ORADIM SQL*Plus Oracle Enterprise Manager
Arrêter une base	Gestionnaire de services de Windows	ORADIM SQL*Plus Oracle Enterprise Manager
Paramétrer le démarrage et l'arrêt d'une base	sOracle Administration Assistant for Windows	Regedit

Tâche à effectuer	Choix prioritaire	Choix complémentaire
Changer le mot de passe d'une base	ORAPWD	ORADIM
Gérer les utilisateurs d'une base	Oracle Enterprise Manager Console	SQL*Plus Windows Oracle Administration Assistant for Windows
Exporter les données d'une base	Data Pump Export (EXPDP)	Export (EXP)
Importer les données d'une base	Data Pump Import (IMPDP)	Import (IMP)
Charger des données dans une base	Oracle Enterprise Manager Load Wizzard	SQL*Loader (SQLLDR)
Supprimer une base	Oracle Database Configuration Assistant (DBCA)	ORADIM et SQL*Plus
Supprimer le service d'une base	Oracle Database Configuration Assistant (DBCA)	ORADIM
Sauvegarder une base	Oracle Enterprise Manager Backup Wizzard	Recovery Manager (RMAN) OCOPY
Restaurer une base	Oracle Enterprise Manager Recovery Wizzard	Recovery Manager (RMAN) OCOPY

Consultez l'index pour identifier les chapitres où sont présentés chaque utilitaire.

Démarrer et arrêter Oracle sous Windows

Le fonctionnement d'Oracle sous Windows impose que le service Windows soit démarré avant de pouvoir démarrer/arrêter l'instance Oracle. L'ordre imposé est donc :

1. Démarrer le service OracleServiceSID.
2. Démarrer l'instance Oracle SID.
3. Arrêter l'instance Oracle SID.
4. Arrêter le service OracleServiceSID.

Certaines étapes peuvent s'effectuer simultanément. Par exemple, le démarrage du service OracleServiceSID peut entraîner celui de l'instance Oracle. De même, l'arrêt du service OracleServiceSID peut provoquer un arrêt « propre » ou « brutal » de l'instance Oracle. Étudions chacune de ces étapes, spécifiques à l'environnement Windows.

Pour cela, nous allons utiliser plusieurs outils présentés au chapitre 4, *Environnement Windows : l'indispensable*, ainsi que d'autres logiciels fournis par Oracle :

- le Gestionnaire de services ;
- le Gestionnaire de tâches ;

- le Process Viewer ;
- l'assistant *Oracle Administration Assistant for Windows* accessible depuis le menu Démarrer > Programmes > Oracle - OraDb10g_home1 > Configuration and Migration Tools.

Démarrer le service Windows OracleServiceSID

Le premier écran montre comment démarrer le service d'une base Oracle 10g à partir du Gestionnaire des services Windows. Nous utilisons pour cela de la base TEST, créée au chapitre 13, *Installation d'Oracle 10g sous Windows*. Vous noterez que ce service démarre automatiquement lorsque le système Windows est lancé.

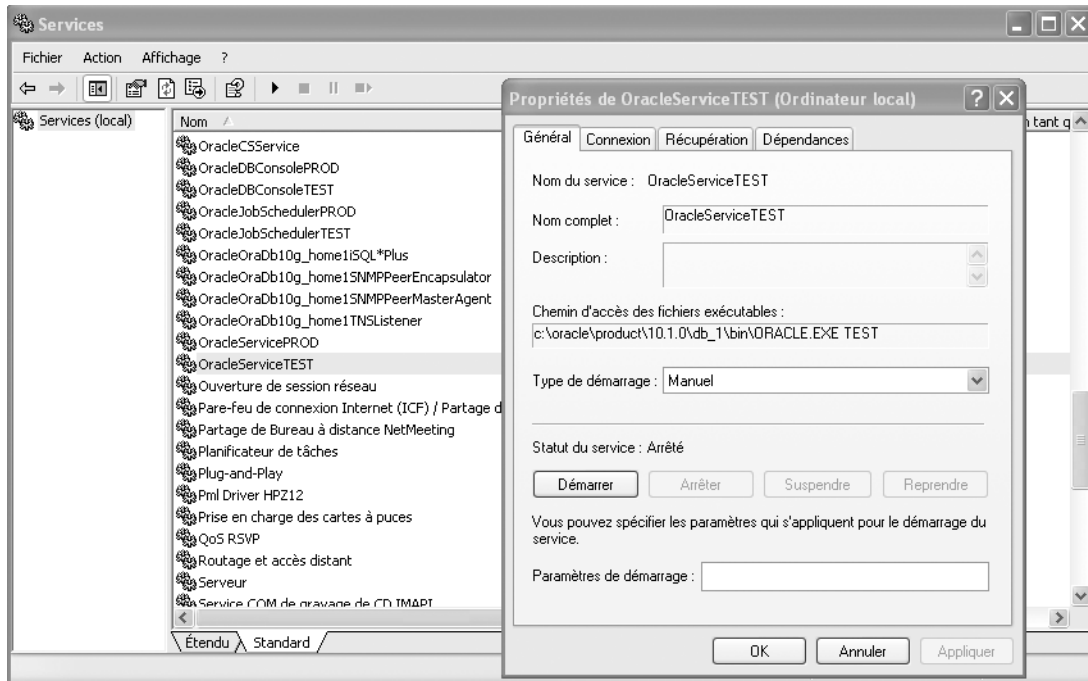


Figure 5-6

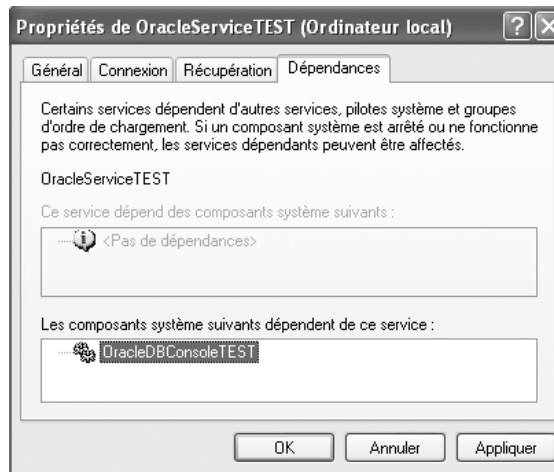
Le service OracleServiceTEST

Comme indiqué précédemment, le service lance l'exécutable oracle.exe en lui passant le nom de l'instance comme paramètre.

Par défaut, il s'exécute sous le compte local SYSTEM de Windows (à ne pas confondre avec le compte Oracle SYSTEM). Il est possible de le lancer sous un compte utilisateur particulier. Comme cela complique l'administration, nous vous conseillons de laisser l'option par défaut.

Figure 5-7

Propriétés du service
OracleServiceTEST



Si vous avez installé Oracle Enterprise Manager Database Control, l'onglet *Dépendances* indique que le service OracleDBConsoleTEST démarre juste après le service OracleServiceTEST. Il permet l'accès à OEM.

Vérifions à l'aide du Gestionnaire de tâches de Windows que l'exécutable oracle.exe, dédié à la gestion de cette instance TEST, a été lancé. Il y aura autant de fois cet exécutable oracle.exe qu'il y aura d'instances Oracle lancées sur le serveur.

Figure 5-8

Compte d'exécution
du service

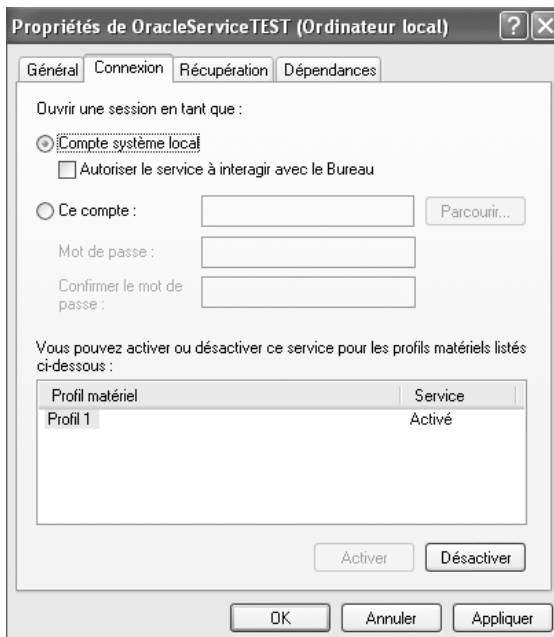
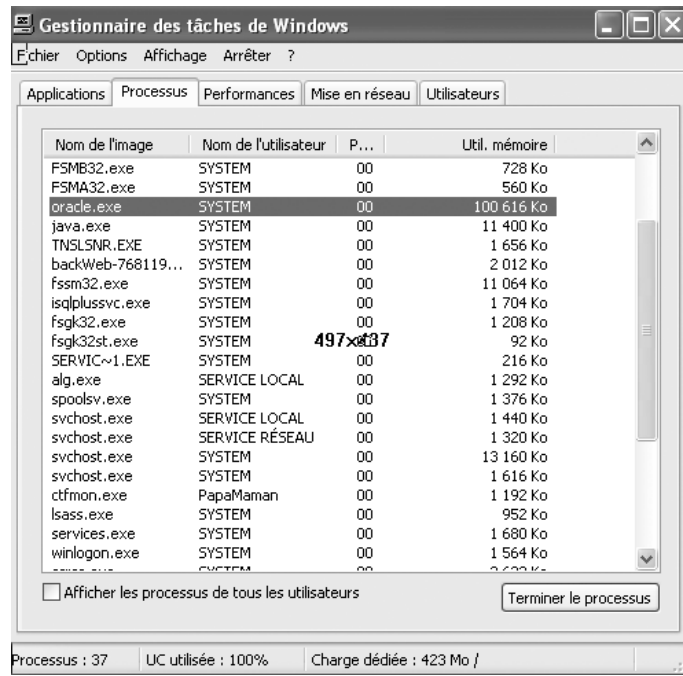
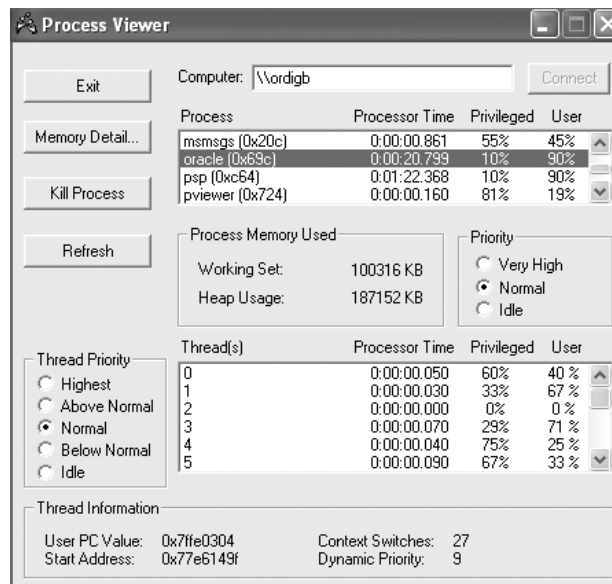


Figure 5-9

Le processus oracle.exe

Avec le Process Viewer (outil fourni avec le *Windows 2000 Support Tools*), nous pouvons entrer dans l'exécutable oracle.exe lancé et visualiser les threads actifs de cette instance.

Figure 5-10

Visualisation des threads avec le Process Viewer

Dans cet exemple, de nombreux threads sont lancés : tout indique que l'instance a été lancée en même temps que le service. Vérifions cette hypothèse avec l'*Oracle Administration Assistant pour Windows* (faites un « clic droit » sur la base de données cible, TEST dans notre exemple).

Vous constaterez que seuls les threads liés à Oracle et ceux liés aux connexions utilisateurs sont affichés. Les autres threads, nécessaires au fonctionnement interne du processus et qui figurent dans le Process Viewer, ne sont pas montrés.

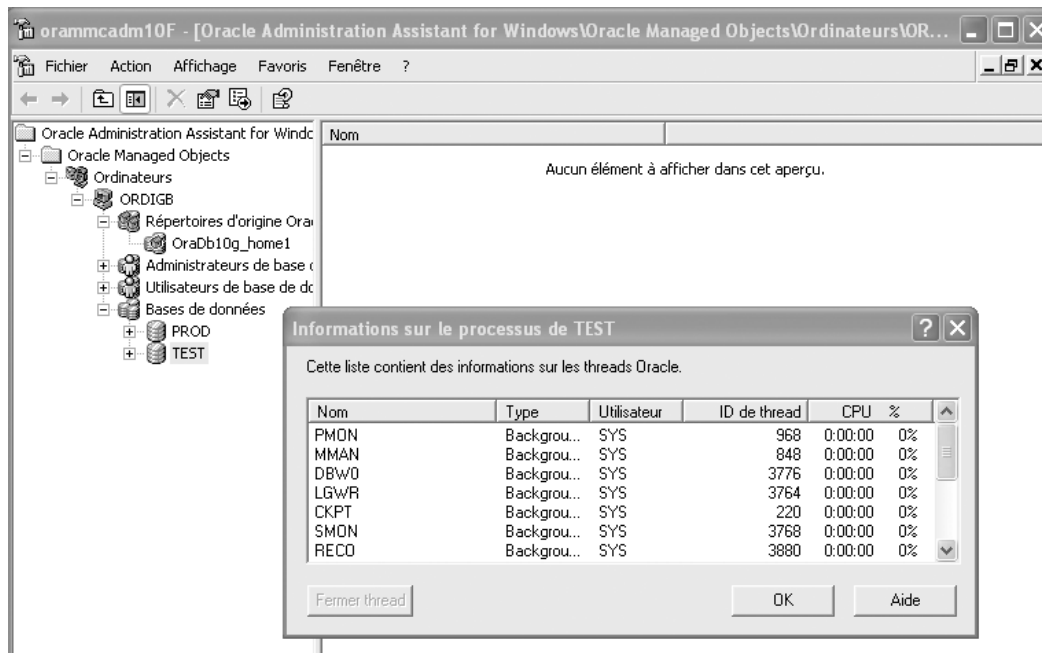


Figure 5-11

Visualisation des threads de l'instance

Nous sommes donc dans le cas idéal, où le service puis l'instance Oracle ont été démarrés. Nous verrons plus loin où se situe ce paramétrage.

Voyons quelles sont les possibilités offertes pour le démarrage du service OracleServiceSID :

Démarrer le service OracleServiceSID

On peut démarrer ce service depuis le Gestionnaire de services ou manuellement, à l'aide de la commande net start.

```
# Depuis une fenêtre Invite de Commandes
# lancement du service de l'instance TEST

c:\> net start OracleServiceTEST
```

Lors de cette étape, rien n'indique si le démarrage de l'instance est synchronisé avec celui sur service ou non.

Démarrer le service OracleServiceSID avec ORADIM

L'utilitaire Oracle ORADIM (présenté dans ce chapitre) gère la création et la suppression des services OracleServiceSID. Il permet aussi de démarrer ou d'arrêter le service et l'instance, en fournissant les paramètres requis.

```
# Depuis une fenêtre Invite de Commandes
# lancement de l'utilitaire ORADIM
# (démarrage du service uniquement)
```

```
c:\> ORADIM -startup -sid TEST -usrpwd oracle -starttype srvc -spfile
```

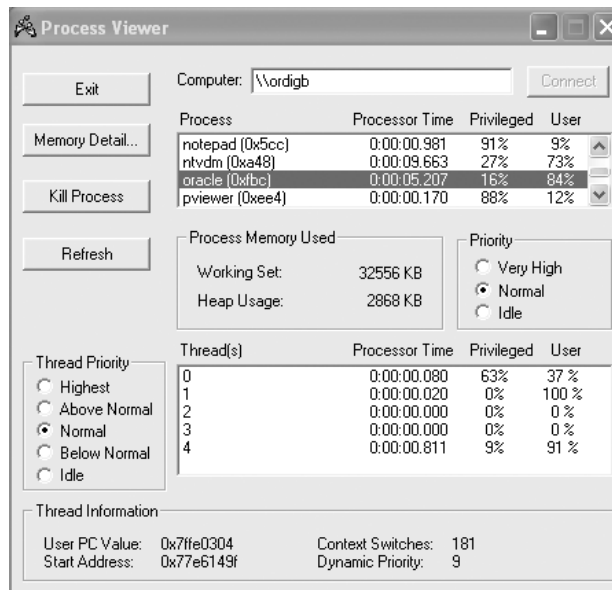
L'utilitaire Oracle ORADIM servant à la gestion des services Windows, il n'existe que sur les systèmes Microsoft.

Démarrer l'instance Oracle

Lorsque le service OracleServiceSID est lancé mais que l'instance associée n'est pas démarrée, la base de données est alors inactive. Dans ce cas, les threads de l'exécutable oracle.exe en attente du démarrage de l'instance sont visibles à l'aide du Process Viewer.

Figure 5-12

Threads d'oracle.exe avant le démarrage de l'instance



Vous pouvez maintenant démarrer l'instance de plusieurs façons :

- en arrêtant puis relançant le service ;
- avec l'utilitaire ORADIM ;
- à partir de SQL*Plus ;
- depuis Oracle Enterprise Manager.

Démarrer l'instance par le service OracleServiceSID

Comme précédemment, il est possible de démarrer le service depuis le Gestionnaire de services ou manuellement, à l'aide de la commande net start.

```
# Depuis une fenêtre Invite de Commandes  
# lancement du service  
  
c:\> net start OracleServiceTEST
```

Comment synchroniser le démarrage du service avec celui de l'instance ?

L'utilitaire *Oracle Administration Assistant for Windows* permet de paramétrer les options de démarrage et d'arrêt d'une instance Oracle par rapport au démarrage et à l'arrêt du service Windows. Sélectionnez la base cible (TEST dans cet exemple) et faites un « clic droit » pour accéder aux options de configuration du démarrage et de l'arrêt du service.

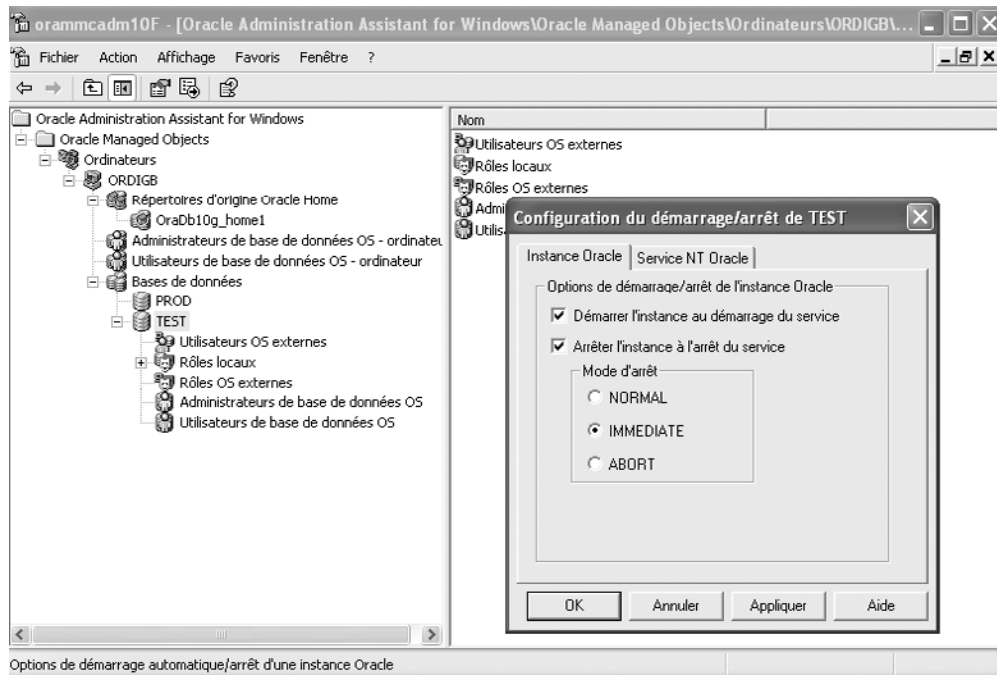


Figure 5-13

Paramétrage du service

L'onglet Instance Oracle synchronise le démarrage et l'arrêt de l'instance Oracle avec celui du service. L'onglet Service NT Oracle reprend les paramètres présents dans le Gestionnaire de services Windows.

Où sont conservés ces paramètres ?

Ces paramètres sont conservés dans la base de registre. Ils sont situés en HKEY_LOCAL_MACHINE\SOFTWARE\ORACLE\KEY_OraDb10g_home1.

Comme plusieurs instances peuvent fonctionner simultanément, le nom de l'instance figure dans chacune des clés.

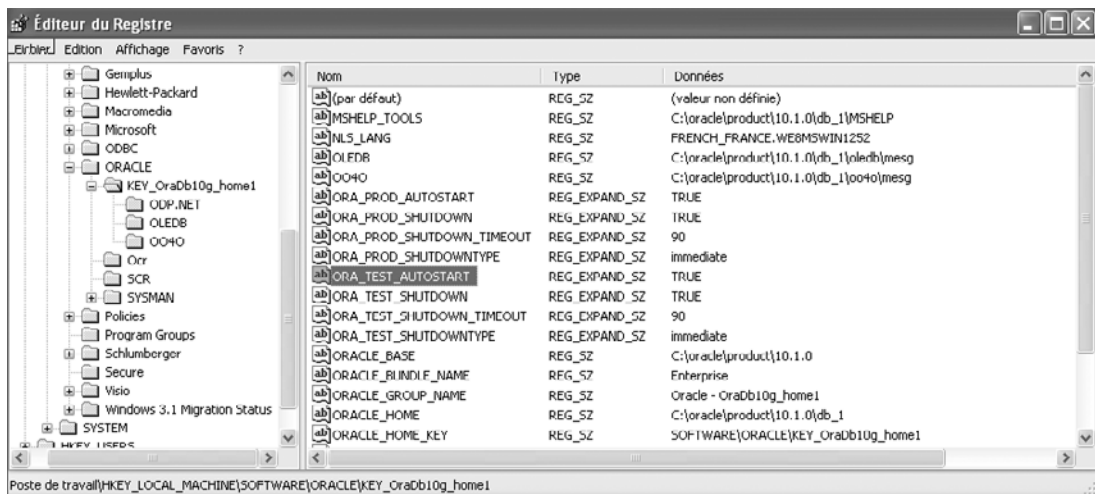


Figure 5-14

Les clés du registre concernées

Démarrer l'instance avec l'utilitaire ORADIM

L'utilitaire Oracle ORADIM (présenté dans ce chapitre) gère la création et la suppression des services OracleServiceSID. Il permet également de démarrer ou d'arrêter l'instance, en fournissant tous les paramètres requis.

```
# Depuis une fenêtre Invite de Commandes
# lancement de l'utilitaire ORADIM (démarrage de l'instance)
```

```
c:\> ORADIM -startup -sid TEST -starttype inst -spfile
```

Démarrer l'instance avec SQL*Plus

Depuis plusieurs versions d'Oracle, l'ancien outil Server Manager a fusionné avec SQL*Plus. C'est maintenant SQL*Plus qu'il faut utiliser pour démarrer et arrêter les bases Oracle : c'est l'outil standard privilégié pour effectuer toutes les opérations sur une base Oracle.

Pour démarrer l'instance Oracle sous Windows, le service OracleServiceSID doit être préalablement lancé.

```
# Depuis une fenêtre Invite de Commandes
# Définissez le SID de l'instance
C:\> set ORACLE_SID=TEST

# lancez SQL*Plus
C:\> sqlplus /nolog
SQL> connect / as sysdba
Connecté.
SQL> startup
Instance ORACLE lancée.
Total System Global Area 135338868 bytes
Fixed Size 453492 bytes
Variable Size 109051904 bytes
Database Buffers 25165824 bytes
Redo Buffers 667648 bytes
Base de données montée.
Base de données ouverte.
SQL> exit
```

Démarrer l'instance avec Oracle Enterprise Manager (OEM)

Oracle Enterprise Manager permet de démarrer et d'arrêter les bases Oracle. Les bases peuvent être locales ou distantes et fonctionner sous Windows ou sous Unix et Linux. Le chapitre 25, *Oracle Enterprise Manager*, lui est consacré.

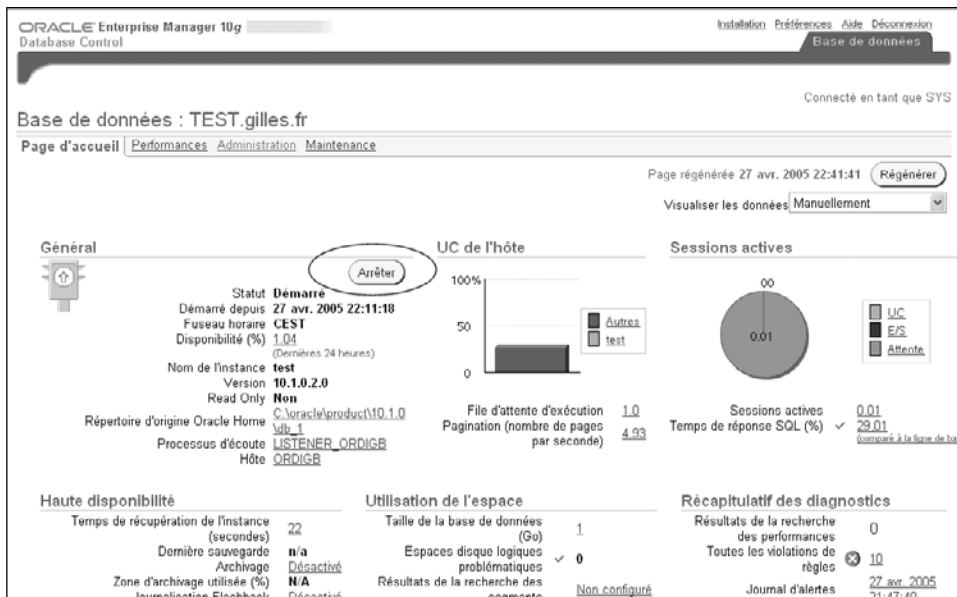


Figure 5-15

Arrêt de l'instance avec OEM

Arrêter l'instance Oracle

L'arrêt de l'instance s'effectue à l'aide des mêmes outils que leur démarrage :

- en arrêtant le service ;
- avec l'utilitaire ORADIM ;
- à partir de SQL*Plus ;
- depuis Oracle Enterprise Manager

Arrêt de l'instance par le service OracleServiceSID

Il est possible d'arrêter le service depuis le Gestionnaire de services ou manuellement, à l'aide de la commande net stop.

```
# Depuis une fenêtre Invite de Commandes
# arrêt du service de l'instance TEST

c:\> net stop OracleServiceTEST
```

Par défaut, l'arrêt du service impose l'arrêt « propre » de l'instance Oracle. Si ce choix n'a pas été fait, l'arrêt du service entraînera la fermeture brutale de l'instance Oracle. Les conséquences sont abordées au chapitre 10, *Démarrer et arrêter une base Oracle 10g*.

Arrêter l'instance avec SQL*Plus

SQL*Plus permet d'arrêter l'instance Oracle sans arrêter le service OracleServiceSID.

```
# Depuis une fenêtre Invite de Commandes
# Définissez le SID de l'instance
C:\> set ORACLE_SID=TEST

# lancez SQL*Plus
C:\> sqlplus /nolog
SQL> connect / as sysdba
Connecté.
SQL> shutdown immediate
Base de données fermée.
Base de données démontée.
Instance ORACLE arrêtée.
SQL>
SQL> exit
```

Arrêter l'instance avec Oracle Enterprise Manager

La console d'administration d'Oracle Enterprise Manager permet de démarrer et d'arrêter toutes vos bases Oracle.

Avant l'arrêt de la base, comme elle fonctionne donc, le service OracleServiceSID est forcément actif à cet instant.

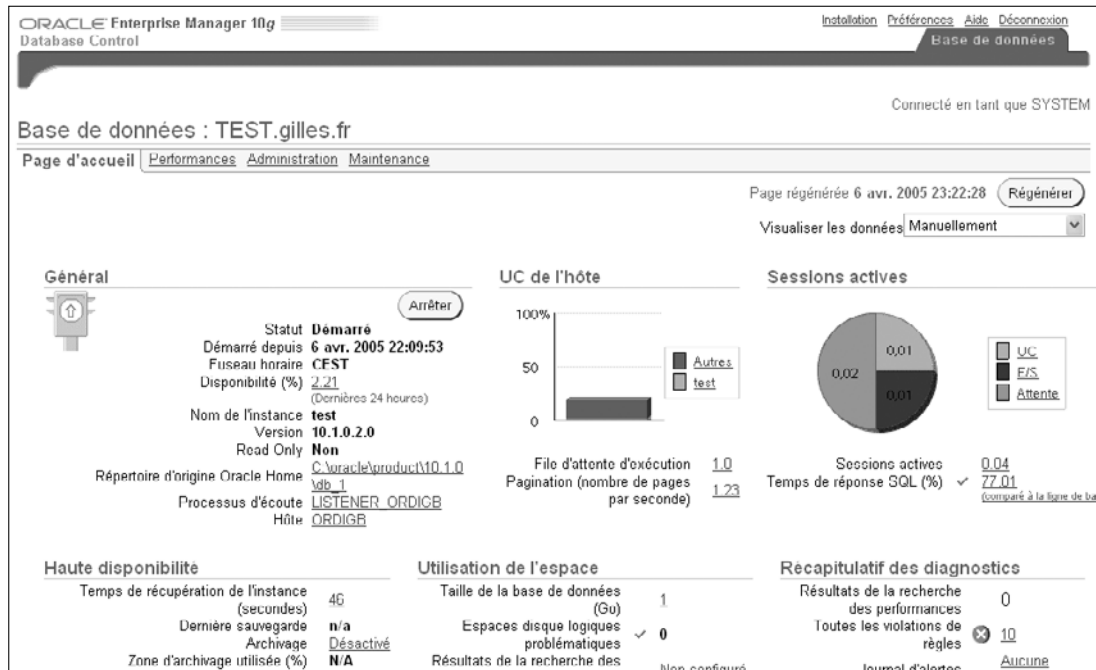


Figure 5-16

Arrêt de l'instance par OEM

Lorsque seul le service fonctionne, quels sont les threads restants ?

Après l'arrêt de l'instance Oracle, le service OracleServiceSID peut rester actif. Dans ce cas, seuls les threads nécessaires à son fonctionnement et à une demande de démarrage de l'instance restent actifs.

Arrêter le service Windows

De même que pour son démarrage, l'arrêt du service OracleServiceSID peut être effectué de plusieurs manières :

Arrêter le service OracleServiceSID par le Gestionnaire de services

On peut arrêter le service depuis le Gestionnaire de services ou manuellement, à l'aide de la commande net stop.

```
# Depuis une fenêtre Invite de Commandes
# arrêt du service de l'instance TEST
```

```
c:\> net stop OracleServiceTEST
```

Arrêter le service OracleServiceSID avec ORADIM

L'utilitaire Oracle ORADIM (présenté dans ce chapitre) gère la création et la suppression des services OracleServiceSID. Il permet également de démarrer ou d'arrêter le service et l'instance, en fournissant les paramètres requis.

```
# Depuis une fenêtre Invite de Commandes
# lancement de l'utilitaire ORADIM (arrêt du service uniquement)

c:\> ORADIM -shutdown -sid TEST -shuttype srvc
```

Que se passe-t-il si l'instance fonctionne lors de l'arrêt du service ?

Reportez-vous au paragraphe précédent pour paramétrer les options de démarrage et d'arrêt du service avec celui de l'instance Oracle.

Après l'arrêt du service, l'exécutable oracle.exe associé au service s'arrête.

Comment associer les threads aux process Oracle ?

En ce qui concerne le système d'exploitation, nous avons vu qu'il est simple de visualiser les threads du processus oracle.exe. De même, en ce qui concerne Oracle, on accède facilement à la liste des process Oracle. Mais comment relier un thread à un processus Oracle ? Comment connaître le nom de l'utilisateur connecté sous Windows ? Comment savoir quel est le programme qui a donné naissance au thread utilisateur ?

C'est ce que propose l'ordre SQL suivant, qui relie tous ces éléments. Nous vous le donnons comme exemple de la puissance des renseignements qu'il est possible de puiser dans le dictionnaire de données d'Oracle.

```
SQL> select p.spid "NoThread",
        b.name "Process",
        s.username "UserName",
        s.osuser "OS User",
        s.status "STATUS",
        s.sid "Session",
        s.serial# "Serial",
        s.program "OS Program"
        from v$process p, v$bgprocess b, v$session s
        where s.paddr = p.addr and b.paddr(+) = p.addr
```

No Thread	Proce ss	UserName	OS User	STATUS	Session	OS Program
532	PMON		SYSTEM	ACTIVE	170	ORACLE.EXE (PMON)
812	MMAN		SYSTEM	ACTIVE	169	ORACLE.EXE (MMAN)
1016	DBWO		SYSTEM	ACTIVE	168	ORACLE.EXE (DBWO)
1824	LGWR		SYSTEM	ACTIVE	167	ORACLE.EXE (LGWR)
1928	CKPT		SYSTEM	ACTIVE	166	ORACLE.EXE (CKPT)
2008	SMON		SYSTEM	ACTIVE	165	ORACLE.EXE (SMON)
248	RECO		SYSTEM	ACTIVE	164	ORACLE.EXE (RECO)

300	CJQO	SYSTEM	ACTIVE	163	ORACLE.EXE (CJQO)
2004	QMNC	SYSTEM	ACTIVE	159	ORACLE.EXE (QMNC)
152	MMON	SYSTEM	ACTIVE	158	ORACLE.EXE (MMON)
244	MMNL	SYSTEM	ACTIVE	156	ORACLE.EXE (MMNL)
2916	SYSTEM	ORDIGB\GB	ACTIVE	147	sqlplusw.exe
1356	SCOTT	ORDIGB\	ACTIVE	155	sqlplusw.exe
		Administra			
		teur			

Dans cet exemple, on visualise les processus Oracle et les numéros de threads Windows associés. On constate que les utilisateurs SCOTT et SYSTEM sont tous deux connectés depuis le compte administrateur du serveur ORASERV et qu'ils utilisent la version graphique de SQL*Plus pour accéder à la base. L'identification sans risque d'erreur du numéro de thread utilisateur permet ainsi de « tuer » une session utilisateur.

L'ordre SQL suivant permet de visualiser l'ensemble des threads qu'une base Oracle sous Windows peut créer. La liste est impressionnante ! Lors d'un fonctionnement « normal », seule une partie de ces threads sera démarrée.

```
SQL> select name, description from v$bgprocess
```

NAME	DESCRIPTION
-----	-----
PMON	process cleanup
DIAG	diagnosibility process
FMON	File Mapping Monitor Process
LMON	global enqueue service monitor
LMD0	global enqueue service daemon 0
LMS0	global cache service process 0
LMS1	global cache service process 1
LMS2	global cache service process 2
LMS3	global cache service process 3
LMS4	global cache service process 4
LMS5	global cache service process 5
LMS6	global cache service process 6
LMS7	global cache service process 7
LMS8	global cache service process 8
LMS9	global cache service process 9
LMSa	global cache service process 10
LMSb	global cache service process 11
LMSc	global cache service process 12
LMSd	global cache service process 13
LMSe	global cache service process 14
LMSf	global cache service process 15
LMSg	global cache service process 16
LMS h	global cache service process 17
LMSi	global cache service process 18
LMSj	global cache service process 19
MMAN	Memory Manager
DBW0	db writer process 0
DBW1	db writer process 1

DBW2 db writer process 2
DBW3 db writer process 3
DBW4 db writer process 4
DBW5 db writer process 5
DBW6 db writer process 6
DBW7 db writer process 7
DBW8 db writer process 8
DBW9 db writer process 9
DBWa db writer process 10 (a)
DBWb db writer process 11 (b)
DBWc db writer process 12 (c)
DBWd db writer process 13 (d)
DBWe db writer process 14 (e)
DBWf db writer process 15 (f)
DBWg db writer process 16 (g)
DBWh db writer process 17 (h)
DBWi db writer process 18 (i)
DBWj db writer process 19 (j)
ARC0 Archival Process 0
ARC1 Archival Process 1
ARC2 Archival Process 2
ARC3 Archival Process 3
ARC4 Archival Process 4
ARC5 Archival Process 5
ARC6 Archival Process 6
ARC7 Archival Process 7
ARC8 Archival Process 8
ARC9 Archival Process 9
LNS0 Network Server 0
LNS1 Network Server 1
LNS2 Network Server 2
LNS3 Network Server 3
LNS4 Network Server 4
LNS5 Network Server 5
LNS6 Network Server 6
LNS7 Network Server 7
LNS8 Network Server 8
LNS9 Network Server 9
MRP0 Managed Standby Recovery
LGWR Redo etc.
LCK0 Lock Process 0
CKPT checkpoint
LSP0 Logical Standby
LSP1 Dictionary build process for Logical Standby
LSP2 Set Guard Standby Information for Logical Standby
CTWR Change Tracking Writer
RVWR Recovery Writer
SMON System Monitor Process
RECO distributed recovery
CJQ0 Job Queue Coordinator
EMN0 Event Monitor Process 0
QMNC AQ Coordinator

```

DMON  DG Broker Monitor Process
RSM0  Data Guard Broker Resource Guard Process 0
RSM1  Data Guard Broker Resource Guard Process 1
NSV0  Data Guard Broker NetSlave Process 0
NSV1  Data Guard Broker NetSlave Process 1
NSV2  Data Guard Broker NetSlave Process 2
NSV3  Data Guard Broker NetSlave Process 3
NSV4  Data Guard Broker NetSlave Process 4
NSV5  Data Guard Broker NetSlave Process 5
NSV6  Data Guard Broker NetSlave Process 6
NSV7  Data Guard Broker NetSlave Process 7
NSV8  Data Guard Broker NetSlave Process 8
NSV9  Data Guard Broker NetSlave Process 9
INSV  Data Guard Broker INstance SlaVe Process
RBAL  ASM Rebalance master
ARB0  ASM Rebalance 0
ARB1  ASM Rebalance 1
ARB2  ASM Rebalance 2
ARB3  ASM Rebalance 3
ARB4  ASM Rebalance 4
ARB5  ASM Rebalance 5
ARB6  ASM Rebalance 6
ARB7  ASM Rebalance 7
ARB8  ASM Rebalance 8
ARB9  ASM Rebalance 9
ARBA  ASM Rebalance 10
ASMB  ASM Background
MMON  Manageability Monitor Process
MMNL  Manageability Monitor Process 2

```

Rassurez-vous, vous n'avez pas à connaître l'ensemble de ces threads. Seuls les principaux sont à identifier. Ils sont présentés dans les chapitres suivants.

L'utilitaire ORADIM

Le paragraphe précédent a montré le rôle important que joue l'utilitaire ORADIM. N'existant que dans l'environnement Windows, il permet :

- la création et la suppression des services OracleServiceSID dans la base de registre ;
- le démarrage et l'arrêt des services OracleServiceSID ;
- le démarrage et l'arrêt des instances Oracle SID.

L'utilitaire Oracle Database Assistant permet de piloter graphiquement toutes les options d'ORADIM. Cet utilitaire est présenté au chapitre 14, *Création d'une base Oracle 10g*.

L'aide en ligne d'ORADIM est accessible par la commande ORADIM -help. Dans la syntaxe qui suit, les paramètres entre crochets [] sont optionnels, les paramètres séparés par « | » indiquent une condition « ou ». Par exemple « A|B » signifie « A ou B ».

ORADIM permet aussi de créer le service Windows associé à une instance Automatic Storage Management (ASM). Les options nécessaires à ASM ne sont pas abordées dans ce chapitre.

Création d'une instance

ORADIM -NEW -SID *votre_sid* | -SRVC *nom_service* [-SYSPWD *mot_de_passe_SYS*] [-STARTMODE AUTO | MANUAL] [-SRVCSTART system | demand] [-PFILE *fichier_initialisation* | -SPFILE] [-SHUTMODE normal | immediate | abort] [-TIMEOUT *secondes*]

- NEW indique que l'on va créer un nouveau service ;
- *votre_sid* est l'identifiant de l'instance ou *SID* (utilisez quatre caractères majuscules) ;
- *nom_service* est le nom du service Windows à créer. Respectez une syntaxe du type OracleService*SID* (comme pour OracleServiceTEST) ;
- *mot_de_passe_SYS* est le mot de passe utilisé lorsqu'on se connecte sous le compte utilisateur possédant le privilège SYSDBA. Il est facultatif. Si vous ne l'utilisez pas, le groupe Windows ORA_DBA permet aux utilisateurs qui le possèdent de se connecter à Oracle sans fournir de mot de passe. Par défaut, seul les administrateurs Windows sont membres de ce groupe ;
- AUTO ou MANUAL indique le mode de démarrage du service lorsque Windows se lance ;
 - PFILE *fichier_initialisation* précise où se trouve le fichier d'initialisation INIT-SID.ORA (fournissez le nom du fichier et son chemin d'accès). Vous pouvez le remplacer par le SPFILE, qu'Oracle va chercher par défaut dans le répertoire ORACLE_HOME/database ;
 - SHUTMODE normal | immediate | abort indique le mode d'arrêt de l'instance lors de l'arrêt de Windows ;
 - TIMEOUT *secondes* détermine la durée d'attente d'un service avant son arrêt. L'arrêt de Windows est alors différé le temps que la base s'arrête « proprement ».

Cette commande crée uniquement les services, les variables dans la base de registre et le fichier mot de passe associé à la nouvelle base. La base de données n'est pas créée à cette étape.

Par exemple, pour créer le service associé à une base nommée TEST :

```
ORADIM -NEW -SID TEST -STARTMODE auto -SPFILE
```

Démarrage d'une instance

ORADIM -STARTUP -SID *votre_sid* [-SYSPWD *mot_de_passeSYS*] [-STARTTYPE SRVC, INST] [-PFILE *fichier_initialisation* | -SPFILE]

- -STARTUP indique le démarrage ;

- *votre_sid* est l'identifiant de votre instance ou SID à démarrer ;
- *mot_de_passeSYS* est le mot de passe SYS s'il a été saisi lors de la création de l'instance ;
- SRVC démarre le service associé à l'instance (oracle.exe) sans démarrer l'instance ;
- INST démarre l'instance Oracle ;
- SRVC, INST : les deux paramètres peuvent être donnés ;
- -PFILE *fichier_initialisation* précise où se trouve le fichier d'initialisation INITSID.ORA (fournissez le nom du fichier et son chemin d'accès). Vous pouvez le remplacer par le SPFILE, qu'Oracle va chercher par défaut dans le répertoire ORACLE_HOME\database.

Les paramètres -STARTUP, -SID, -STARTTYPE sont obligatoires, les autres sont optionnels.

Par exemple, pour démarrer le service et la base associés à une base nommée TEST :

```
ORADIM -STARTUP -SID TEST -STARTTYPE SRV,INST auto -SPFILE
```

Arrêt d'une instance

```
ORADIM -SHUTDOWN -SID votre_sid ] [-SYSPWD mot_passeSYS ] [-SHUTTYPE  
SRVC | INST | SRVC,INST] -SHUTMODE [nomal, immediate, abort]
```

- -SHUTDOWN indique l'arrêt ;
- *votre_sid* est l'identifiant de votre instance ou SID ;
- *mot_passeSYS* est le mot de passe SYS s'il a été saisi lors de la création de l'instance ;
- SRVC arrête le service associé à l'instance (oracle.exe), INST arrête uniquement l'instance ;
- SRVC, INST : les deux paramètres peuvent être saisis ;
- SHUTMODE indique le type d'arrêt de l'instance : Abort ou « shutdown abort », le plus violent, Immediate (toutes les connexions sont fermées puis la base arrêtée), Normal (on attend que toutes les connexions soient fermées avant d'arrêter la base). Méfiez-vous de « N » si des process en tâche de fond sont connectés à la base (c'est le cas de l'agent intelligent Oracle et des serveurs Apache), ils empêchent alors l'arrêt de votre instance. Si vous omettez ce paramètre, l'option Normal est utilisée.

L'arrêt d'une base Oracle est traité au chapitre 10, *Démarrer et arrêter une base Oracle 10g*.

Les paramètres -SUTDOWN, -SID, -SHUTTYPE sont obligatoires, les autres sont optionnels.

Par exemple, pour arrêter l'instance et le service associés à une base nommée TEST :

```
ORADIM -SHUTDOWN -SID TEST -SHUTTYPE SRV,INST -SHUTMODE IMMEDIATE
```

Modification des caractéristiques d'une instance

ORADIM -EDIT -SID *vousre_sid* [-SYSPWD *mot_passeSYS*] [-STARTMODE AUTO,MANUAL] [-SRVSTART SYSTEM | DEMAND]] [-PFILE *fichier_initialisation* | -SPFILE] [-SHUTMODE nomal, immediate, abort] [-SHUTTYPE SRVC | INST | SRVC,INST]

- -EDIT : modification de presque tous les paramètres de l'instance ;
- *vousre_sid* est l'identifiant de l'instance ou SID (utilisez quatre caractères majuscules) ;
- *mot_passeSYS* modifie le mot de passe SYS qui a été saisi lors de la création de l'instance ;
- AUTO ou MANUAL indique si l'instance doit être démarrée lorsque le service est lancé ;
- SYSTEM ou DEMAND indique si le service créé doit démarrer automatiquement en même temps que le serveur Windows ;
- -PFILE *fichier_initialisation* précise où se trouve le fichier d'initialisation INITSID.ORA (fournissez le nom du fichier et son chemin d'accès). Vous pouvez le remplacer par le SPFILE qu'Oracle va chercher par défaut dans le répertoire ORACLE_HOME\database ;
- SHUTMODE indique le type d'arrêt de l'instance : Abort ou « shutdown abort », le plus violent, Immediate (toutes les connexions sont fermées puis la base arrêtée), Normal (on attend que toutes les connexions soient fermées avant d'arrêter la base). Méfiez-vous de « N » si des process en tâche de fond sont connectés à la base (c'est le cas de l'agent intelligent Oracle et des serveurs Apache), ils empêchent alors l'arrêt de votre instance. Si vous omettez ce paramètre, l'option Normal est utilisée ;
- SRVC arrête le service associé à l'instance (oracle.exe), INST arrête uniquement l'instance ;
- SRVC, INST : les deux paramètres peuvent être saisis.

Cette commande ne modifie pas le fichier mot de passe associé à votre base. Pour créer un nouveau fichier mot de passe, utilisez l'utilitaire ORAPWD (décrit au chapitre 24, *Stratégie de sécurité sous Windows*), ou alors supprimez le service OracleServiceSID (qui détruit le fichier mot de passe), puis recréez ce même service (le fichier mot de passe est alors créé).

Par exemple, pour indiquer le changement d'emplacement du fichier d'initialisation d'une instance nommée TEST :

```
ORADIM -EDIT -SID TEST -PFILE c:\oracle\product\TEST\pfile\init.ora
```

Suppression d'une instance

ORADIM -DELETE -SID *vousre_sid* -SRVC *vousre_service*

ORADIM -DELETE, *vousre_serviceB*, *vousre_serviceC*...

- -DELETE : suppression d'une instance. Méfiez-vous !
- *votre_sid* est l'identifiant (ou SID) de l'instance à supprimer. Méfiez-vous !
- *votre_service* est le nom du service à supprimer. Méfiez-vous !

Une fois votre instance supprimée, vos fichiers qui constituent votre base de données existent toujours, c'est à vous de les détruire manuellement.

Par exemple, pour supprimer l'instance nommée TEST :

```
ORADIM -DELETE -SID TEST
```

Comment déplacer une base Oracle existante vers une autre machine ?

L'usage d'ORADIM est indispensable pour effectuer des opérations d'administration comme le déplacement d'une base Oracle 10g d'un serveur Windows vers un autre serveur Windows. La démarche est alors la suivante :

1. Installez la même version des logiciels Oracle sur le nouveau serveur.
2. Créez avec ORADIM les services sur le nouveau serveur, avec le même *SID*.
3. Arrêtez la base sur l'ancien serveur.
4. Copiez les fichiers de l'ancien serveur vers le nouveau dans les mêmes répertoires. Si les répertoires sont différents, regardez les commandes `alter tablespace rename datafile xxxx,yyyy to vvvv,www`.
5. Copiez les fichiers d'initialisation *INITSID.ORA*.
6. Copiez les fichiers d'administration situés en *C:\oracle\admin\SID*.
7. Modifiez vos fichiers *listener.ora* et *tnsnames.ora* pour tenir compte de cette nouvelle base sur votre serveur.
8. Démarrez la base sur le nouveau serveur.

Oracle Administration Assistant for Windows

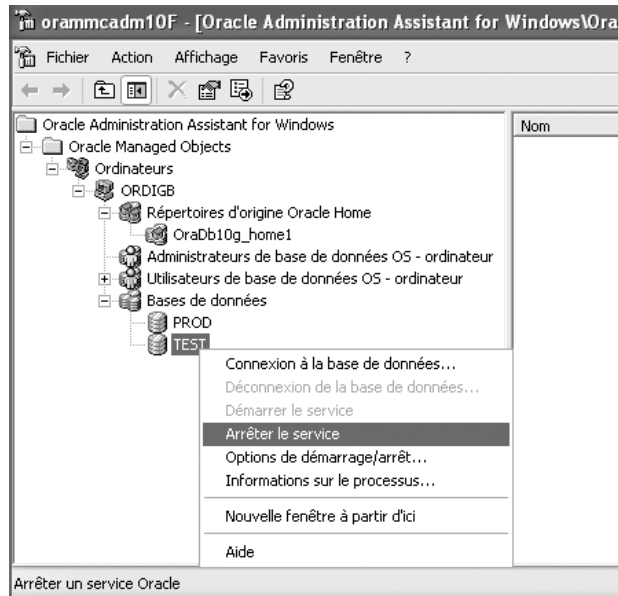
Cet assistant graphique n'existe que sur les systèmes Microsoft. Il est utile principalement pour :

- configurer les comptes utilisateur Windows pour éviter d'avoir à saisir un mot de passe lors du démarrage d'Oracle ;
- configurer l'instance Oracle de manière à ce qu'elle démarre et qu'elle s'arrête en même temps que le service *OracleServiceSID* ;
- paramétrer le démarrage et l'arrêt du service *OracleServiceSID* ;
- définir quelle sera la base de données par défaut ;
- modifier les paramètres de l'Oracle Home situé dans la base de registre.

Nous présentons rapidement chacune de ces possibilités.

Oracle Administration Assistant for Windows est accessible par le menu Démarrer > Programmes > Oracle-OraDb10g_home1 > Configuration and Migration Tools. Il s'exécute dans la fenêtre d'une console *Microsoft Management Console* (MMC). Pour accéder aux différents menus, il faut sélectionner un nœud et cliquer avec le bouton droit de la souris dessus.

Figure 5-17
*La console MMC
de l'assistant*



Configurer les comptes Windows

Vous devez saisir un mot de passe pour vous connecter sous Windows, puis en saisir un autre pour accéder à Oracle. Pourquoi ne pas synchroniser ces deux accès ? Dans ce cas, Oracle considère que l'utilisateur Windows s'est déjà identifié et il lui permet de se connecter à Oracle pour effectuer certaines opérations.

Résumé comme cela, la chose est simple. Elle est plus complexe à mettre en place car les enjeux de sécurité sont importants et Oracle y est très sensible. Les droits nécessaires pour démarrer et arrêter une base (rôle dévolu à un administrateur système) ne lui donnent pas forcément le droit d'accéder à tout le contenu de cette base (rôle dévolu à un administrateur Oracle).

Cet assistant est l'outil privilégié à utiliser pour configurer les comptes Windows et Oracle.

Ces points sont détaillés au chapitre 24, *Stratégie de sécurité sous Windows*.

Configurer le démarrage de l'instance Oracle

Nous avons décrits dans ce chapitre le rôle respectif du service OracleServiceSID et celui de la base de données. Pour une plus grande facilité d'administration, nous vous conseillons :

- de démarrer le service OracleServiceSID en même temps que l'ordinateur ;
- de configurer l'instance Oracle de manière à ce qu'elle démarre et qu'elle s'arrête en même temps que le service OracleServiceSID.

Pour cela, il faut sélectionner chaque instance et sélectionner Options de démarrage/arrêt (voir la figure précédente). L'écran suivant permet alors de gérer toutes ces options.

Figure 5-18

*Options de démarrage
et d'arrêt du service et
de l'instance*



Les clés de registre impactées par ces modifications sont situées pour le registre en HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\SERVICES et pour l'instance en HKEY_LOCAL_MACHINE\SOFTWARE\ORACLE\KEY_OraDb10g_home1.

Définir la base par défaut et paramétrer les variables de l'Oracle Home

Si plusieurs bases de données fonctionnent sur le serveur, vous pouvez indiquer quelle sera la base de données par défaut. Pour vous connecter, nous vous conseillons toutefois de toujours donner le nom d'utilisateur, son mot de passe et le nom d'hôte (qui identifie le serveur et la base de données cible), car se fier à une valeur par défaut peut être source d'erreur.

Pour modifier le paramétrage des variables liées à l'Oracle Home, il suffit de sélectionner la variable désirée pour la modifier.

Vous pouvez constater que l'on retrouve les variables ORA_PROD_XXX qui correspondent aux options de démarrage de l'instance PROD sélectionnée.

Les clés de registre affectées par ces modifications sont situées en HKEY_LOCAL_MACHINE\SOFTWARE\ORACLE\KEY_OraDb10g_home1.

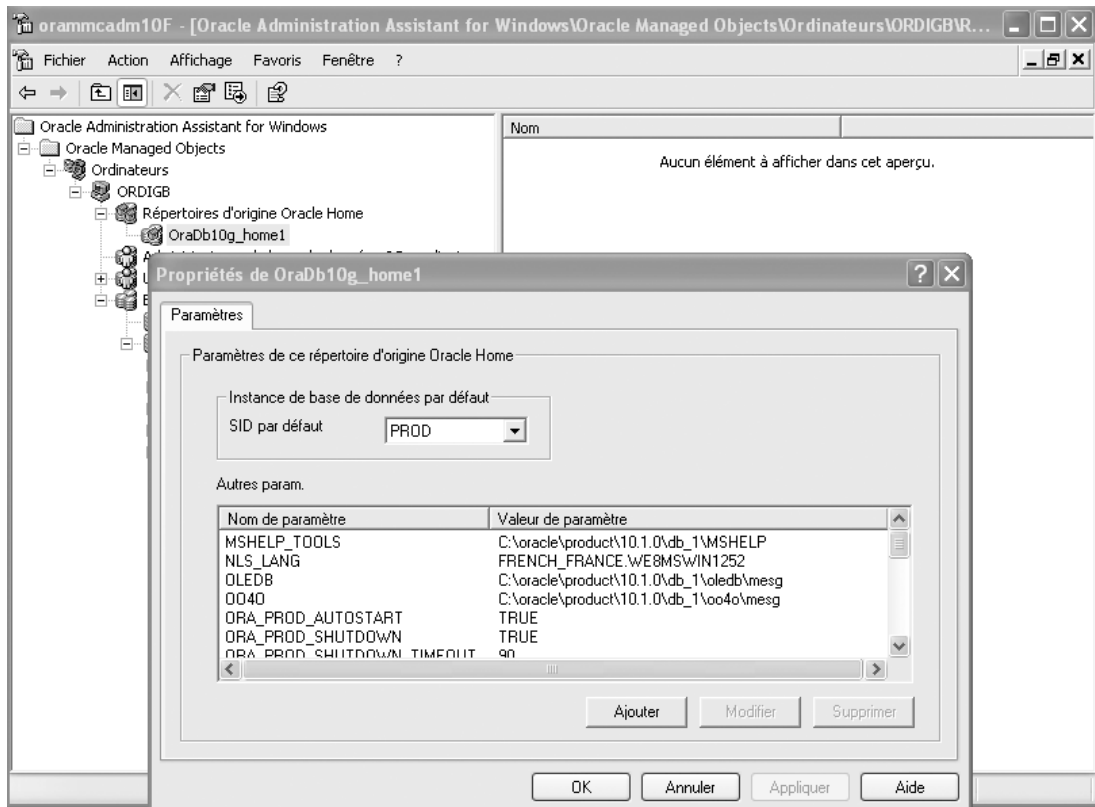


Figure 5-19

Modification des variables de l'Oracle Home

Résumé

Ce chapitre constitue une introduction indispensable pour comprendre le fonctionnement d'Oracle sous Windows. Après avoir rapidement présenté le fonctionnement d'Oracle, nous avons abordé les notions de base de données et d'instance.

Les spécificités du fonctionnement d'Oracle sur Windows ont été traitées. Pour plus de clarté dans l'exposé, une vérification pratique des éléments théoriques a été effectuée. Ainsi, l'importance respective du service Windows et de l'instance Oracle a été vue.

Alors que ce chapitre est très lié aux spécificités d'Oracle sous Windows, les chapitres suivants sont consacrés au fonctionnement interne d'Oracle : les processus Oracle, les fichiers, la gestion mémoire, le dictionnaire de données, le démarrage et l'arrêt d'une base, les transactions.

6

Les clusters Oracle

Dans ce chapitre :

- historique et enjeux des clusters ;
- principes d'une architecture cluster ;
- Oracle Fail Safe ;
- Oracle Real Application Clusters ;
- comparaison et conseils sur la mise en place d'un cluster.

L'objectif de ce chapitre est, par-delà l'annonce marketing du Grid Computing et les idées simplistes accolées aux cluster, de vous en présenter les enjeux, objectifs et principes de fonctionnement. Nous abordons Oracle Fail Safe, Oracle Real Application Clusters (RAC) et Microsoft Cluster Server (MSCS) en suivant comme ligne directrice la mise en œuvre de bases de données Oracle.

Historique et enjeux des technologies cluster

Après des années de fonctionnement sous différents systèmes d'exploitation, VMS (Digital), différents systèmes Unix du marché (Sequent, IBM, Sun, HP, etc.), des versions de Windows arrivent sur ce marché et offrent des solutions de type cluster. C'est un domaine nouveau à la fois pour Microsoft et pour les constructeurs de matériel micro-informatique qui devront intégrer cette technologie, ou plutôt ces technologies. Car sous le nom de Microsoft Cluster Server (MSCS), Microsoft introduit étape par étape des technologies allant des plus simples aux plus rigides.

Les enjeux sont grands pour Microsoft et les constructeurs de micro-ordinateurs car ils s'engagent sur un nouveau marché où les constructeurs « classiques » tels que IBM, Sun, Hewlett Packard, etc., vont se défendre bec et ongle face à la galaxie de constructeurs liés à Intel : HP-Compaq, Dell, etc.

Cette nouvelle jeunesse pour la technologie cluster constitue une opportunité pour des constructeurs en difficultés comme Data General, Unisys, etc. qui, forts de leur expérience technique et de leur connaissance des grands comptes, peuvent revenir en tête de la course... ou se faire racheter leur technologie.

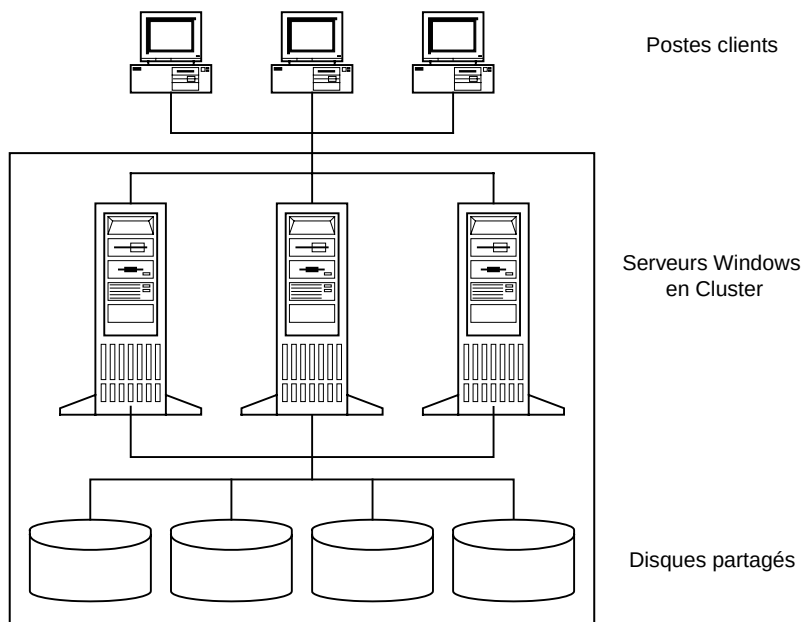
Ne nous y trompons pas : le marché des systèmes Windows évolue. Les mainframes et les minis survivront et devront coexister avec les machines Windows les plus puissantes. Ce sera le marché le plus rentable pour les constructeurs : celui des gros serveurs, soumis à des contraintes technologiques fortes, nécessitant des équipes de consultants performants. À l'autre bout de l'échelle, le micro-ordinateur démocratise tout. Maintenant, vous pouvez acheter un serveur Windows « bas de gamme » au « supermarché du coin ». Retrouvera-t-on les mêmes constructeurs aux deux bouts de l'échelle ? Rien n'est moins sûr.

Principes d'une architecture cluster Windows

Un cluster est une architecture matérielle où plusieurs serveurs indépendants sont reliés afin de fonctionner comme s'ils constituaient un seul serveur. Chaque serveur participant à un cluster est appelé nœud. Dans des conditions normales, chaque nœud agit de manière indépendante et gère son propre travail.

Figure 6-1

Principe d'un cluster



Les caractéristiques dominantes des clusters sont :

- la performance ;
- la fiabilité et la disponibilité ;
- l'évolutivité facile par ajout de nœuds ;
- l'administration de plusieurs machines vues comme une seule.

En effet, les utilisateurs ne voient le cluster que comme une seule et même machine, distribuant des ressources communes. Un cluster est théoriquement (nous verrons pourquoi) indépendant de l'application et simple à administrer car constitué de machines connues et maîtrisées.

Les serveurs sont monoprocesseurs ou multiprocesseurs. Ils partagent un ensemble de disques communs (souvent de type RAID, comme présenté au chapitre 27, *Configurer les disques pour Oracle*) et les machines sont reliées par des canaux spéciaux à haut débit de type SCSI, Ethernet ou autres technologies. L'augmentation de la puissance de l'ensemble est facile à obtenir : il suffit d'ajouter un nœud ou des processeurs.

L'architecture cluster ne fait pas que mettre en commun des ressources. Elle doit les gérer, en vérifier le bon fonctionnement, effectuer une répartition de la charge entre les nœuds les plus chargés et les moins chargés et, surtout, pallier les pannes en basculant des applications d'une machine vers une autre. C'est donc un ensemble technique particulièrement complexe.

Beaucoup de ces arguments n'ont de valeur que marketing. Il faut avant tout déterminer ce dont vous avez le plus besoin : haute disponibilité ? performance ? évolutivité ? Et n'oubliez pas que tout cela a un coût et va requérir des compétences souvent pointues, tant au niveau du système que de la base de données.

Les deux types de clusters proposés par Oracle

Oracle propose deux types de clusters : Oracle Fail Safe et Oracle Real Application Clusters (RAC). Chacun d'eux répond à des objectifs précis. Ils imposent aussi des contraintes bien différentes.

Objectifs de Fail Safe et Real Application Server

Pour positionner les objectifs respectifs d'Oracle Fail Safe et Oracle Real Application Server, le schéma précédent les compare suivant deux axes : celui de la haute disponibilité et celui de la performance et de l'évolutivité.

Le domaine d'Oracle Real Application Server combine à la fois performance et haute disponibilité, alors qu'Oracle Fail Safe se concentre surtout sur la disponibilité.

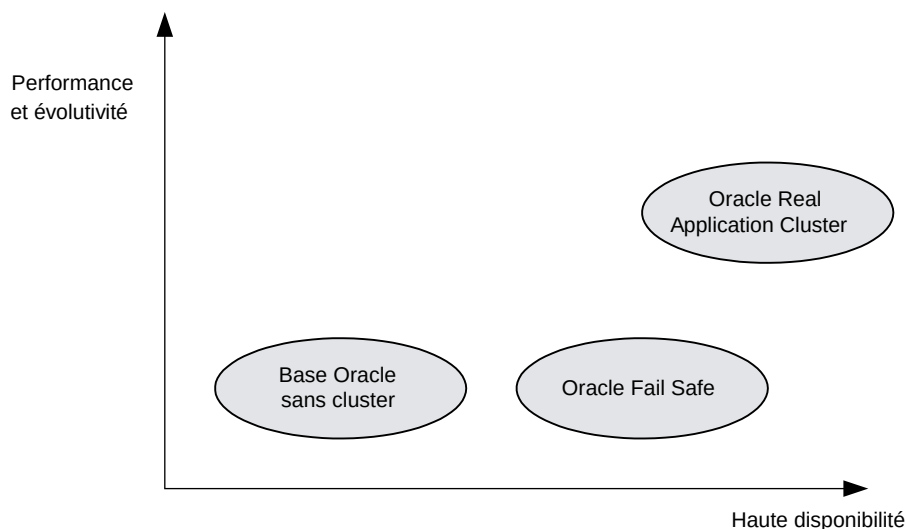


Figure 6-2

Les clusters Oracle

Oracle Fail Safe

Oracle Fail Safe est lié aux possibilités offertes par les récentes fonctionnalités cluster de Microsoft. Il vous permet de disposer d'une solution sécurisée, d'une mise en place relativement facile et d'un coût maîtrisé, qui met en œuvre des configurations matérielles cluster certifiées par Microsoft.

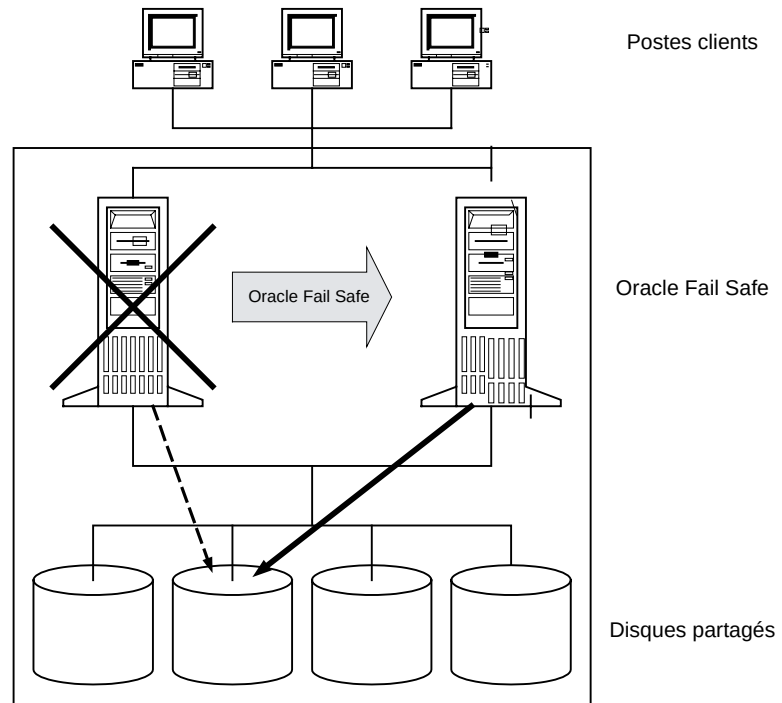
Dans une architecture Oracle Fail Safe, la base de données ne fonctionne que sur un seul serveur à un instant donné. Si une défaillance se produit, la base de données est automatiquement relancée sur l'autre serveur, qui était inactif jusque-là.

Le deuxième serveur n'est pas en « attente » de cet unique événement, il peut lui aussi héberger des bases de données. On peut disposer d'une configuration Oracle Fail Safe où chaque machine héberge ses propres bases, qui peuvent être basculées d'un serveur sur l'autre en cas de problème.

Une configuration Fail Safe est limitée à deux nœuds : l'un actif, l'autre en sauvegarde. Le fait de disposer de deux serveurs n'accroît pas la puissance disponible pour une base, car un seul est actif à un instant donné.

Vos applications sont utilisables telles quelles avec Fail Safe, aucune mise au point n'est nécessaire.

L'outil d'administration graphique Oracle Enterprise Manager permet d'administrer vos clusters Fail Safe.

Figure 6-3*Principe d'Oracle Fail Safe*

Oracle Real Application Clusters

L'architecture d'Oracle Real Application Clusters est plus évoluée que celle d'Oracle Fail Safe. C'est elle qui propose à la fois performance, évolutivité et disponibilité. Dans le « langage Oracle » courant, le terme cluster est implicitement associé à Oracle Real Application Clusters.

Dans une architecture Oracle Real Application Clusters, la base de données fonctionne simultanément sur l'ensemble des serveurs. On peut ajouter ou arrêter dynamiquement un nœud dans la configuration. Si une défaillance se produit, les autres nœuds assurent une continuité de service.

La configuration permet donc d'avoir une grande puissance, car tous les nœuds travaillent simultanément, alors que dans Fail Safe, un seul est actif.

Une configuration Real Application Clusters n'est pas limitée à deux nœuds, comme avec Oracle Fail Safe. On peut rencontrer des clusters à deux, trois, quatre nœuds voire plus.

Vos applications doivent être modifiées pour tirer parti d'Oracle Real Application Clusters, notamment pour se reconnecter automatiquement en cas de panne.

L'outil d'administration graphique Oracle Enterprise Manager permet d'administrer vos clusters Real Application Clusters.

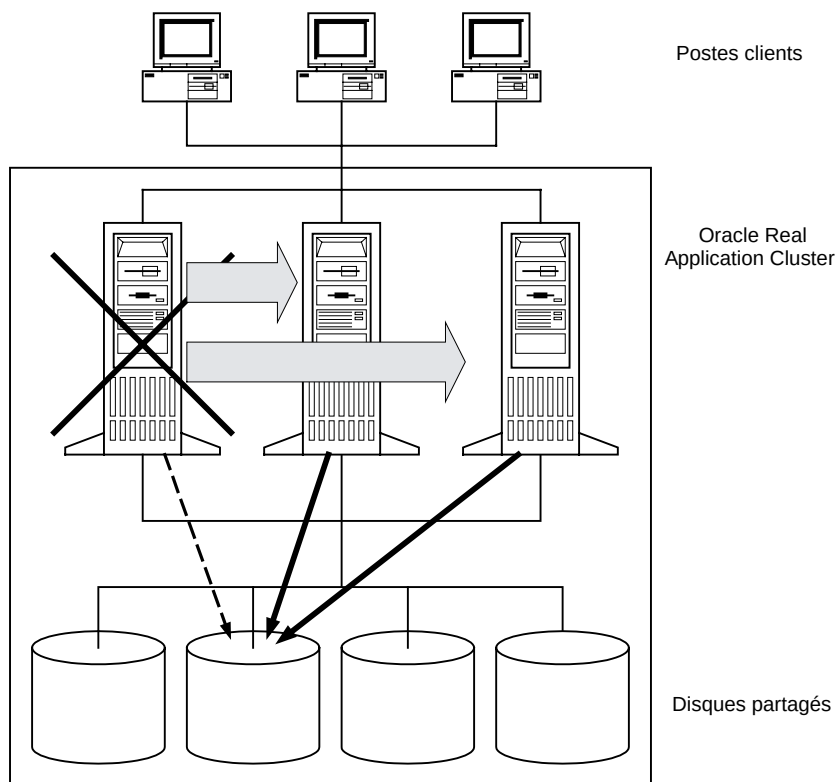


Figure 6-4

Principe d'Oracle Real Application Clusters

Oracle et les serveurs multiprocesseurs

Oracle tire parti des serveurs multiprocesseurs pour améliorer les temps de réponse des ordres SQL. Tout d'abord, un serveur multiprocesseur accepte une charge processeur plus importante, mais l'idée maîtresse est la suivante : plutôt que de laisser travailler un seul processeur, certaines opérations sont découpées et confiées à des processeurs tiers. Elles s'exécutent donc en parallèle.

Quelles sont ces opérations susceptibles d'être parallélisées ? Ce sont forcément des opérations longues qui sont concernées. On trouve ainsi les lectures complètes de tables (*Full Table Scan*), les opérations de tri des données, la création d'index, le chargement de tables avec *SQL*Loader*, etc.

Incorporée dans toutes les versions d'Oracle, et, à ce titre, gratuite, cette caractéristique est particulièrement intéressante car sa capacité de répartition du travail sur plusieurs processeurs peut s'étendre à plusieurs nœuds. On peut ainsi avoir une charge répartie sur plusieurs nœuds et une requête traitée simultanément par plusieurs processeurs.

Quels liens avec Microsoft Cluster Server (MSCS) ?

Microsoft Cluster Server (MSCS) est composé d'un ensemble de logiciels et de recommandations qui permettent de construire une architecture cluster. MSCS est livré à partir de Windows 2000 dans des versions « Advanced Server » ou « Datacenter Server » et ne peut être utilisé que sur des matériels ayant reçu le label de Microsoft. Ceci va constituer une petite révolution pour les acteurs du monde micro : toute la configuration doit être validée, depuis le hardware machine, les cartes installées, les logiciels, jusqu'aux types de câbles reliant les machines devront être certifiés. Les « bidouilleurs » qui sévissent parfois dans le monde micro-informatique n'auront pas leur place dans cet environnement !

Oracle Fail Safe repose sur Microsoft Cluster Server (MSCS). Oracle Real Application Clusters s'affranchit d'un système d'exploitation particulier car les couches logicielles nécessaires au fonctionnement de RAC sont fournies par Oracle. C'est pourquoi Fail Safe ne fonctionne que dans l'environnement Windows alors qu'Oracle RAC fonctionne sur d'autres systèmes d'exploitation. Pour Fail Safe, les matériels doivent être validés par Microsoft dans sa Hardware Compatibility List (HCL). Pour Oracle RAC, les matériels doivent être validés par Oracle. Le choix devient alors de plus en plus réduit !

Avant de choisir l'un ou l'autre des clusters Oracle, contactez des entreprises déjà équipées du même système. En phase d'achat, les logiciels fonctionnent toujours très bien, alors que la réalité peut, parfois, être différente...

L'architecture cluster a fait ses preuves sur plusieurs systèmes d'exploitation mais elle constitue une nouveauté pour Microsoft. La maturité de MSCS sur Windows interviendra certainement avec l'avènement de Windows .NET. Ce sera un chemin long et pavé d'écueils. Microsoft devra donc se méfier car la moindre erreur sera immédiatement exploitée par ses détracteurs, d'autant plus qu'une partie du marché des systèmes « d'entreprise » est en jeu.

Prérequis techniques pour Oracle Fail Safe et Oracle Real Application Clusters

La politique commerciale des éditeurs et d'Oracle varie dans le temps. On constate cependant qu'au fil des versions, de plus en plus de fonctionnalités sont intégrées dans le produit standard.

À ce titre, Oracle Fail Safe est gratuit, mais il nécessite une version spéciale de Windows : Windows 2000 Advanced Server, Windows 2000 Datacenter Server ou une version .NET correspondante.

Oracle Real Application Clusters (RAC) est considéré comme une option technique d'Oracle nécessitant une architecture matérielle très particulière pour fonctionner ; c'est pourquoi il est payant.

Oracle Fail Safe	Oracle Real Application Clusters
Ne fonctionne que dans un environnement Microsoft.	Fonctionne dans un environnement Microsoft et Linux/Unix.
Nécessite un matériel validé par Microsoft.	Nécessite un matériel validé par Oracle.
Nécessite une version Windows comprenant MSCS : Windows 2000 Advanced Server, Datacenter Server ou les versions .NET correspondantes.	Utilise uniquement des logiciels Oracle, sur une version Windows 2000 ou .NET Server « standard ».
Livré en standard avec Oracle Server Standard Edition (peut varier dans le temps).	Utilisable en standard avec Oracle Database 10g Standard Edition mais limité à deux nœuds. Nécessite Oracle Database 10g Enterprise Edition pour un cluster supérieur à 2 nœuds (peut varier dans le temps).

Comparaison d'Oracle Fail Safe et d'Oracle Real Application Server

Les arguments suivants vous offrent des éléments de comparaison entre Oracle Fail Safe et Oracle Real Application Clusters (RAC) :

Oracle Fail Safe	Oracle Real Application Clusters
Améliore la disponibilité.	Améliore la disponibilité et les performances.
Un seul serveur actif à un instant donné.	Plusieurs serveurs actifs à un instant donné.
Toutes les opérations d'administration sont identiques à celles effectuées sur une base mono-instance.	Le Tuning ainsi que les autres opérations d'administration nécessitent des connaissances approfondies du fonctionnement des bases Oracle multi-instances et de leur configuration.
La connaissance des clusters Windows est requise.	La connaissance des clusters Oracle est requise.
Limité à deux nœuds.	Prend en charge des clusters ayant plus de deux nœuds.
Une application cliente n'a pas besoin d'être modifiée.	Les applications doivent être modifiées pour tirer pleinement parti des possibilités techniques du cluster : reprise automatique sur panne, bascule d'une session utilisateur d'un serveur à un autre sans interruption, etc.

Fonctionnement d'Oracle Fail Safe

Oracle Fail Safe travaille en collaboration avec Microsoft Cluster Server (MSCS). Il vous permet de construire facilement une solution sécurisée et d'un faible coût qui met en œuvre une configuration matérielle certifiée par Microsoft.

Cas d'utilisation d'Oracle Fail Safe

Ses utilisations et contraintes sont souvent celles rencontrées lors de l'utilisation de progiciels comme SAP R/3, Oracle Application, etc. Oracle Fail Safe peut être utilisé dans les cas suivants :

- arrêt programmé d'un serveur, pour une opération de maintenance logicielle ou matérielle ;

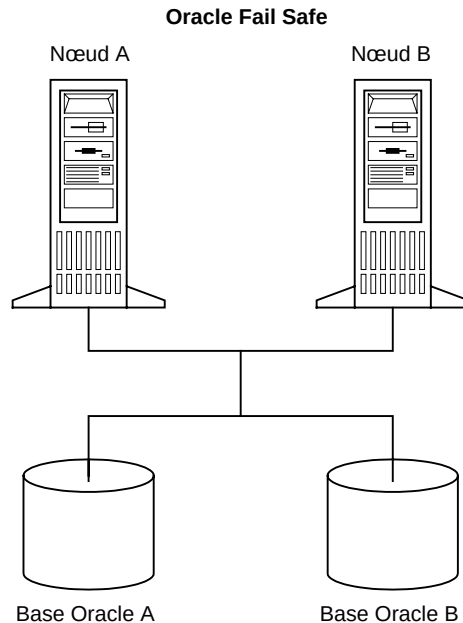
- arrêt non programmé : la base de données est relancée sur le second serveur ;
- répartition de charge manuelle, pour faire passer une base d'un serveur Windows chargé vers un serveur plus disponible ;
- comme un seul nœud est actif à un instant t , Oracle Fail Safe agit uniquement sur la sécurité et la disponibilité.

Principe de fonctionnement

Dans des circonstances normales, chaque nœud agit de manière indépendante et effectue son propre travail. Les nœuds « dialoguent » à la fois par du matériel et du logiciel. Le matériel relie les deux nœuds par un canal de communication haut débit (souvent SCSIII) et le logiciel est basé sur MSCS pour vérifier l'état de chacun des nœuds.

Figure 6-5

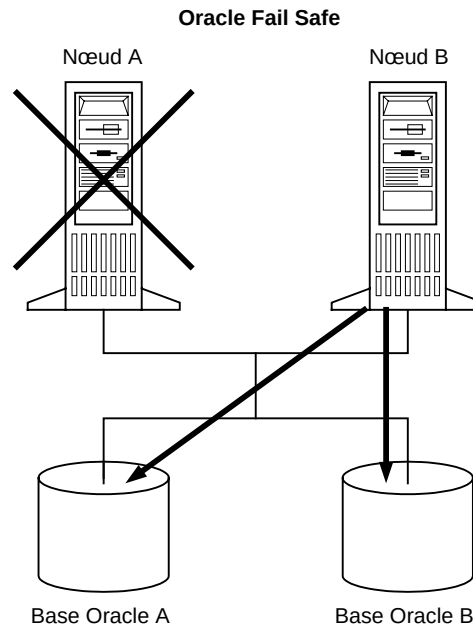
Principe d'Oracle Fail Safe



Dans notre exemple, chaque nœud héberge une instance Oracle. L'instance A pour le nœud A et l'instance B pour le nœud B. Les fichiers utilisés par chacune de ces bases de données sont situés dans des disques (obligatoirement au format NTFS de Microsoft), accessibles par chacun des nœuds, grâce à MSCS.

Si l'un des nœuds du cluster tombe en panne, le second prend en charge le travail effectué normalement par le premier (en addition à sa propre charge de travail). Dans ce cas, les applications du nœud A basculent (*fail over*) vers le nœud B. Les données de la base A sont accessibles à la machine B car les ressources disques NTFS sont communes pour les deux machines.

Figure 6-6

Arrêt d'un nœud Fail Safe

Après quelques minutes (maximum) d'indisponibilité de la base A, ses utilisateurs peuvent se reconnecter sur le serveur B et travailler aussi bien sur la base A que sur la base B. Dans ce cas, le serveur B est plus chargé car il assure le fonctionnement de deux bases au lieu d'une seule.

Oracle Fail Safe :

- minimise et gère les temps d'arrêt ;
- définit le nœud qui héberge la base et celui de « Fail Over » ;
- décrit les dépendances entre les bases et les autres ressources du cluster ;
- vérifie et valide la configuration de vos bases de données ;
- configure Oracle Net sur chaque machine ;
- surveille si vos bases sont lancées et fonctionnelles ;
- s'assure, lorsqu'une base s'arrête et passe d'un nœud vers l'autre, que toutes les ressources requises par la base s'arrêtent puis sont relancées dans le bon ordre.

Avec Oracle Fail Safe, vous n'avez aucune modification à effectuer sur vos applications clientes.

Une configuration cluster est perçue comme une seule adresse réseau. Lors d'une demande de connexion, Oracle Fail Safe s'appuie sur MSCS pour vous aiguiller automatiquement vers le nœud qui prend en charge la base de données active.

Que deviennent les utilisateurs connectés si le serveur s'arrête ?

C'est là que la réponse est la plus souvent évasive de la part d'Oracle et des autres promoteurs de clusters, car elle ne répond pas forcément à vos attentes. Il y a ce qui se fait automatiquement, ce qu'il est possible de faire en le programmant, et ce que l'on pourra faire avec de nouvelles versions.

En cas d'arrêt du serveur sur lequel des utilisateurs sont connectés, il y a trois questions importantes à se poser :

- Que deviennent les transactions en cours ?
- Le basculement vers la base survivante est-il automatique ?
- La connexion est-elle automatique à la nouvelle base ?

Ce qui se fait automatiquement

Vos transactions en cours sont annulées. Si l'application ne le prend pas en compte, vous devez vous reconnecter à la base de données, ce qui revient souvent à relancer l'application cliente. Lors d'une connexion, vous êtes toujours dirigé vers le nœud qui héberge la base Oracle Fail Safe active.

Ce que l'on peut faire en programmant

Une caractéristique de l'OCI (*Oracle Call Interface*), utilisée par les outils de développement qui s'interfacent à Oracle, permet aux postes clients de se connecter automatiquement au nœud survivant. De plus, les utilisateurs qui effectuent des ordres de consultation (SELECT) ne sont pas interrompus. Mais, encore une fois, il faut que le développement ou l'outil de développement utilisé exploitent ces possibilités...

Vous pouvez également programmer des applications pour tester, avant chaque ordre SQL, si le serveur distant est « vivant », et peut, dans ce cas, se connecter de manière transparente au serveur ayant pris la main.

Ce que l'on pourra faire avec de nouvelles versions

Éviter l'annulation des transactions en cours lorsque le serveur s'arrête. Tout ce que l'on souhaite faire autrement doit être programmé.

Fonctionnement d'Oracle Real Application Clusters

Oracle Real Application Clusters (RAC) est la version cluster la plus évoluée d'Oracle, qui nécessite la configuration matérielle la plus pointue et donc la plus chère. C'est un véritable « engin de compétition », qui a ses avantages et ses inconvénients.

Cas d'utilisation d'Oracle Real Application Clusters

Son intérêt est évident lorsque les contraintes et les performances de disponibilité sont plus fortes que celles qui sont possibles avec Oracle Fail Safe. On rencontre deux types d'utilisation :

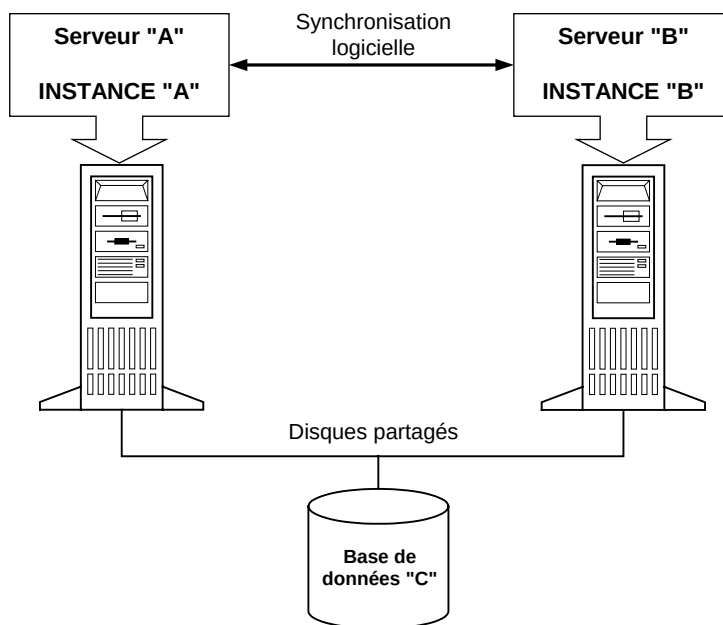
- un seul serveur Windows est incapable de supporter la charge des utilisateurs ou programmes connectés ;
- le système a besoin d'une haute disponibilité, proche de zéro temps d'arrêt toléré.

Principe de fonctionnement

Le point majeur qui distingue Oracle Real Application Clusters (RAC) est que la base Oracle fonctionne simultanément sur chacun des nœuds du cluster. La puissance de chacun des nœuds s'additionne et la sécurité est assurée si l'un des nœuds tombe en panne. Tableau idyllique ? Pas tant que cela, car le coût et la complexité technique de l'ensemble augmentent très fortement.

Figure 6-7

Principe d'Oracle Real Application Clusters



Disposer de plusieurs nœuds est une chose, mais comment s'effectue l'équilibrage de la charge entre ces machines ? Et bien, il n'y en a pas ! Ni au niveau du système d'exploitation, ni au niveau de la base de données Oracle. Un utilisateur est connecté à un nœud et travaille sur ce nœud, un point c'est tout. Si vous souhaitez vous connecter à des bases de données actives d'un cluster, c'est à l'application de le mettre en œuvre. Même chose si vous souhaitez effectuer un équilibrage des différentes charges machines.

Les machines ou applications clientes sont connectées à un nœud. Chaque nœud communique avec les autres grâce à un bus de communication haut débit, qui sert à propager aux autres nœuds les informations sur les données mises à jour sur un nœud, de façon qu'ils soient tous au même niveau d'information. C'est là une des limitations d'Oracle Real

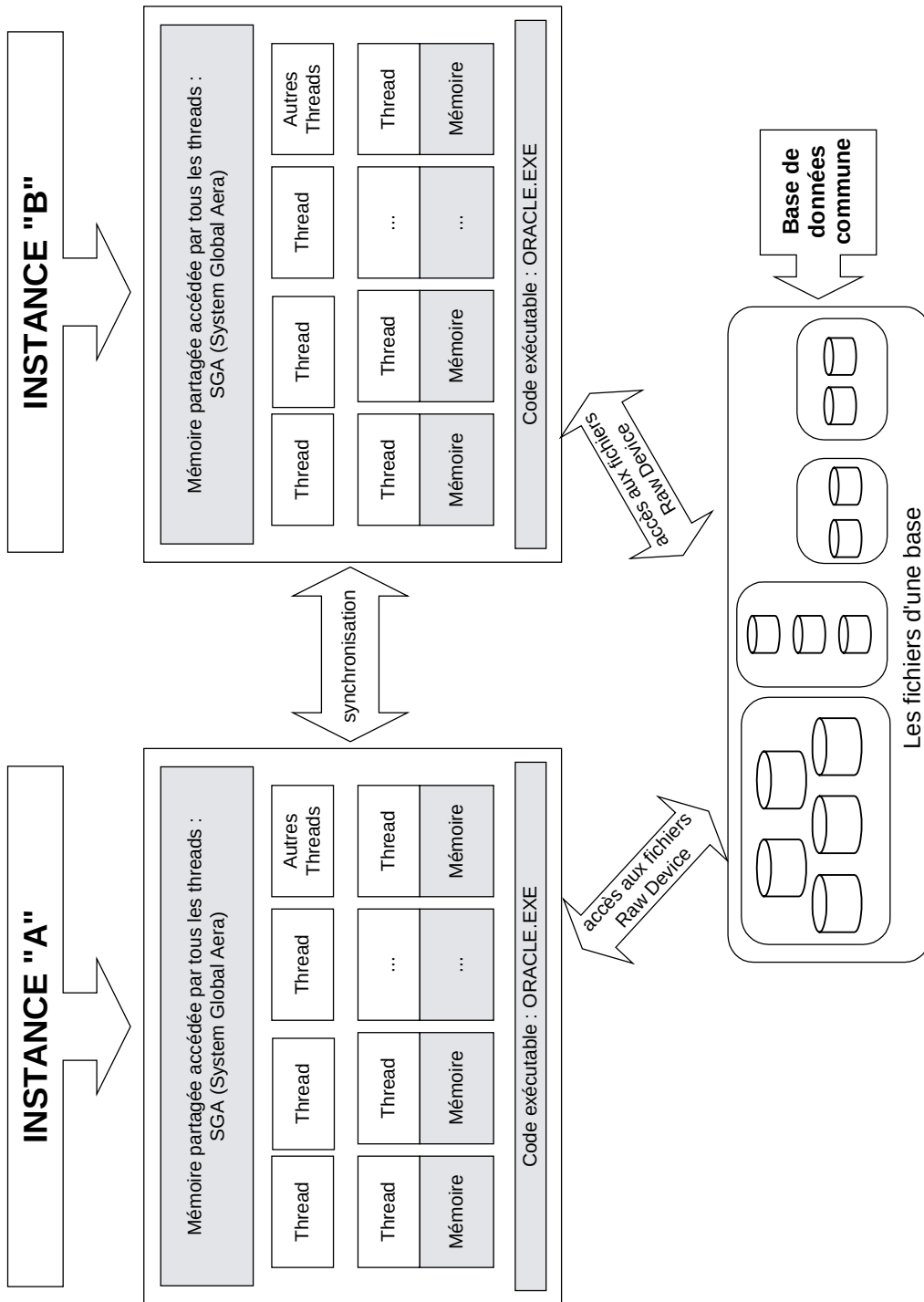


Figure 6-8
Les composants d'Oracle RAC

Application Clusters (RAC) : si tous les nœuds effectuent des modifications intensives des données, le bus de communication devient très chargé et peut même arriver à saturation. Ceci est un point difficile à détecter et nécessite des compétences pointues pour le corriger.

Chaque nœud d'un cluster accède à des ressources communes : les fichiers de données. Pour que différents nœuds accèdent en même temps à un fichier, il faut que ce fichier soit du type Raw Devices présenté au chapitre 27, *Configurer les disques pour Oracle*. Le Raw Devices est plus un type de partition disque (comme FAT, NTFS, HSFS) qu'un fichier. L'inconvénient majeur est qu'il ne se manipule pas comme un fichier, et que cela complexifie singulièrement les opérations de sauvegarde et de restauration. Là aussi, des compétences pointues sont requises pour administrer ce type de fichier.

Lorsqu'un nœud s'arrête, les autres nœuds continuent d'accéder de manière transparente aux données.

Que deviennent les transactions en cours sur le nœud qui « tombe » ?

Les transactions validées sont assurées, les autres transactions en cours mais non validées sont annulées.

Le basculement vers un nœud survivant est-il automatique ?

Non. Il ne faut pas compter sur une transaction qui s'effectue sur un nœud et, après quelques instants d'arrêt, continue sur un autre nœud. Ce type de fonctionnement très évolué doit être développé dans vos applications.

La connexion est-elle automatique et le « contexte d'utilisation » est-il sauvegardé ?

Non. C'est à l'application de tenir compte de tous ces éléments.

Résumé

Après une présentation de l'univers des clusters, Oracle Fail Safe et Oracle Real Application Clusters (RAC) ont été abordés.

On peut réserver Oracle Fail Safe aux configurations qui ont les caractéristiques suivantes :

- le serveur Windows est capable de supporter tout seul la charge de la base de données ;
- ressources et connaissances techniques des clusters limitées ;
- faible coût de la configuration (Oracle est gratuit) ;
- haute disponibilité d'Oracle sous Windows ;
- peu d'impact sur vos applications ;
- relativement facile à mettre en œuvre et à utiliser pour des non spécialistes des clusters.

On peut réserver Oracle Real Application Clusters (RAC) aux configurations qui ont les caractéristiques suivantes :

- ressources et connaissances techniques des clusters importantes ;
- coût important de la configuration (option Oracle facturée) ;
- haute disponibilité d'Oracle sous Windows ;
- augmentation de la puissance totale du cluster par ajout de processeurs ou de nœuds ;
- les applications doivent obligatoirement être écrites pour tirer profit du cluster ;
- nécessite de fortes compétences du fonctionnement interne des clusters Oracle.

Notre recommandation finale nous porte vers Oracle Fail Safe, qui donne avec une technique simple à la fois une bonne disponibilité et une tolérance aux pannes. Oracle Real Application Clusters (RAC) nécessite plus de compétences et des développements spéciaux pour en tirer le meilleur parti.

Le moyen le plus simple pour vérifier notre avis est d'exiger de contacter des références utilisant Oracle Fail Safe ou Oracle Real Application Clusters (RAC) sous Windows pour lesquelles la mise en œuvre, les attentes et les compétences techniques sont proches des vôtres.

Les processus d'Oracle 10g

Dans ce chapitre :

- les threads Oracle ;
- les threads indispensables ;
- les threads optionnels ;
- les threads utilisateurs.

Le chapitre 5, *Oracle 10g sous Windows*, a présenté le processus Oracle 10g. Ce chapitre est maintenant consacré aux threads du processus oracle.exe, qui assurent le fonctionnement d'une instance Oracle 10g. Les prochains chapitres vont présenter tour à tour les principaux composants d'une base Oracle 10g : fichiers, threads et espaces mémoire. Ici, nous entrons dans le détail du fonctionnement interne de chacun de ces threads, pour vous permettre de mieux comprendre son rôle, son impact sur le fonctionnement d'une instance et les performances de vos bases Oracle 10g.

Les threads indispensables, optionnels et utilisateur

Le fonctionnement d'une base Oracle est assuré par un ensemble de threads imbriqués qui réalisent de nombreuses actions. Pour plus de simplicité, nous avons regroupé les threads en trois familles : les indispensables, les optionnels et les threads utilisateur.

Les threads indispensables sont présents dès qu'une base Oracle fonctionne. Ils sont requis pour en assurer le fonctionnement minimal. Si l'un d'eux s'arrête, la base de données n'est plus opérationnelle.

D'autres threads peuvent être lancés pour assurer des fonctions complémentaires, comme la réplication ou la sauvegarde automatique de vos fichiers redo-log. Si l'un de ces threads optionnels n'est pas démarré, cela ne met pas en cause le fonctionnement global de la base de données. Seule la tâche assurée par ce threads optionnel ne sera pas réalisée.

Les threads utilisateur sont mis en œuvre dès qu'un utilisateur ouvre une connexion avec sa base Oracle. Leur existence et leur utilité sont très souvent méconnues.

Les threads indispensables

Pour un fonctionnement minimal, Oracle utilise 4 threads :

- DBWR (*Database Writer*) ;
- LGWR (*Log Writer*) ;
- CKPT (*Checkpoint*) ;
- PMON (*Process Monitor*) ;
- SMON (*System Monitor*).

La figure suivante décrit les liens inter-threads de l'architecture d'une instance Oracle 10g. Ce sont ces threads qui relient les fichiers de la base de données, la zone mémoire réservée à l'instance SGA (System Global Area), ainsi que la mémoire allouée à chaque connexion utilisateur PGA (Process Global Area). Nous commenterons largement cette figure tout au long de ce chapitre.

Ces threads correspondent à une instance unique. Comme présenté au chapitre 5, *Oracle 10g sous Windows*, on peut les visualiser à l'aide de l'*Oracle Administrative Assistant for Windows* ou encore depuis Oracle Enterprise Manager.

Dans ce chapitre, nous traitons chaque thread séparément. La tâche est difficile puisqu'ils sont très imbriqués et les dépendances mutuelles sont nombreuses.

Le threads DBWR

Le threads DBWR (*Database Writer* ou *Dirty Buffer Writer*) transfère les blocs de données modifiés ou « sales » (« dirty buffers ») du buffer mémoire de la SGA dans les fichiers disque de la base de données. Dès qu'un ordre SQL de type INSERT, UPDATE, DELETE intervient, il travaille prioritairement avec les buffers de données en mémoire, pour plus de performance. Les données non modifiées sont copiées dans la zone d'annulation (ou UNDO) de la SGA et les données modifiées dans la zone des buffers de données. La zone mémoire redo-log (destinée à être traitée par le Log Writer) est elle aussi modifiée.

En cas de validation de transaction (ordre COMMIT), le DBWR libère la place des enregistrements dans le Segment UNDO. S'il y a annulation de transaction (ou ROLL-BACK), les enregistrements non modifiés lus dans le UNDO Segment supprimeront les enregistrements modifiés en buffer de données et les rétabliront en leur état initial.

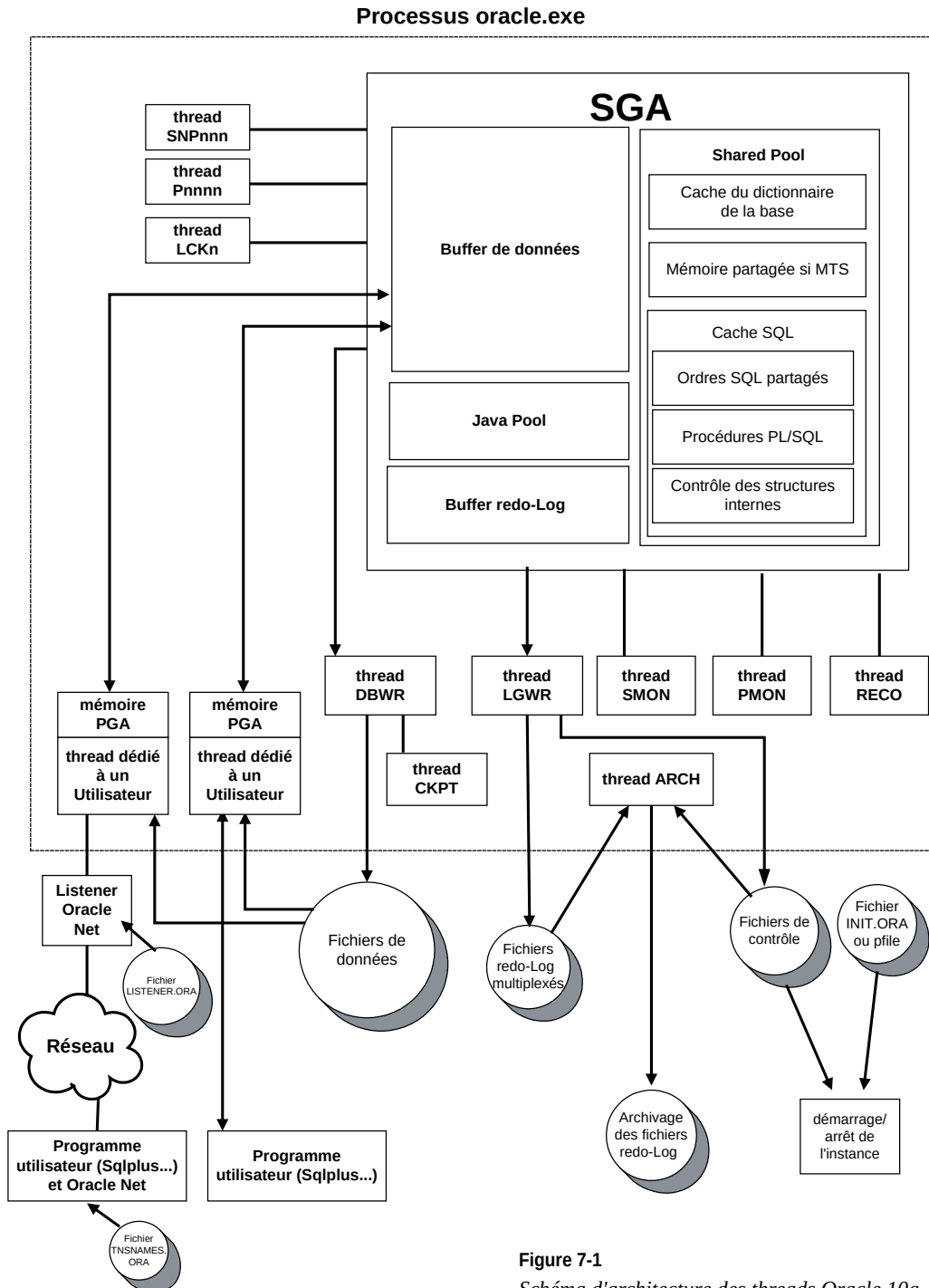


Figure 7-1

Schéma d'architecture des threads Oracle 10g

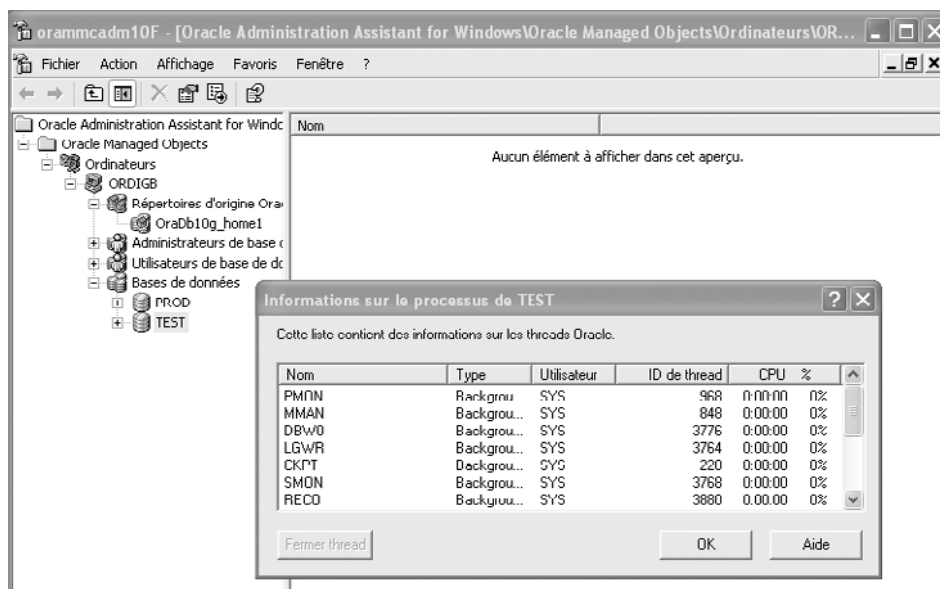


Figure 7-2

Les threads d'un processus

Le rôle de DBWR est de vérifier qu'il y a toujours suffisamment de buffers libres en mémoire. Comme le nombre de blocs existants en mémoire est fixe, chaque buffer modifié (ou « sale ») diminue le nombre de buffers libres disponibles. Dans ce cas, si un ordre SQL a besoin de lire des données depuis les fichiers de données pour les placer dans les buffers de données mémoire, un algorithme LRU (Least Recently Used) écrit les plus anciennes données modifiées sur le disque pour libérer de la place mémoire.

Pour optimiser les entrées/sorties disque, les écritures dans les fichiers de la base de données sont différées. La sécurité des transactions est assurée par les fichiers redo-log et le threads LGWR.

Le comportement de DBWR est contrôlé par le paramètre d'initialisation DB_WRITERS, qui permet de démarrer plusieurs threads DBWR, afin d'augmenter le taux d'écriture sur disque dans les systèmes très fortement sollicités. Nous vous conseillons de démarrer une base Oracle 10g avec un seul thread DBWR et d'en augmenter progressivement le nombre dans les systèmes très sollicités en entrées/sorties disque.

Lors de tous les transferts entre la mémoire et les fichiers présents sur le disque, la plus petite unité de transfert est le « bloc Oracle » dont la taille est fixée lors de la création de chaque base de données. En effet, le système d'exploitation accède aux disques par des « unités élémentaires » qui lui sont propres. Ce bloc permet une cohérence entre les blocs manipulés par Oracle et les unités élémentaires manipulées par le système d'exploitation. Dans beaucoup de systèmes, l'unité d'échange de l'OS est 512 octets, et celle d'Oracle est un multiple de cette valeur : 2048, 4096, etc. Le chapitre 28, *Optimisation et performances traite de l'influence du bloc Oracle sur les performances.*

Les threads LGWR et CKPT

Les fichiers redo-log garantissent la préservation des données validées, même si la base est arrêtée brutalement (coupure électrique par exemple). Le thread LGWR a pour mission d'écrire toutes les informations utiles à cette sécurité dans les fichiers redo-log.

Dès qu'une transaction est validée, Oracle écrit les données modifiées à deux emplacements différents, de façon à pouvoir « repartir » si un problème matériel survient. La première copie vue précédemment est assurée par le Database Writer dans les fichiers contenant les données. Cette copie n'est pas forcément immédiate : pour augmenter les performances et éviter des goulots d'étranglement, un délai d'écriture peut exister. Pour conserver l'intégralité des données présentes en mémoire mais en attente d'écriture sur disque, une seconde copie immédiate est assurée par le Log Writer dans les fichiers redo-log.

Dès qu'un COMMIT ou un ROLLBACK intervient, le thread LGWR (Log Writer) écrit immédiatement les données modifiées depuis la zone mémoire redo-log dans les fichiers redo-log, suivi de l'ordre de validation (COMMIT) ou d'annulation (ROLLBACK). Ainsi, toute modification validée ou annulée est immédiatement écrite sur le disque, puis la zone mémoire redo-log occupée est libérée.

À intervalles réguliers, toutes les données modifiées et présentes dans la SGA sont écrites dans les fichiers de la base de données par le thread DBWR. Cet événement se nomme un *checkpoint*. Le thread CKPT signale les checkpoints au thread DBWR et modifie l'ensemble des fichiers qui composent la base de données, pour que le numéro d'ordre du plus récent checkpoint soit inscrit en en-tête de fichier.

Le thread System Monitor

Le thread SMON (System Monitor) surveille la base de données lors de son démarrage puis au cours de son fonctionnement.

La principale fonction du thread SMON a lieu lors du démarrage de la base de données : il vérifie si le dernier arrêt a été correctement effectué. Si tel est le cas, il ne fait rien. Mais en cas d'arrêt brutal, il existe certainement des transactions en cours qui n'ont été ni validées, ni invalidées. SMON lit alors les informations contenues dans les UNDO segments (l'emplacement où sont conservées les données en attente de validation) puis les annule. En ce qui concerne les enregistrements validés, SMON récupère dans les fichiers redo-log ceux qui ont été modifiés (par un COMMIT ou un ROLLBACK) mais n'ont pas encore été écrits dans la base Oracle, et ce afin de les y insérer. Les autres enregistrements présents dans les fichiers redo-log, mais qui n'ont pas été validés ou annulés explicitement, sont alors annulés par SMON. Enfin, SMON libère toutes les ressources de la base de données (verrous, segments temporaires...).

Lorsqu'elle est en cours de fonctionnement, SMON surveille l'activité de la base. Il recycle les segments temporaires (utilisés par exemple lors des tris de requêtes SQL) devenus inutiles. Tous les tris ne pouvant avoir lieu en mémoire, c'est SMON qui assure ce rôle important : ainsi, les nouvelles transactions ayant besoin d'espace de tri disposent d'un maximum de place.

SMON vérifie aussi que des espaces libres subsistent dans les fichiers de la base de données. Dans ce cas, il essaie de les regrouper. En effet, même si vous disposez de beaucoup de place dans un fichier de la base, cet espace peut ressembler à du « gruyère ». Par exemple, il n'est pas possible de créer une table de 10 Mo, si le plus grand espace libre contigu est inférieur à cette taille (il existe des contournements). Cependant, les capacités de SMON étant limitées, vous devez parfois effectuer un Export/Import pour réorganiser physiquement le contenu de vos fichiers et regrouper l'ensemble des espaces libres en un seul espace contigu.

SMON intervient aussi dans le fonctionnement en mode Parallel Server ou cluster. Il vérifie que l'ensemble des bases de données composant le cluster sont actives.

SMON peut aussi être appelé par d'autres threads. Ainsi, si le thread Database Writer a besoin d'espace disque pour effectuer un tri, il demande à SMON s'il peut en libérer.

Le fonctionnement de SMON est automatique : aucune action de l'administrateur de la base de données n'est requise. C'est l'un des points forts d'Oracle par rapport à ses concurrents.

Si SMON s'arrête, il faut redémarrer l'instance Oracle.

Le thread Monitor

Le thread PMON (*Process Monitor*) nettoie les transactions défailtantes, comme celle d'un poste distant arrêté brutalement durant une transaction. Ce nettoyage libère les zones mémoire allouées, supprime les verrous posés par les transactions et annule les ressources affectées aux threads de la transaction.

Le rôle de PMON est très important si vous utilisez un système comportant de nombreux utilisateurs ou encore si vous effectuez des requêtes lourdes. Chaque connexion à une base Oracle consomme quelques mégaoctets de mémoire et du temps processeur. Si un utilisateur arrête brutalement son PC au cours d'une longue requête SQL, il peut ainsi bloquer inutilement un ensemble de ressources, si PMON ne détecte pas et ne nettoie pas les transactions anormalement interrompues. Ce sont des threads comme PMON qui font la différence entre des systèmes qu'il faut fréquemment « rebooter » et d'autres, stables dans le temps.

Si PMON s'arrête, il faut redémarrer l'instance Oracle. Dans ce cas, SMON annulera (ordre ROLLBACK) toutes les transactions en attente de validation qu'il trouvera dans les buffers de données.

Les threads optionnels

Les threads détachés optionnels sont les suivants :

- Listener ou listener Net ;
- ARCH ou Archiver ;
- RECO ou Recover ;
- SNPN ou Snapshot ;

- Dnnnn et Snnnn Dispatcher et Server ;
- Pnnnn ou Parallel Query Slave.

Le processus listener Oracle Net

Le listener Oracle Net n'est généralement pas compris dans la liste des processus indispensables au fonctionnement d'Oracle. Pourtant, il doit être lancé pour permettre d'établir des connexions client-serveur avec la base de données.

Son lancement n'est pas lié à celui de la base de données et il possède ses propres fichiers de configuration. Son fonctionnement sous forme d'un service Windows est détaillé au chapitre 15, *Oracle Net, le middleware Oracle*.

Le thread ARCH

Le thread ARCH (Archiver) n'existe que si vous utilisez la base de données en mode ARCHIVELOG. Il est responsable de la copie des fichiers redo-log, lorsqu'ils sont saturés, vers leur destination de stockage (disque ou bande). Il faut bien distinguer le thread ARCH responsable de la copie, de l'ordre à donner au niveau de la base de données pour lui indiquer qu'elle est en mode ARCHIVELOG et que les redo-log seront conservés en historique. Ce sont deux étapes séparées.

Il est très dangereux de se fier à la seule présence du thread ARCH pour en conclure que la base est en mode ARCHIVELOG.

Si la destination de stockage est saturée et ne peut accueillir de nouveaux fichiers, la base de données se bloque. En effet, Oracle considère qu'il vaut mieux bloquer une base de données active plutôt que renoncer à un moyen de sauvegarde déterminant sans en avertir l'administrateur. Lorsqu'elle se bloque, la base de données est « silencieuse ». Seules de nouvelles connexions ou l'exploration des fichiers d'alerte de la base de données signalent cette situation. En revanche, si l'on place la base en mode ARCHIVELOG sans démarrer le thread ARCH, la base de données se bloquera dès qu'elle aura besoin « d'historiser » un fichier redo-log saturé. Un message sera alors écrit dans le fichier d'alerte.

Pour démarrer le thread ARCH, le paramètre d'initialisation ARCHIVE_LOG_START doit avoir pour valeur TRUE et la base de données doit être en mode ARCHIVELOG.

Le thread RECO

Le thread RECO (Recover) intervient uniquement si vous utilisez la réplication ou base de données distribuée. Dans ce cas, il faut que l'option « distributed database » soit installée dans la base.

La liste des options disponibles installées dans la base de données s'affiche lors de chaque connexion à SQL*Plus. Vous pouvez aussi interroger la vue V\$VERSION.

Le thread RECO est lancé si le paramètre d'initialisation de l'instance DISTRIBUTED_TRANSACTION est supérieur à zéro (pour les bases de données distribuées, sa valeur par défaut est 32).

Le thread SNPnn

Le thread SNPnn (*Snapshot*, *nn* représentant un nombre entier) était initialement utilisé pour le traitement des snapshots ou clichés. Lorsqu'une demande de cliché est placée en file d'attente dans la base de données, le thread SNPnn est chargé de le traiter.

Un thread SNPnn défaillant empêche le traitement des clichés et génère un message dans le fichier d'alerte. Le fonctionnement global de la base de données n'est pas affecté par l'arrêt d'un thread SNPnn.

Les threads SNPnn sont contrôlés par les paramètres d'initialisation de l'instance : JOB_QUEUE_PROCESSES, JOB_QUEUE_INTERVAL et JOB_QUEUE_KEEP_CONNECTIONS (pour certains portages, JOB peut être remplacé par SNAPSHOT). JOB_QUEUE_PROCESSES détermine le nombre de threads SNPnn démarrés au lancement de l'instance. JOB_QUEUE_INTERVAL définit la fréquence de réveil des threads SNPnn qui vont scruter la file d'attente des clichés à diffuser. JOB_QUEUE_KEEP_CONNECTIONS conserve les liens avec des bases de données distantes (data-base links) ouvertes.

Les threads Dnnnn et Snnnn

Les threads Dnnnn (Dispatcher) et Snnnn (Server) avec *nnnn*, une suite de nombres entiers, n'interviennent que pour des systèmes ayant mis en place une configuration MTS (Multi-Thread Server). Les threads Dnnnn distribuent les demandes de connexion des postes client-serveur distants aux serveurs Snnnn.

Dans un système MTS, le listener Net détermine si une demande de connexion d'un utilisateur distant doit utiliser un thread dédié ou partager un thread Snnnn avec d'autres utilisateurs.

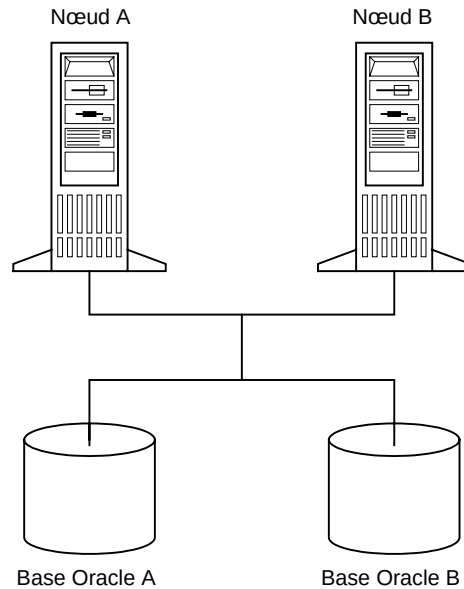
Les threads Dnnnn et Snnnn sont contrôlés par les paramètres d'initialisation de l'instance : MTS_DISPATCHERS, MTS_MAX_DISPATCHERS, MTS_SERVER et MTS_MAX_SERVER. MTS_DISPATCHERS et MTS_SERVER déterminent respectivement le nombre de threads Dnnnn et Snnnn lancés au démarrage de l'instance. Le système s'allouant automatiquement de nouveaux threads Dnnnn et Snnnn, les deux autres paramètres fixent les seuils à ne pas dépasser.

Le thread Pnnnn

Le thread Pnnnn (*Parallel Query Slave*) avec *nnnn*, une suite de nombres entiers, est automatiquement lancé et arrêté si la base Oracle est utilisée dans un environnement Oracle *Real Application Clusters* (RAC) ou cluster de base de données. Dans cette configuration, des bases hébergées sur des machines différentes constituent un « tout » logique.

Figure 7-3

Principe d'un cluster
de bases Oracle 10g



Ces configurations très élaborées, présentées au chapitre 6, *Les clusters Oracle*, sont destinées à des applications nécessitant une haute disponibilité ou de hautes performances.

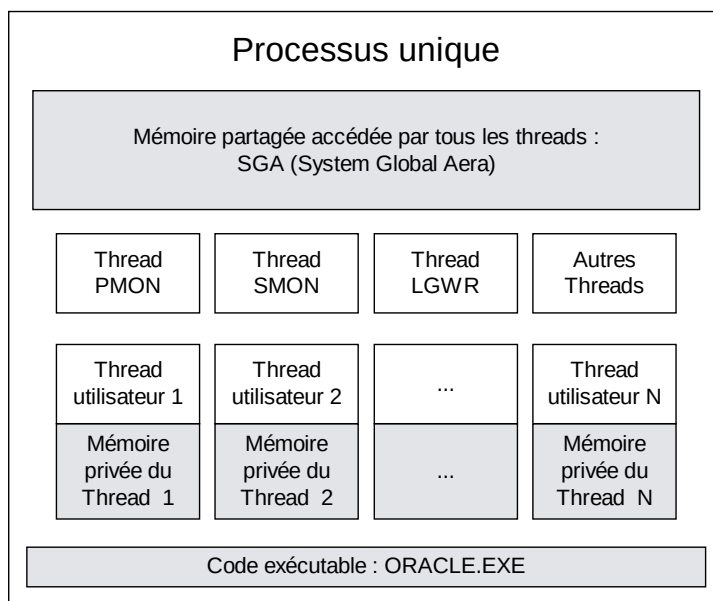
Modes single-task et two-tasks

Suivant les systèmes d'exploitation, Oracle fonctionne en mode *single-task* ou en mode *two-tasks*. Ces modes de fonctionnement sont internes aux produits Oracle et n'ont aucune incidence sur votre façon de travailler ou de développer des applications.

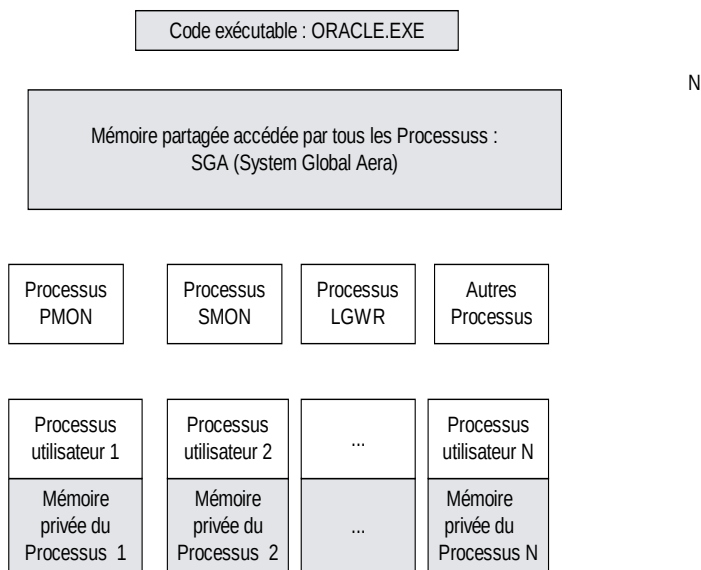
En mode *single-task*, le noyau Oracle, le programme utilisateur et le programme interface qui les lient, fonctionnent dans le même thread. C'est le cas de VAX VMS (que l'on rencontre de moins en moins souvent) et de Windows : lorsque de nouveaux utilisateurs se connectent à la base Oracle, ils donnent naissance à des « threads » supplémentaires qui s'exécutent au sein du même processus.

En mode *two-tasks*, le noyau Oracle, le programme utilisateur et le programme interface qui les lient, fonctionnent dans des processus différents. C'est le cas de Linux et de l'ensemble des Unix : lorsque des utilisateurs supplémentaires se connectent à la base Oracle, ils engendrent de nouveaux processus, appelés *shadow processus*. Ils possèdent une « vie » indépendante de celle des autres processus.

L'approche est donc différente pour visualiser les caractéristiques (mémoire utilisée, pourcentage de CPU...) d'un processus utilisateur sous Windows et sous Linux. Dans le premier cas, il faut explorer les threads d'un processus, dans l'autre, il faut consulter les processus gérés par le système d'exploitation.

**Figure 7-4**

Principe du mode single-task : des threads au sein d'un processus

**Figure 7-5**

Principe du mode two-tasks : des processus coordonnés

Les threads utilisateur

Ce sont les plus méconnus, dans la mesure où ils sont intégralement gérés par la base Oracle. Leur rôle est fondamental. Dès que l'application demande l'accès à l'instance, ce sont eux qui vérifient que les données sont présentes en mémoire. Dans l'affirmative, le thread utilisateur traite vos ordres SQL. Dans la négative, les threads utilisateur lisent les fichiers de la base de données pour les « monter » en mémoire SGA, avant de traiter l'ordre SQL.

Ce n'est pas un thread permanent de l'instance qui assure la lecture des données contenues dans vos fichiers, mais le thread utilisateur créé lors de chaque connexion. Il assure le lien entre le programme de l'utilisateur, les threads de la base de données, la SGA et les fichiers qui composent la base.

Ce thread peut prendre plusieurs formes suivant l'architecture retenue. Étudions le thread utilisateur dans les cas suivants :

- configuration sans réseau ;
- configuration réseau en mode client-serveur ;
- configuration MTS Multi-Thread Server.

Configuration sans réseau

C'est le type de configuration la plus simple, celle que vous utilisez si la base de données et l'application sont situées sur la même machine. Dans ce cas, le thread utilisateur est en lien direct avec l'application.

C'est l'application qui demande à se connecter à l'instance créant le thread utilisateur. Il existe un thread utilisateur par application connectée à l'instance.

Chaque thread utilisateur est responsable de la lecture des données. Il donne naissance à la PGA (*Program Global Area* ou *Process Global Area*), zone mémoire privée allouée à chaque thread utilisateur.

Cette architecture, présente l'inconvénient de ne pas être extensible. Le programme utilisateur, son thread et la base de données se trouvent sur la même machine. Vous pouvez rapidement atteindre une saturation des ressources du système par accumulation de thread.

Les deux configurations suivantes permettent de lever ces limites, en permettant d'abord au programme utilisateur de fonctionner sur une autre machine (le client-serveur), en mettant ensuite en œuvre l'option Multi-Thread Server (MTS) pour limiter le nombre de threads dédiés à l'utilisateur sur le serveur de données.

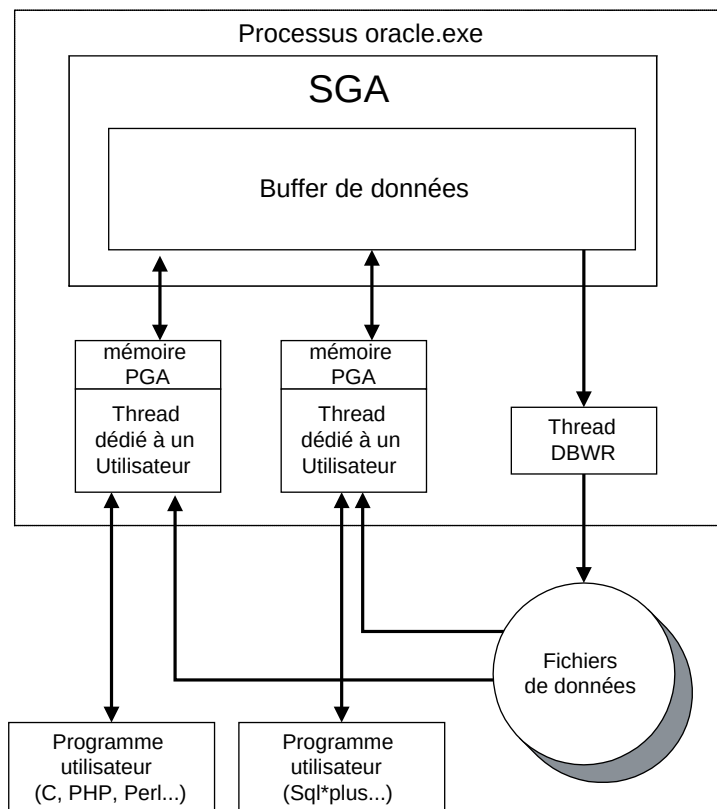


Figure 7-6

Thread utilisateur sans réseau

Configuration réseau en mode client-serveur

En mode client-serveur, le programme utilisateur est situé sur une machine (le client) et la base de données sur le serveur. Le lien entre ces deux programmes est assuré par le middleware de communication Oracle, traité au chapitre 15, *Oracle Net, le middleware Oracle*.

Ce mode introduit le thread Listener Oracle Net qui fonctionne en résident sur le serveur : il est en attente d'une demande de connexion en provenance d'un client. Dès qu'une demande intervient, le Listener Oracle Net crée le thread utilisateur et gère tous les échanges entre le poste client et le serveur.

Cette architecture offre une grande souplesse. Des programmes utilisateur fonctionnant sous des OS différents peuvent accéder à des données situées dans une même base. De plus, en déportant le programme utilisateur sur une autre machine, vous pouvez équilibrer les charges entre vos ordinateurs.

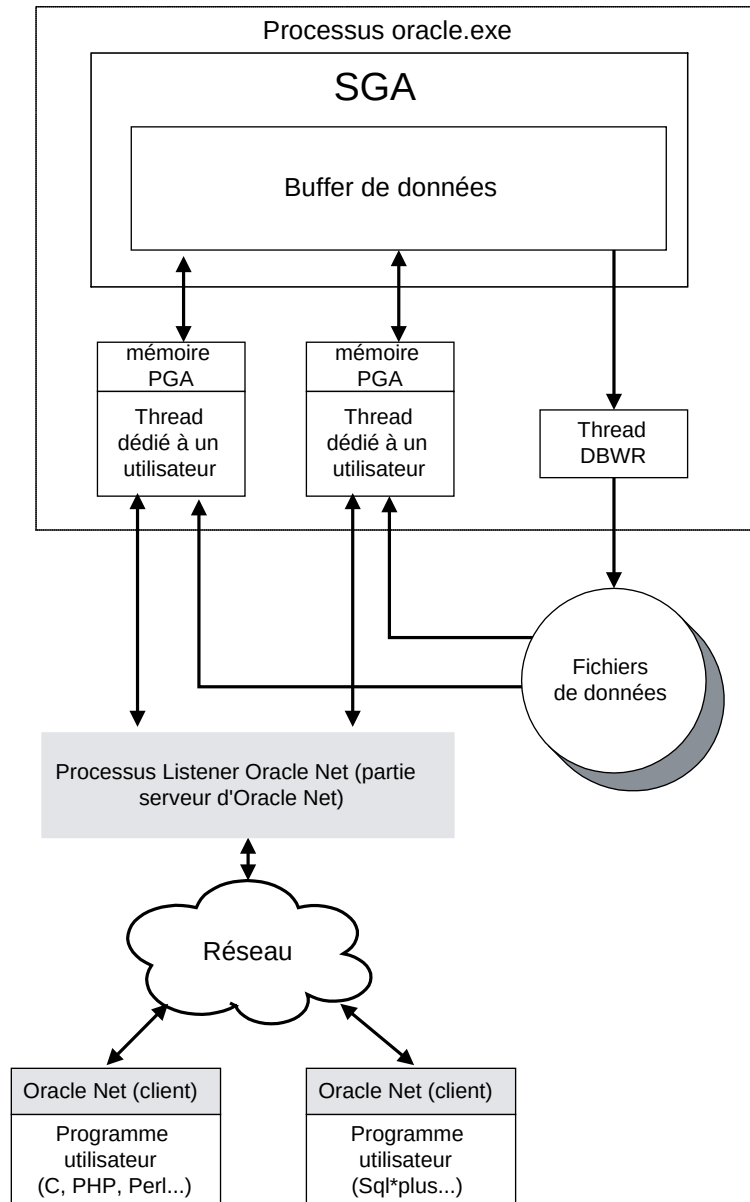


Figure 7-7

Thread utilisateur en mode client-serveur

Configuration MTS (Multi-Thread Server)

Dans les deux configurations précédentes, il y a toujours un thread utilisateur créé par un programme client. Ces threads utilisateur consomment non seulement de la puissance machine (CPU), mais aussi de la mémoire. Pour augmenter le nombre de connexions acceptées par un serveur, la configuration MTS (*Multi-Thread Server*) propose une solution de multiplexage des threads utilisateur.

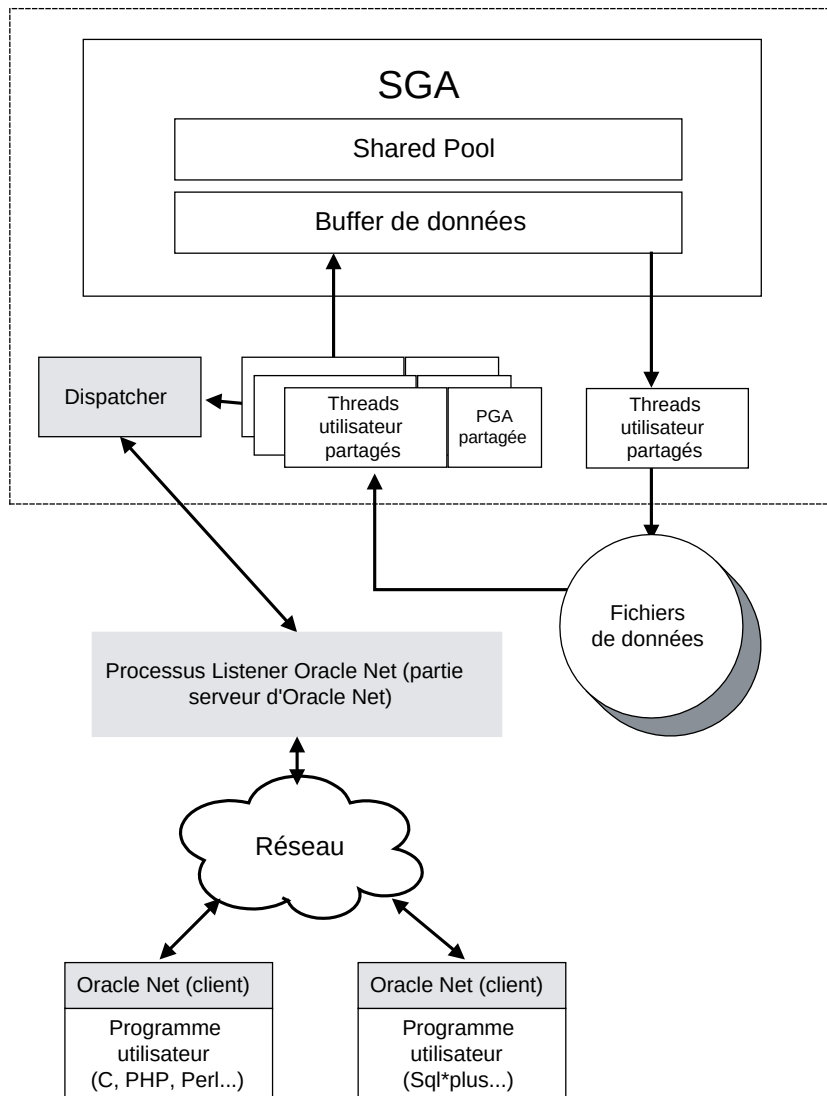


Figure 7-8

Threads utilisateur en configuration MTS (Multi-Thread Server)

Cette configuration économise les ressources serveur grâce à la présence des dispatchers et des threads utilisateur multiplexés. À l'inverse, certaines opérations comme le tri des données sont maintenant partagées en SGA, la taille de la Shared Pool augmente alors fortement.

Oracle préconise d'utiliser la configuration MTS à partir de 100-150 utilisateurs simultanés.

Résumé

Ce chapitre a détaillé le fonctionnement et l'interaction des différents threads pouvant exister autour d'une base de données Oracle 10g. Après une présentation générale, les threads ont été classés en trois familles : les indispensables présents dans chaque base Oracle 10g en fonctionnement, les optionnels qui peuvent être mis en place suivant les options retenues et enfin les threads utilisateur.

La connaissance de ces threads permet de mieux comprendre le fonctionnement interne d'Oracle 10g afin d'en tirer le meilleur parti et de déterminer quels leviers actionner pour en optimiser le fonctionnement.

L'utilisation de la mémoire par Oracle 10g

Dans ce chapitre :

- l'utilisation générale de la mémoire par Oracle 10g ;
- la mémoire partagée SGA (*System Global Area*) ;
- la mémoire allouée pour chaque programme PGA (*Program Global Area*) ;
- l'espace mémoire de travail des utilisateurs.

Dans tout système informatique, l'utilisation de mémoire est synonyme de performance. Les données auxquelles on accède et qui sont manipulées en mémoire le sont beaucoup plus rapidement que sur disque. C'est pourquoi Oracle 10g, au moyen de techniques efficaces, utilise le plus possible la mémoire physique de vos ordinateurs.

Ce chapitre traite des différents types de mémoires existants et la manière dont Oracle 10g les utilise. Il est important de bien comprendre ces éléments, car ils interviennent dans les opérations d'amélioration des performances.

Mémoire physique et mémoire virtuelle

Un point essentiel consiste à bien définir le type de mémoire dont on parle. Pour ceux qui ne sont pas familiarisés avec ces concepts, rappelons que la mémoire est tout d'abord la mémoire RAM (*Read Access Memory*), c'est-à-dire les circuits installés dans votre ordinateur. C'est le type de mémoire le plus rapide, le plus performant et tous les systèmes d'exploitation l'utilisent.

Néanmoins, la quantité de mémoire RAM n'est pas illimitée. Pour l'augmenter artificiellement, tous les systèmes d'exploitation attribuent à la mémoire RAM un espace disque de débordement. On parle alors de mémoire physique (RAM) et de mémoire virtuelle (espace de *pagination* sous Windows).

La mémoire virtuelle permet de prendre en charge une consommation mémoire plus importante que celle qui tiendrait uniquement dans la mémoire physique. Cela fonctionne particulièrement bien pour les threads qui ne s'activent qu'occasionnellement : leur temps de transfert entre le disque (lieu de stockage) et la mémoire (lieu de traitement) est alors minime.

S'il y a trop de threads actifs en mémoire virtuelle, l'échange continu entre la mémoire physique et virtuelle risque d'engorger votre système. On parle alors de swap ou de pagination du système. Pour pallier la difficulté, il convient soit d'augmenter la mémoire RAM, soit de diminuer la consommation RAM des différents processus.

Le swap transfère un processus entier de la mémoire vers le disque, la pagination ne transfère que quelques pages. Dans un souci de rapidité, les systèmes d'exploitation utilisent massivement la pagination et ils ont recours au swap lorsqu'ils sont engorgés.

Le détail de la consommation mémoire de Windows peut être visualisé de nombreuses façons. L'une des plus simples est d'utiliser le Gestionnaire de tâches.

Mémoire virtuelle sous Windows

Les systèmes d'exploitation Windows 2000, 2003, XP, .NET sont généralement des systèmes 32 bits dont la capacité d'adressage est de 4 Go. Sur la plupart des systèmes, Windows se réserve 2 Go d'espace adresse pour le fonctionnement du système d'exploitation et alloue 2 Go aux processus du système. L'option Microsoft /3GB sur ses serveurs haut de gamme permet de répartir différemment la mémoire : 1 Go pour le système et 3 Go pour les processus utilisateur.

Pour une base de données comme Oracle, cela signifie que chaque processus `oracle.exe` peut adresser un maximum de 2 Go (ou 3 Go avec l'option précitée). La mémoire totale occupée par une instance, sa mémoire partagée et la mémoire privée de tous les utilisateurs connectés, ne pourra excéder cette valeur.

Pour une configuration nécessitant plus de mémoire, Oracle dispose de versions « 64 bits » pour versions « 64 bits » de Windows (2003, XP...). Cette combinaison repousse les limites actuelles des systèmes Windows « 32 bits ».

Toute l'architecture d'Oracle 10g a été conçue en vue d'une utilisation massive de la mémoire physique RAM. Pour des raisons de performance, il est impératif d'éviter que ses différents composants ne se retrouvent dans la zone de mémoire virtuelle.

Les zones mémoire utilisées par Oracle 10g

Comme toute base de données multi-utilisateur, Oracle 10g a besoin de mémoire pour assurer diverses fonctions, à savoir :

- mettre en mémoire, à la disposition de multiples utilisateurs, un maximum d'informations et de données provenant de la base ;
- permettre aux programmes qui gèrent la base Oracle 10g (le processus et ses threads) de fonctionner en mémoire ;
- assurer la transmission des données entre la base Oracle 10g et les threads des utilisateurs ;
- assurer la liaison avec l'application qui utilise les données.

En outre, la mémoire doit accueillir tous les processus nécessaires au fonctionnement de Windows.

Les paramètres statiques et dynamiques

Toutes les bases Oracle utilisent un fichier d'initialisation pour démarrer. Il peut être local (sous forme d'un fichier texte modifiable) ou avoir la forme d'un fichier binaire persistant. Ce point est détaillé au chapitre 23, *Gestion de l'espace disque et des fichiers*. Parmi les paramètres définis dans le fichier d'initialisation figurent tous ceux qui concernent l'allocation mémoire d'Oracle.

Pour assurer une plus grande disponibilité, les valeurs d'allocation mémoire des instances sont maintenant presque toutes dynamiques, c'est-à-dire qu'elles peuvent être modifiées alors que la base fonctionne :

```
alter system set paramètre=valeur scope= memory ;
```

Les valeurs ainsi fixées peuvent avoir un impact sur la mémoire, être enregistrées dans le fichier d'initialisation persistant ou les deux actions à la fois :

```
alter system set paramètre=valeur scope= memory ;  
alter system set paramètre=valeur scope= spfile ;  
alter system set paramètre=valeur scope= both ;
```

S'il s'agit d'un fichier d'initialisation au format texte, il faut le modifier manuellement pour que la nouvelle valeur soit prise en compte au prochain démarrage. Le fichier d'initialisation persistant est géré directement par Oracle. Il est situé en C:\oracle\product\10.1.0\db_1\database, mais c'est un fichier « caché ». Bien qu'il soit lisible, aucune intervention manuelle n'est permise pour éviter de le corrompre.

Les paramètres d'initialisation dynamiques et ceux qui ne le sont pas sont consultables par la vue V\$PARAMETER.

La zone SGA (System Global Area)

La SGA (System Global Area) représente la zone mémoire déterminante d'une instance, tant par sa taille que par son rôle. C'est elle qui assure le partage des données entre les utilisateurs.

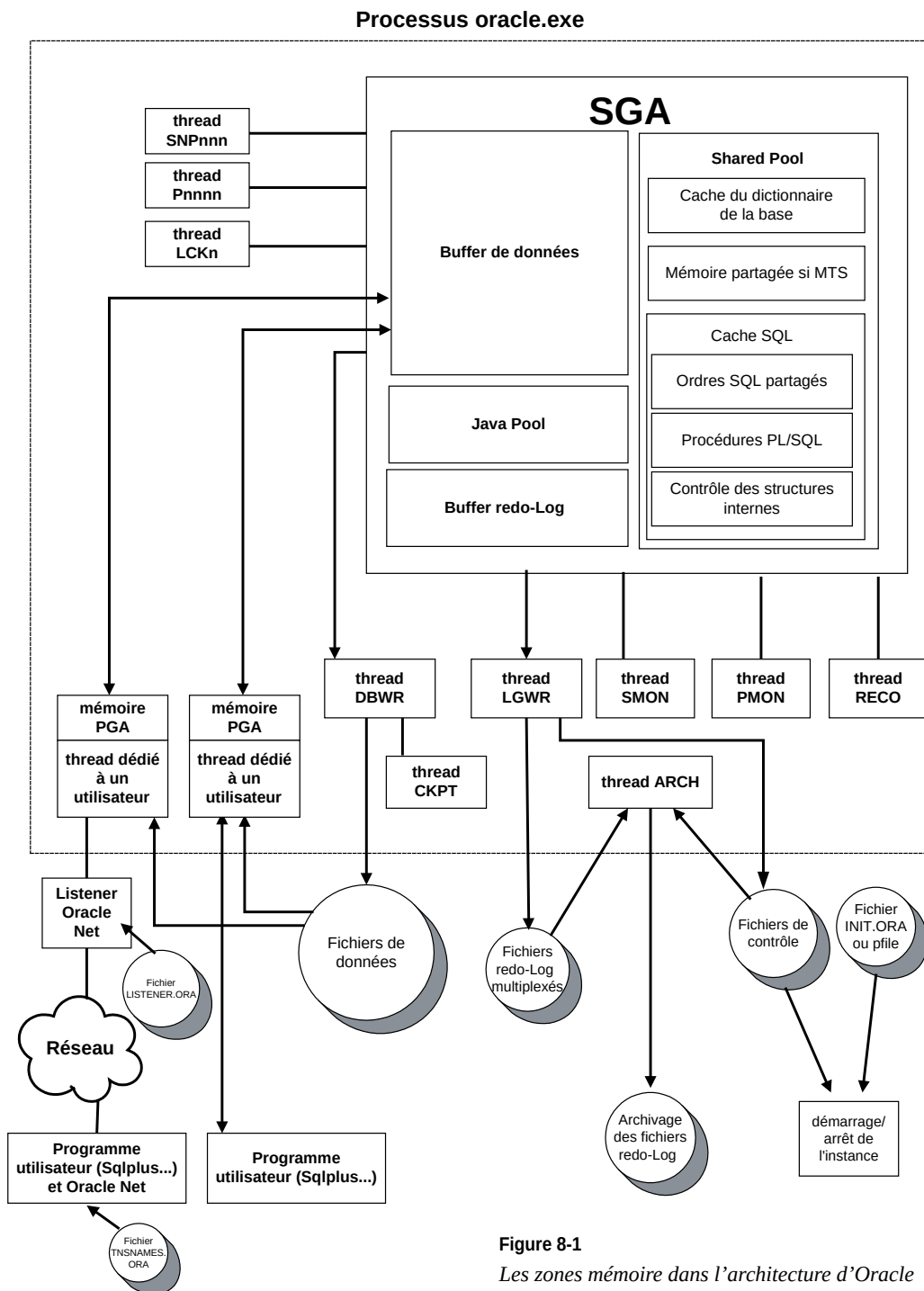


Figure 8-1

Les zones mémoire dans l'architecture d'Oracle

Toute donnée lue ou modifiée transite par la SGA. Même si la totalité du fonctionnement propre à la SGA n'est pas entièrement divulguée par Oracle, il est important d'en comprendre les fonctions majeures, pour faire face à des problèmes de performance ou à un événement inattendu.

Le contenu de la SGA est très structuré. Pour mieux en cerner les différentes zones, nous allons les présenter une par une :

- le cache des blocs de données (*buffer cache*) ;
- les buffers redo-log ;
- la Shared Pool ;
- la Java Pool ;
- une zone de communication inter-processus ;
- des espaces partagés si l'option Multi-Thread est activée.

Le cache des blocs de données

Cette zone comporte toutes les données en provenance de la base ou destinées à y être écrites (données lues par un SELECT ou modifiées par un UPDATE, INSERT, DELETE).

Cette zone mémoire est d'une taille fixe, bien inférieure à la dimension de votre base de données. Il est nécessaire que des mécanismes libèrent de l'espace pour permettre à de nouveaux blocs de « monter » en mémoire. C'est le rôle du processus DBWR et du processus utilisateur.

Le processus DBWR libère de l'espace en écrivant sur disque les blocs de données modifiés. Dans le cas où seuls des blocs de données non modifiés sont présents en mémoire, DBWR ne peut agir. C'est alors qu'intervient le processus utilisateur, créé automatiquement pour chaque connexion à une base Oracle 10g. Ce processus utilisateur obéit à un algorithme qui remplace les blocs non modifiés et inutilisés par des blocs provenant d'une lecture disque. Il lui incombe également de traiter les ordres SQL envoyés par le programme utilisateur et de lui retourner les données.

Comme Oracle 10g n'échange pas des informations mais des blocs d'informations (d'une taille fixe, choisie lors de la création de la base), d'autres données présentes dans ce bloc accompagnent celle à laquelle on a accédé ou qu'on a modifié. Il s'agit là d'un facteur d'amélioration des performances, car ces informations complémentaires seront peut-être utilisées et, dans ce cas, il ne sera pas nécessaire de provoquer un accès disque pour les atteindre.

Nous vous conseillons de consulter le chapitre 28, *Optimisation et performances*, avant de modifier les paramètres d'initialisation d'une instance.

Pour augmenter ou diminuer la taille du cache mémoire, il faut modifier le paramètre d'initialisation DB_CACHE_SIZE. Ce paramètre est dynamique. La valeur affectée est adaptée pour être un multiple du DB_BLOCK_SIZE.

Vous pouvez également utiliser l'ancien paramètre DB_BLOCK_BUFFER qui présente l'inconvénient de ne pas être dynamique. Il s'exprime en nombre de blocs mémoire et non en taille mémoire.

Pour visualiser la taille de la SGA, vous pouvez lancer sous le compte administrateur SYSTEM la commande SQL :

```
show sga
ou
select * from v$sga;
```

NAME	VALUE
Fixed Size	279600 bytes
Variable Size	167772160 bytes
Database Buffers	67108864 bytes
Redo Buffers	532480 bytes

Total System Global Area	235693104 bytes

Ici, le cache de données (*Database Buffers*) est d'environ 65 Mo pour une SGA de quelques 220 Mo. Cet ordre SQL permet de récupérer la taille des autres zones de la SGA, comme la taille allouée à l'environnement Java ou aux buffers redo-log. Ces valeurs peuvent être modifiées dynamiquement ou dans le fichier d'initialisation.

En interrogeant le dictionnaire de données d'Oracle, d'autres ordres SQL permettent d'entrer plus précisément dans le détail des différentes zones mémoire.

La SGA est un élément clé des performances d'Oracle. Il faut absolument éviter qu'elle « sorte » de la mémoire vive et qu'elle soit paginée. Le paramètre d'initialisation LOCK_SGA permet « d'accrocher » la SGA en mémoire vive.

Les paramètres de base et les paramètres avancés

Oracle 10g apporte une simplification de son paramétrage. Il existe maintenant deux types de paramètres :

- les paramètres de base ;
- les paramètres avancés.

Oracle indique que la plupart des bases de données nécessitent uniquement l'utilisation des paramètres de base pour fonctionner correctement. Ces paramètres sont :

```
CLUSTER_DATABASE
COMPATIBLE
CONTROL_FILES
DB_BLOCK_SIZE
DB_CREATE_FILE_DEST
DB_CREATE_ONLINE_LOG_DEST_n
DB_DOMAIN
DB_NAME
```

```
DB_RECOVERY_FILE_DEST
DB_RECOVERY_FILE_DEST_SIZE
DB_UNIQUE_NAME
INSTANCE_NUMBER
JOB_QUEUE_PROCESSES
LOG_ARCHIVE_DEST_n
LOG_ARCHIVE_DEST_STATE_n
NLS_LANGAGE
NLS_TERRITORY
PGA_AGGREGATE_TARGET
PROCESSES
REMOTE_LISTENER
REMOTE_LOGIN_PASSWORDFILE
ROLLBACK_SEGMENTS
SESSIONS
SGA_TARGET
SHARED_SERVERS
START_TRANSFORMATION_ENABLED
UNDO_MANAGEMENT
UNDO_TABLESPACE
```

Consultez l'index de cet ouvrage pour identifier la page où sont abordés ces différents paramètres. Pour les autres, reportez-vous à la documentation Oracle.

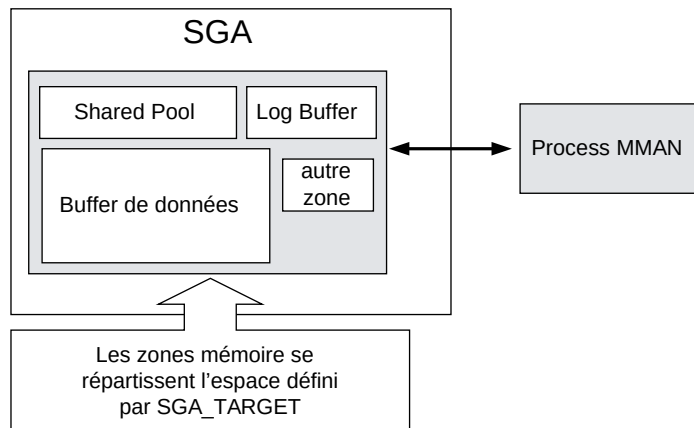
Le tuning automatique de la SGA

Oracle 10g apporte une nouveauté : l'Automatic Shared Memory Management (ASMM) et son processus Memory Manager (MMAN). Il laisse Oracle répartir automatiquement un espace mémoire disponible, le **sga_target** entre le buffer de données, la Shared Pool, la Large Pool et la Java Pool. C'est une fonction très utile pour des bases dont l'activité variable peut charger temporairement l'une de ces zones mémoire.

Ce paramètre est situé dans le fichier init.ora. Sa valeur par défaut est **sga_target = 0** qui inhibe le fonctionnement de l'ASMM.

Figure 8-2

Le principe d'ASMM



Pour utiliser ASMM, il suffit :

- de donner une taille mémoire à `sga_target` (autre que 0). Vous pouvez vous baser sur le résultat de l'ordre SQL présenté dans le paragraphe précédent ;
- de positionner le paramètre `statistics_level` à `TYPICAL` ou `ALL` ;
- de positionner les valeurs définissant les buffers de données, la Shared Pool, la Java Pool, etc., à 0. Si une valeur autre que 0 est conservée, ASMM l'interprétera comme une taille minimale de mémoire à conserver pour cette zone.

Le paramètre `sga_target` est dynamique, c'est-à-dire qu'il peut être changé base ouverte, en fonctionnement. Veuillez toutefois à ne pas lui allouer un espace mémoire dont vous ne disposez pas !

Les buffers redo-log

Dès qu'une modification intervient sur vos données (INSERT, UPDATE, DELETE), elle se répercute sur les blocs de données en mémoire, qui sont simultanément copiés dans les buffers redo-log.

Dès qu'un COMMIT ou un ROLLBACK est utilisé, le processus LGWR écrit les données (non encore transférées) dans les fichiers redo-log. L'utilisateur n'a pas à attendre l'écriture des éléments nouveaux dans les fichiers de données (ce qui peut être long). Ce travail est effectué en temps masqué par le processus DBWR. Aucune information ne sera perdue, quoi qu'il arrive, car toutes sont écrites dans les fichiers redo-log.

Les buffers redo-log sont conçus pour assurer des écritures très rapides dans les fichiers redo-log. Comme pour tous les autres échanges, ce sont des blocs de données qui sont écrits dans ces fichiers, ce processus étant le plus performant.

Le paramètre dynamique LOG_BUFFER dimensionne cette zone mémoire. Il s'exprime en octets et il peut être modifié dynamiquement.

La Shared Pool

Nous avons vu comment les buffers de données peuvent améliorer l'accès aux données. Néanmoins, une base de données ne comporte pas que les informations de l'utilisateur, elle inclut également toutes celles nécessaires à son fonctionnement. Là aussi, des mécanismes ont été construits pour fournir d'excellentes performances. Pour cela, les zones mémoire utilisées sont :

- le partage des ordres SQL et PL/SQL : Shared SQL Area ;
- le dictionnaire de données ;
- les curseurs ;
- le Multi-Thread.

Le paramètre dynamique `SHARED_POOL_SIZE` dimensionne cette zone mémoire. Il s'exprime en octets et il peut être modifié dynamiquement.

Le langage SQL utilisé pour dialoguer avec Oracle 10g doit être traduit et interprété par la base pour vous apporter une réponse. À cette fin, une analyse syntaxique (la syntaxe est-elle correcte ?) et sémantique (tous les éléments sont-ils compris ?) est mise en œuvre. Oracle 10g vérifie aussi que vous disposez des droits d'accès aux données demandées. Enfin, Oracle 10g consulte ses statistiques (en tenant compte du volume de vos tables et des index) pour optimiser le chemin d'accès à vos données. Cette opération correspond au *parsing* de votre ordre SQL. Tout ce travail est rapide, mais quand il s'agit d'applications qui répètent très souvent le même ordre SQL ou des ordres proches, certains éléments sont conservés et partagés en vue d'une réutilisation.

Ce mécanisme est particulièrement efficace dans le traitement des programmes PL/SQL. Le code d'exécution de vos procédures, fonctions, triggers et packages est conservé, prêt à être réemployé.

Lors de l'analyse des ordres SQL, Oracle 10g vérifie le nom des tables, des colonnes, les droits d'accès, etc. Tous ces éléments sont gérés dans le dictionnaire de la base de données, propriété de l'utilisateur SYS. Une zone mémoire est allouée au stockage d'informations provenant du dictionnaire de données.

Les curseurs sont utilisés dans les programmes PL/SQL et dans des opérations internes réalisées par Oracle. Tout comme pour les ordres SQL et les programmes PL/SQL, un espace mémoire est réservé en Shared Pool pour les curseurs.

La Java Pool

Oracle 10g propose un environnement Java intégré à la base. Il nécessite de la mémoire pour fonctionner. L'installation du moteur Java dans la base Oracle est facultative. Ce point est abordé au chapitre 13, *Création d'une base Oracle 10g*.

La mémoire allouée en SGA pour le fonctionnement du moteur Java est précisée par la variable `JAVA_POOL_SIZE`. Cette valeur est fixée à 50 Mo minimum lors de l'installation du catalogue Java, mais elle peut être augmentée ou diminuée par la suite.

La zone PGA (Program Global Area)

Le rôle de la SGA est de permettre un partage de données entre tous les utilisateurs. Il existe néanmoins d'autres informations qui n'ont pas besoin d'être partagées. C'est le cas des informations contenues dans la zone PGA (*Program Global Area*, parfois appelée *Process Global Area*). Cette zone mémoire est allouée pour le fonctionnement de chaque thread utilisateur.

Une connexion à une base Oracle 10g engendre donc la création d'un thread utilisateur et de sa mémoire associée, la PGA.

La PGA est toujours située sur le serveur qui héberge votre base de données. Elle stocke des informations concernant les variables utilisées, la session utilisateur et l'état des transactions en cours. Elle contient également la zone mémoire dans laquelle s'effectue le tri de vos données.

Lorsque de nombreux utilisateurs accèdent à cette base, la multiplication des processus utilisateur et de chacune de leur PGA conduit à une surcharge du système. C'est à ce moment que l'utilisation de l'option MTS (*Multi-Thread Server*) peut économiser des ressources en transférant une partie des opérations effectuées dans la PGA vers la SGA. C'est le cas de la zone de tri mémoire de vos ordres SQL. L'inconvénient majeur de cette technique est la forte augmentation de la SGA. Il convient donc de trouver un équilibre entre une « petite » SGA et des PGA « consommatrices », et une SGA « imposante » et des PGA « plus petites ».

Le choix d'une configuration MTS se justifie à partir de cent ou cent cinquante utilisateurs simultanés.

La taille de la PGA est déterminée par des paramètres d'initialisation de votre instance : la taille mémoire prévue pour le tri des données, le nombre de « database links » autorisés, le nombre de fichiers composant votre base de données et le nombre de fichiers redo-log. En ce qui concerne l'espace de tri des données, il faut bien réaliser l'importance du paramètre d'initialisation SORT_AREA_SIZE qui influe sur la taille de toutes les PGA créées.

Depuis Oracle 10g, la gestion de la taille de la PGA peut être automatique : PGA_AGGREGATE_TARGET donne la mémoire maximale que vous allouez pour l'ensemble des PGA utilisateurs. Cette zone ne réside pas en SGA. La valeur WORKAREA_SIZE_POLICY peut prendre les valeurs *manual* ou *auto*.

L'administrateur d'une base Oracle 10g n'a pas à se soucier de l'existence et du fonctionnement de la PGA (c'est pourquoi elle est si peu connue !). Il convient d'y songer lorsque vous rencontrez des limites de mémoire sur votre système : vous pouvez alors augmenter sa capacité, diminuer certains paramètres ou envisager une configuration MTS.

Résumé

Ce chapitre a détaillé les différentes zones mémoire utilisées par Oracle 10g. Elles sont partagées ou privées et vous permettent d'accéder le plus rapidement possible à vos données. La compréhension de leur fonctionnement pour les dimensionner est très importante : c'est l'un des facteurs clés de l'amélioration des performances de vos bases Oracle 10g.

9

Les fichiers d'une base Oracle 10g

Dans ce chapitre :

- les fichiers qui composent une base Oracle 10g ;
- les fichiers de données ;
- les fichiers redo-log ;
- les fichiers de contrôle ;
- les fichiers d'initialisation et de configuration ;
- l'architecture OFA proposée par Oracle.

Si la mémoire est largement utilisée pour accélérer le fonctionnement de la base, les fichiers inscrits sur des disques durs permettent le stockage persistant des données. Oracle a réservé à certains fichiers le rôle de conserver les informations, d'autres sont destinés uniquement au fonctionnement interne de la base ou à son paramétrage.

Dans ce chapitre, nous ne nous limitons pas aux fichiers qui composent la base de données et qui doivent être sauvegardés, mais nous étudions aussi les fichiers de paramétrage, trace, log, etc. Ils sont méconnus et pourtant d'une grande utilité pour rechercher l'origine de problèmes.

Nous décrirons l'OFA (*Optimal Flexible Architecture*), qui propose une méthode d'organisation des fichiers Oracle sur votre système. L'approche OFA, rigoureuse et structurée, simplifie énormément le travail de l'administrateur Oracle 10g.

Les fichiers Oracle 10g et leur rôle

Installer Oracle sur un système signifie y placer un nombre impressionnant de fichiers. C'est pourquoi, en amont de toute installation, il faut pouvoir identifier les différentes catégories de fichiers. Du point de vue d'un administrateur Oracle, on peut classer ces fichiers comme suit :

- les logiciels Oracle ;
- les fichiers de données ;
- les fichiers redo-log ;
- les fichiers redo-log archivés ;
- les fichiers de contrôle ;
- le fichier d'initialisation ;
- les fichiers d'initialisation d'Oracle Net ;
- le fichier d'alerte ;
- le fichier trace des utilisateurs.

Les logiciels Oracle

Ce sont tous les fichiers dont l'installation est nécessaire au fonctionnement d'une base de données. Leur nombre est impressionnant, mais vous n'avez pas à vous en soucier. Tout comme les fichiers qui composent le système Windows, ils sont installés suivant les options choisies. Seul l'ajout d'un nouveau logiciel ou la modification d'une option d'un composant Oracle vous obligerait à intervenir sur ces fichiers.

En jargon Oracle, l'ensemble de ces fichiers s'appelle une *distribution*. Tous les logiciels d'une même distribution sont placés dans une même Oracle Home. Ce point est détaillé plus loin dans ce chapitre.

Plusieurs distributions d'Oracle peuvent coexister sur la même machine, par exemple, une version 9.2 en production et une version 10.1 en cours de validation. Les règles OFA, abordées dans ce chapitre, vous aident à organiser vos distributions, les bases de données et les fichiers d'administration.

Les fichiers de données

Une fois votre distribution Oracle installée, une ou plusieurs bases de données sont créées. Leur dimension dépend de la quantité d'informations à stocker. Pour répondre à ces besoins, le nombre, la taille et l'emplacement des fichiers de données seront adaptés.

Les fichiers de données sont les plus volumineux de votre base. Si celle-ci est démarrée, il suffit d'interroger son dictionnaire interne pour connaître les caractéristiques de l'ensemble des fichiers qui la composent. Si elle est arrêtée, mieux vaut avoir suivi des

règles d'installation rigoureuses, comme les règles OFA, pour savoir où se trouvent ces fichiers. Comme le volume des informations est susceptible d'évoluer, il est fréquent qu'un administrateur Oracle intervienne sur ces fichiers.

Oracle évolue : historiquement, l'emplacement, le nom et parfois la taille des fichiers doivent être précisés par le DBA. À partir d'Oracle9, vous aviez la possibilité d'indiquer juste les répertoires où placer les fichiers, Oracle se chargeait de les nommer, les créer et les gérer. Oracle 10g apporte la notion d'ASM (*Automatic Storage Management*). Nous franchissons une nouvelle étape : c'est un substitut au système de fichiers du système d'exploitation (Windows, Linux). Vous créez un « système de fichiers » ASM et Oracle s'occupe de tout ! ASM est présenté au chapitre 2, *Configurer les disques pour Oracle*. Comme ASM dépasse les objectifs de ce livre, seule la gestion « classique » des fichiers sera abordée.

Les fichiers de données contiennent toutes les informations de votre base dans un format spécifique à Oracle. Il n'est pas possible d'en visualiser le contenu avec un éditeur de texte. Même un éditeur binaire montre un format interne très complexe : mieux vaut renoncer !

Le seul et unique moyen pour accéder et manipuler des données stockées dans Oracle est d'utiliser le langage SQL. Vous ne pourrez jamais y accéder en manipulant les fichiers.

Pour ajouter de nouveaux fichiers de données, il faut utiliser des ordres SQL. Pour déplacer ou supprimer ces fichiers, il faut associer des ordres SQL (par exemple préciser à la base le déplacement) et des commandes Windows (pour le transfert physique du fichier). Une option SQL permet maintenant d'associer la suppression physique d'un fichier à la commande SQL. Assurez-vous de disposer d'une bonne sauvegarde avant d'utiliser cette option !

Lorsqu'un fichier de la base est plein, il peut automatiquement augmenter en taille. Le DBA précise un incrément et fixe une dimension maximale. Cette limite supérieure évite de nombreux désagréments, comme la saturation de votre espace disque, la création de fichiers d'une taille supérieure à la celle permise par l'espace disque disponible ou dépassant les capacités de votre outil de sauvegarde.

Les threads utilisateurs lisent les données dans les fichiers de données et le thread Data-base Writer (DBWR) les écrit dans ces fichiers.

Les fichiers de données contiennent deux types d'informations : d'une part, celles du dictionnaire de données et de travail, d'autre part celles des utilisateurs.

Les données des utilisateurs et le dictionnaire constituent le cœur de la base de données Oracle. Ces éléments sont tous composés de tables et d'index. Les informations de travail sont utilisées dans le fonctionnement interne d'Oracle. Elles permettent de gérer la notion de transaction, de trier et de manipuler de gros volumes de données.

Tous les fichiers de données ont un format spécifique à Oracle : le *bloc Oracle* est défini lors de la création de la base et il est difficile de le modifier. Tous les échanges entre les disques, la mémoire et les threads se feront par des multiples de ce bloc. Entre une base

de données conçue pour effectuer de très nombreuses lectures/écritures « petites » et un infocentre accédant de manière linéaire à de très gros volumes de données, on perçoit l'importance du bloc oracle sur les performances.

Un fichier de données appartient à un unique tablespace, mais un tablespace peut posséder plusieurs fichiers. La notion de tablespace est abordée plus loin dans ce chapitre.

Les fichiers redo-log

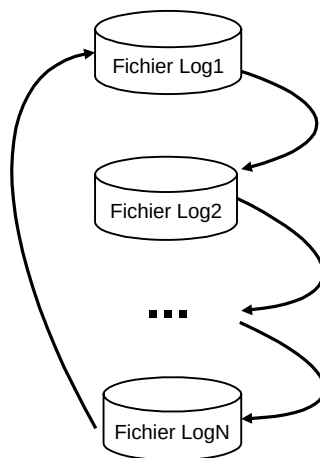
Les fichiers redo-log sont des fichiers de journalisation utilisés pour le fonctionnement interne d'Oracle, ils enregistrent et conservent toutes les modifications successives de votre base de données. Ils sont utiles lors d'une restauration à la suite d'un problème d'instance ou de disque. Cette restauration consiste à rejouer le contenu des fichiers redo-log dans la base.

Le thread LGWR (*Log Writer*) est chargé de l'écriture dans les fichiers redo-log. Il y enregistre les valeurs avant et après modification, ainsi que des renseignements internes. À l'instar des fichiers de données, il n'est pas possible d'accéder au contenu des fichiers redo-log avec un éditeur de texte. Un nouvel outil livré avec Oracle 10g, le Log Miner, permet d'analyser le contenu des fichiers log. Il s'appuie sur des packages PL/SQL (DBMS_LOGMNR et DBMS_LOGMNR_D) et des vues V\$ de la base.

Lorsque le fichier redo-log courant est saturé, Oracle poursuit sur le suivant et ainsi de suite jusqu'au dernier. Quand celui-ci est plein, Oracle réutilise le premier, puis le second, etc. L'utilisation des fichiers est donc circulaire.

Figure 9-1

*Chaînage circulaire
des fichiers redo-log*



Avant la réutilisation d'un fichier redo-log, Oracle le sauvegarde à un autre emplacement si la base fonctionne en mode archivage (ARCHIVELOG MODE). Dans les autres cas, Oracle le réemploie en supprimant son contenu sans sauvegarde préalable. Entre-temps, les données présentes en mémoire et copiées dans les fichiers redo-log à écraser ont été

écrites dans les fichiers de données. Le mode d'archivage est optionnel : c'est l'administrateur Oracle qui décide de sa mise en service ou non.

Pour des raisons de sécurité, ces fichiers peuvent être multiplexés (c'est-à-dire dupliqués) dans des groupes (*Groups*). Si chaque groupe est situé sur un disque différent, cela évite une perte d'informations en cas de panne d'un disque.

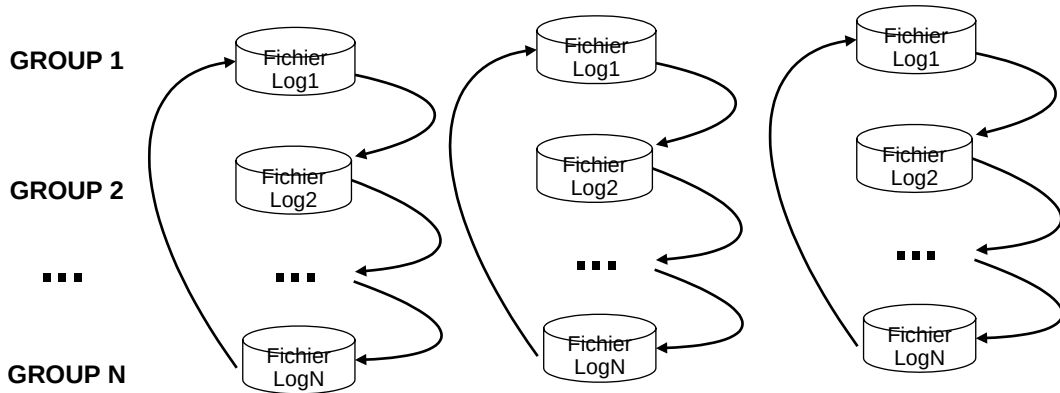


Figure 9-2

Fichiers redo-log groupés

Les fichiers redo-log doivent se trouver sur les disques les plus rapides, car chaque modification s'y trouve inscrite. Les disques rapides étant souvent les moins sécurisés, la fiabilité est alors assurée par le multiplexage des fichiers redo-log placés sur des disques différents.

Des vues du dictionnaire interne de votre base permettent de localiser vos fichiers redo-log. Un administrateur Oracle intervient sur ces fichiers uniquement pour les déplacer ou en modifier la taille afin, sous certaines conditions, d'améliorer la performance de la base. Oracle Enterprise Manager dispose d'une interface graphique pour réaliser ces actions.

Les fichiers redo-log archivés

Lorsqu'une base est en mode ARCHIVELOG, les fichiers redo-log sont archivés à mesure de leur remplissage. Vous devez apporter la plus grande attention à cette opération. Si le répertoire de destination des archives est saturé, la base de données se bloque et un message s'inscrit dans le fichier d'alerte (voir plus loin). Oracle considère qu'il vaut mieux bloquer les utilisateurs que débrayer un mécanisme de sécurité majeur. L'emplacement et le nettoyage régulier des anciens fichiers archivés représentent les principales préoccupations de l'administrateur d'une base en mode ARCHIVELOG.

En cas de restauration, l'opération consiste à réaliser une sauvegarde complète de la base de données puis à redérouler les fichiers redo-log archivés les uns à la suite des autres.

Les fichiers de contrôle

Les fichiers de contrôle sont créés en même temps que la base de données. Ils sont principalement utilisés à chaque démarrage de celle-ci, puis mis à jour automatiquement par Oracle. Pour des raisons de sécurité, on peut créer plusieurs fichiers de contrôle, mais ils sont tous identiques.

Le fichier d'initialisation utilisé pour lancer l'instance situe l'emplacement des fichiers de contrôle. Chacun d'eux précise la localisation de tous les autres fichiers (données et redo-log) qui composent la base.

Les fichiers de contrôle indiquent si la base de données a été correctement fermée et si une restauration est nécessaire. Il est impossible de les visualiser pour en exploiter le contenu.

Des vues du dictionnaire interne de la base permettent de les localiser. Si vous les perdez, il est possible de les créer à nouveau, à partir d'une base Oracle en fonctionnement. Si celle-ci est arrêtée, il faudra repartir d'une sauvegarde. Sauf à les déplacer et à les sauvegarder, un administrateur Oracle n'a pas à s'en préoccuper.

Le fichier d'initialisation

Toute instance Oracle nécessite des paramètres d'initialisation utilisés à chaque démarrage. Ils peuvent être conservés sous forme d'un fichier texte ou enregistrés dans un fichier d'initialisation persistant.

Dans le premier cas, on parlera du fichier `initSID.ora` et du fichier persistant `spfileSID.ora` dans le second. Le SID correspond à l'identifiant de l'instance qu'il doit démarrer.

Le fichier d'initialisation persistant et de nouveaux paramètres d'initialisation dynamiques pouvant être modifiés « base en marche » sont des nouveautés apportées par Oracle 10g. Leur utilisation conjointe est très souple et nous vous engageons fortement à les utiliser. Ces points sont décrits au chapitre 10, *Démarrer et arrêter une base Oracle 10g*.

Les fichiers d'initialisation persistants `spfileSID.ora` sont placés par défaut en `ORACLE_HOME\database`, soit `C:\oracle\product\10.1.0\db_1\database\pfileSID.ora`. L'explorateur Windows doit être paramétré car ce sont des fichiers cachés.

Le fichier d'initialisation détermine entre autre la taille mémoire allouée à la SGA (System Global Area), zone mémoire commune à tous les utilisateurs de votre instance. Son rôle est déterminant dans l'optimisation des ressources d'une machine.

Pour modifier votre fichier d'initialisation `initSID.ora`, utilisez un éditeur de texte « propre », tel que le notepad. N'utilisez surtout pas un traitement de texte qui incorpore ses propres caractères de contrôle, comme Word ou Wordpad. Ne modifiez jamais le fichier d'initialisation persistant `spfileSID.ora` à l'aide d'un éditeur. Il risquerait d'être corrompu et inutilisable.

Lorsque vous installez Oracle 10g et qu'une base de données est créée par défaut, un fichier exemple documenté est fourni. Pour chaque paramètre important, des exemples de valeurs de configuration petite, moyenne ou importante sont fournies.

Il existe de nombreux paramètres, mais peu sont réellement utiles. Si vous ne modifiez pas la valeur de l'un d'entre eux, une valeur par défaut lui est affectée. Celle-ci dépend de votre système d'exploitation. Pour connaître la liste des paramètres du fichier d'initialisation, utilisez sous SQL*Plus la commande suivante :

```
connect system/mot_de_passe
show parameters

# vous pouvez aussi accéder à la vue V$PARAMETER
select name, value from V$PARAMETER
/
```

La vue V\$PARAMETER vous permet aussi de retrouver les paramètres ayant une valeur différente de celle par défaut. Ce renseignement figure également dans le fichier d'alerte de votre base de données (voir plus loin).

Voici une liste des paramètres les plus courants rencontrés dans le fichier d'initialisation d'une instance Oracle. Vous les retrouverez dans pratiquement toutes les bases Oracle.

Paramètres d'initialisation sans impact sur les performances

- **instance_name** : nom de l'instance qui doit correspondre à la variable d'environnement ORACLE_SID. Souvent nommé SID, il est codé sur 8 caractères maximum et ne peut plus être changé une fois la base créée ;
- **db_name** : nom interne de la base de données. Nous vous conseillons très fortement de toujours attribuer à la base de données et au SID (qui identifie l'instance) le même nom (4 lettres majuscules). Ce nom est codé à l'intérieur des fichiers de contrôle ;
- **db_domain** : chaîne alphanumérique identifiant le domaine réseau auquel appartient la base. Le nom global de la base de données sera la concaténation du db_name avec le db_domain. Par exemple, db_name=TEST, db_domain=gilles.fr, global_dbname = TEST.gilles.fr. Ce paramètre est très important si la base doit communiquer avec d'autres bases Oracle par réplication ou autre ;
- **control_files** : noms et emplacements des fichiers de contrôle de la base. Ils sont tous identiques et doivent être disponibles à chaque démarrage. Si l'un d'eux est perdu (destruction, défaillance disque...), il faut le supprimer de la liste des fichiers pour démarrer l'instance ;
- **open_cursors** : nombre maximum de curseurs pouvant être ouverts simultanément par une session utilisateur ;
- **db-files** : nombre maximum de fichiers de la base de données ;
- **remote_login_passwordfile** : indique le mode de gestion souhaité pour permettre à des utilisateurs distants d'arrêter/démarrer la base ;

- `max_dump_file_size` : limite la taille du fichier dump (erreur) lorsqu'un problème grave est détecté dans Oracle. Évite de saturer l'espace disque réservé aux fichiers dump ;
- `background_dump_dest` : précise l'emplacement des fichiers d'alerte et de log associés à l'instance ;
- `core_dump_dest` : précise l'emplacement de création des fichiers résultant d'erreurs internes Oracle ou « core » ;
- `user_dump_dest` : localise l'emplacement des fichiers d'alerte et de log associés à des problèmes concernant les threads des utilisateurs ;

Paramètres d'initialisation importants pour les performances

- `compatibles` : assure un mode de compatibilité entre différentes versions de base Oracle. Ainsi, vous pouvez demander à ce qu'une base Oracle9.2 se comporte comme la version 7.x ;
- `db_block_size` : détermine la taille d'un bloc Oracle utilisé par le tablespace SYSTEM pour son format interne et les échanges entre les fichiers et la mémoire. Utilisé lors de l'étape initiale de création de la base, ce paramètre influe énormément sur les performances. Nouveauté apportée par Oracle 10g, certains tablespaces peuvent avoir un `db_block_size` différent de celui du tablespace SYSTEM ;
- `sga_target` : indique la taille totale de tous les composants de la SGA. Si `sga_target` est précisée, alors les buffers de données (`db_cache_size`), la zone réservée pour Java (`java_pool_size`), la Large Pool (`large_pool_size`) et la mémoire partagée (`shared_pool_size`) seront dimensionnés automatiquement en fonction de l'activité de la base ;
- `db_cache_size` : (anciennement `db_block_buffers`) taille affectée au buffer de données en mémoire. C'est l'un des paramètres les plus visibles sur la taille de la SGA en mémoire. Comme tous les paramètres qui gèrent la taille de la SGA, il est modifiable dynamiquement ;
- `db_block_buffers` : par souci de compatibilité, on peut toujours utiliser ce paramètre mais il n'est pas dynamique alors que `db_cache_size` l'est ;
- `shared_pool_size` : taille en octets de la mémoire partagée affectée à l'instance. Ce paramètre dynamique influe sur la taille de la SGA ;
- `large_pool_size` : taille en octets de la mémoire partagée affectée à l'instance. Ce paramètre dynamique influe sur la taille de la SGA ;
- `java_pool_size` : taille en octets de la mémoire partagée affectée au fonctionnement des options Java incorporées dans la base ;
- `pga_aggregate_target` : taille cible de la mémoire PGA (tris, fusions...) disponible pour l'ensemble des processus attachés à l'instance ;
- `sort_area_size` : mémoire PGA affectée à un seul processus. Maintenu pour compatibilité ascendante. Oracle recommande d'utiliser `pga_aggregate_target` ;

- `processes` : nombre maximal de threads associés à une instance Oracle. Permet de limiter le nombre de connexions à une instance, car chacune crée un thread dédié. Pour éviter un blocage, fixer une valeur supérieure au nombre maximum de connexions envisagées ;
- `sessions` : valeur déduite automatiquement par Oracle depuis la valeur `processes` ;
- `undo_management` : détermine si l'instance est lancée en gestion automatique des annulations gérées dans un tablespace spécifique ou dans le mode manuel utilisant des rollback segments. Le premier mode correspond à la valeur `AUTO` et le second à `MANUAL`. Oracle recommande d'utiliser la gestion automatique ;
- `undo_tablespace` : indique le nom du tablespace utilisé par la base pour gérer automatiquement les segments d'annulation ou *undo* ;
- `rollback_segment` : mode de gestion des segments d'annulation conservé pour compatibilité avec les versions antérieures à Oracle 10g. Oracle recommande d'utiliser les tablespaces de type `UNDO` ;
- `db_file_multiblock_read_count` : fixe le nombre de blocs de données écrits simultanément ;
- `log_buffer` : nombre de blocs (d'une taille unitaire de `db_block_size` octets) alloués aux buffers redo-log en mémoire ;
- `log_checkpoint_interval` : durée en secondes entre deux points de contrôle qui forcent l'écriture dans les fichiers redo-log ;
- `audit_trail` : `TRUE` ou `FALSE`, si vous souhaitez utiliser les capacités d'audit d'une base ;
- `log_archive_start` : détermine si le thread destiné à archiver les fichiers redo-log doit être démarré. Attention, pour activer l'archivage, il faut aussi mettre la base en mode `ARCHIVELOG` ;
- `timed_statistics` : collecte les statistiques avec `sql_trace` ;
- `sql_trace` : autorise l'analyse de toutes les sessions sur une base Oracle (très consommateur de CPU et d'espace disque). Les statistiques sont activées par `timed_statistics`.

Un administrateur Oracle intervient sur ce fichier lors de la création puis du tuning de la base. De plus en plus de paramètres sont modifiables base en marche par la commande `ALTER SYSTEM`.

Les fichiers d'initialisation d'Oracle Net

Les fichiers `tnsnames.ora` et `listener.ora` sont les principaux fichiers de paramétrage d'Oracle Net. Ils sont détaillés au chapitre 15, *Oracle Net, le middleware Oracle*. Si vous créez une nouvelle base, il faut modifier le fichier `tnsnames.ora` pour en tenir compte. Ne les oubliez pas dans vos sauvegardes, car leur paramétrage est parfois délicat à mettre au point.

Le fichier d'alerte de la base

Ce fichier d'alerte est ignoré par la majorité des administrateurs. Pourtant, son exploitation est précieuse car, en cas de problèmes, Oracle y laisse des messages explicites. Lorsqu'une difficulté survient, pensez-y !

Chaque base Oracle possède son fichier d'alerte. Il ajoute des informations durant la vie d'une base. Vous pouvez archiver régulièrement son contenu, puis le supprimer. Pendant le démarrage de la base, si le fichier d'alerte n'existe pas, Oracle en crée un pour y écrire des informations de type « DBWR process started ». L'exemple suivant fournit le contenu d'un fichier d'alerte obtenu après le lancement d'une base sous WIndows.

```
Starting up ORACLE RDBMS Version: 10.1.0.2.0.
System parameters with non-default values:
  processes                = 150
  shared_pool_size         = 83886080
  large_pool_size          = 8388608
  java_pool_size           = 50331648
  nls_language              = FRENCH
  nls_territory             = FRANCE
  control_files             = C:\ORACLE\PRODUCT\10.1.0\ORADATA\PROD\CONTROL01.CTL,
                           ↳C:\ORACLE\PRODUCT\10.1.0\ORADATA\PROD\CONTROL02.CTL,
                           ↳C:\ORACLE\PRODUCT\10.1.0\ORADATA\PROD\CONTROL03.CTL

  db_block_size             = 8192
  db_cache_size            = 25165824
  compatible               = 10.1.0.2.0
  db_file_multiblock_read_count= 16
  db_recovery_file_dest     = C:\oracle\product\10.1.0\flash_recovery_area
  db_recovery_file_dest_size= 2147483648
  undo_management           = AUTO
  undo_tablespace           = UNDOTBS1
  remote_login_passwordfile= EXCLUSIVE
  db_domain                 = gilles.fr
  dispatchers               = (PROTOCOL=TCP) (SERVICE=PRODXDB)
  job_queue_processes       = 10
  background_dump_dest      = C:\ORACLE\PRODUCT\10.1.0\ADMIN\PROD\BDUMP
  user_dump_dest            = C:\ORACLE\PRODUCT\10.1.0\ADMIN\PROD\UDUMP
  core_dump_dest            = C:\ORACLE\PRODUCT\10.1.0\ADMIN\PROD\CDUMP
  sort_area_size            = 65536
  db_name                   = PROD
  open_cursors              = 300
  pga_aggregate_target      = 25165824
CKPT started with pid=6, OS id=2572
LGWR started with pid=5, OS id=2536
PMON started with pid=2, OS id=2388
DBW0 started with pid=4, OS id=2500
RECO started with pid=8, OS id=2644
SMON started with pid=7, OS id=2608
MMAN started with pid=3, OS id=2464
Fri May 06 10:04:46 2005
starting up 1 dispatcher(s) for network address
↳ '(ADDRESS=(PARTIAL=YES)(PROTOCOL=TCP))'...
```

```
CJQ0 started with pid=9, OS id=2680
Fri May 06 10:04:46 2005
starting up 1 shared server(s) ...
Fri May 06 10:04:53 2005
alter database mount exclusive
Fri May 06 10:04:53 2005
Controlfile identified with block size 16384
Fri May 06 10:04:57 2005
Setting recovery target incarnation to 1
Fri May 06 10:04:58 2005
Successful mount of redo thread 1, with mount id 4264330181
Fri May 06 10:04:58 2005
Database mounted in Exclusive Mode.
Completed: alter database mount exclusive
Fri May 06 10:04:58 2005
alter database open
```

L'exploitation des renseignements contenus dans le fichier d'alerte dépend de votre expérience Oracle. Beaucoup de ces messages ne sont que des informations sans lien avec une éventuelle erreur.

Le fichier trace de chaque base est placé dans le répertoire indiqué par la variable `background_dump_dest` du fichier d'initialisation *initSID.ora*.

Lorsque la base de données démarre, tous les paramètres du fichier *initSID.ora* qui ne correspondent pas aux valeurs par défaut sont écrits. Sont également indiqués : le lancement des différents threads et la restauration automatique de la base si elle n'a pas été fermée proprement. En général, le fichier d'alerte contient une trace de tous les démarrages et arrêts, de la création de tablespaces et de segment d'annulation, de quelques ordres de modification (ALTER) ainsi que des informations concernant la permutation des fichiers redo-log et des erreurs rencontrées.

Chaque renseignement est daté et un message précise souvent le lancement d'une action et un autre sa fin.

Pour un DBA Oracle, il est très important de consulter régulièrement le contenu du fichier trace de chaque base. C'est le « carnet de santé » de celle-ci. Les messages que vous y trouverez vous aideront à corriger les erreurs.

L'observateur d'événements

De nombreuses informations importantes concernant la vie d'une base Oracle sont inscrites dans le *journal d'application* de l'*observateur d'événements* : démarrage, arrêt, connexion d'un administrateur, actions importantes réalisées sur la base, etc.

On y retrouve nombre d'informations présentes dans le fichier d'alerte, mais il moins facile à exploiter en cas de problème.

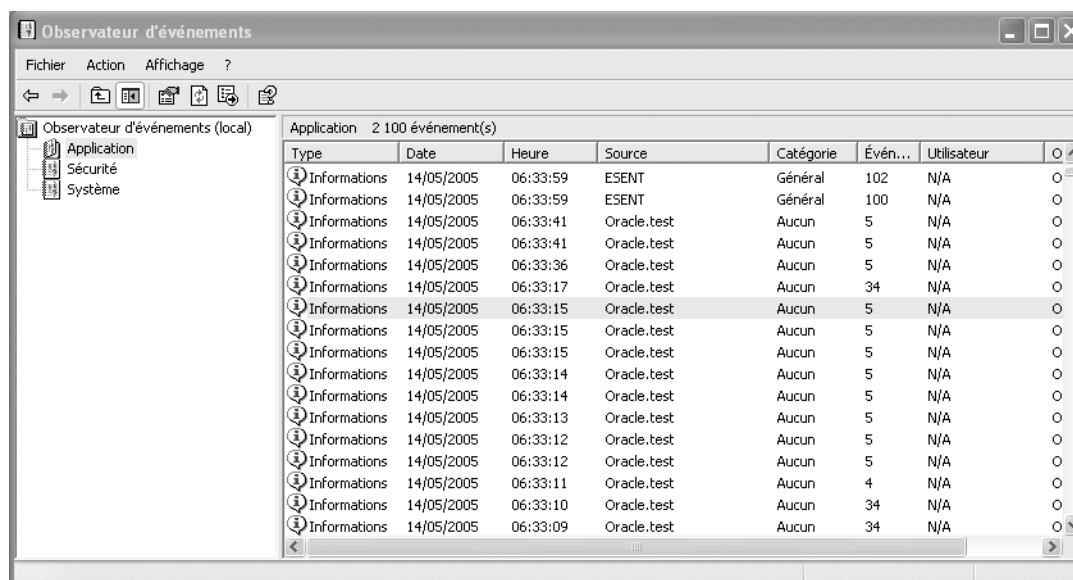


Figure 9-3

Observateur d'événements

Les fichiers trace d'Oracle Net

Oracle Net, le middleware réseau Oracle utilisé en client-serveur, dispose lui aussi de fichiers trace. Côté serveur, le listener enregistre tous les démarrages, arrêts, connexions réussies ou ayant échoué dans un fichier « log ». Ce fichier, qui enregistre les connexions (et non leur contenu) est indispensable pour contrôler d'éventuelles tentatives de piratage de la base.

Côté poste client (client Windows, Linux ou autre...), il est possible de forcer la création d'un fichier trace pour enregistrer tout ce que votre application confie à Oracle Net dans une communication client-serveur. Ces points sont traités au chapitre 15, *Oracle Net, le middleware Oracle*.

Les fichiers trace des utilisateurs

Il est possible de demander à Oracle de réaliser un fichier trace pour chaque connexion utilisateur à la base. L'utilisation de ces fichiers trace doit être exceptionnelle, car elle engendre sur le serveur une surconsommation de place disque et de temps processeur.

Le fichier trace des utilisateurs est placé dans le répertoire indiqué par la variable `user_dump_dest` du fichier d'initialisation `initSID.ora`. La trace est déclenchée/arrêtée par la commande `ALTER SESSION`.

Faire coexister plusieurs versions Oracle

Il est possible de faire coexister plusieurs versions différentes d'Oracle sur un même serveur. Toutes ces versions peuvent fonctionner simultanément. Vous pouvez ainsi avoir une base en version 9.2 de production tout en testant l'installation de la version 10.0.1.

Pour cela, des variables d'environnement ainsi que des valeurs de la base de registre sont utilisées. Elles sont présentées dans ce paragraphe.

Qu'est-ce qu'un Oracle Home ?

Pour chaque installation, Oracle Universal Installer vous demande de saisir un nom qui décrira un nouvel Oracle Home. Sous ce nom, Oracle regroupe des variables qui définissent l'environnement de fonctionnement d'une base. Ainsi, un Oracle Home détermine :

- le répertoire où seront installés les logiciels Oracle. Cette valeur correspond à la variable ORACLE_HOME, par exemple C:\orant\ora92 ou C:\oracle\product\10.1.0\db_1 ;
- la variable Windows PATH qui précise où sont les fichiers exécutables d'Oracle ;
- des valeurs de la base de Registre ;
- des services.

L'Oracle Home joue le rôle d'aiguillage qui permet de faire pointer un ensemble de variables vers un environnement ou vers un autre. Il est impératif d'installer chaque version d'Oracle dans un Oracle Home unique. Cela facilite la maintenance des différentes versions qui peuvent coexister sur un serveur.

Il est conseillé d'utiliser un nom d'Oracle Home qui soit en relation avec le logiciel à installer. Par exemple, OraHome92 si vous installez Oracle version 9.2 ou OraDb10g_home1 si vous installez la version 10g.

Quel est l'impact d'un Oracle Home sur le serveur ?

Oracle utilise l'Oracle Home pour personnaliser son installation. Par exemple, dans le menu Démarrer > Programmes, le menu « Oracle - *HomeName* » sera créé avec HomeName correspondant au nom d'Oracle Home.

On trouvera ainsi les menus « Oracle – OraHome92 » et « Oracle – OraDb10g_home1 » si les noms donnés aux Oracle Home sont respectivement OraHome92 et OraDb10g_home1.

On retrouve aussi le nom d'Oracle Home dans l'identifiant de certains services. Par exemple, un listener installé dans un Oracle Home nommé OraDb10g_home1 donnera naissance au service nommé « OracleOraDb10g_home1TNSListener ».

Quel est l'impact d'un Oracle Home sur le Registre ?

Comme les paramètres d'une installation Oracle sont conservés dans la base de registre, l'Oracle Home permet de structurer l'organisation des clés de registre.

- HKEY_LOCAL_MACHINE\SOFTWARE\ORACLE\KEY_OraDb10g_home1 permet de numérotter les différentes installations d'Oracle. Elle contient les informations sur

toutes les variables d'environnement utilisées par tous les logiciels Oracle liés à cet Oracle Home. La terminaison « homeN » indiquera le numéro d'installation. Cette clé contient les informations :

- ORACLE_HOME_NAME : le nom donné à l'Oracle Home ;
- ORACLE_HOME : le répertoire pointant où est installé Oracle ;
- NLS_LANG : le langage utilisé lors de cette installation ;
- ORACLE_BUNDLE_NAME : la version Oracle installée ;
- ORACLE_SID_XXX : les caractéristiques de démarrage et d'arrêt des instances *SID* (TEST et PROD sur la figure). Ne figurent à ce niveau que les instances relatives à la version Oracle installée dans le ORACLE_HOME.

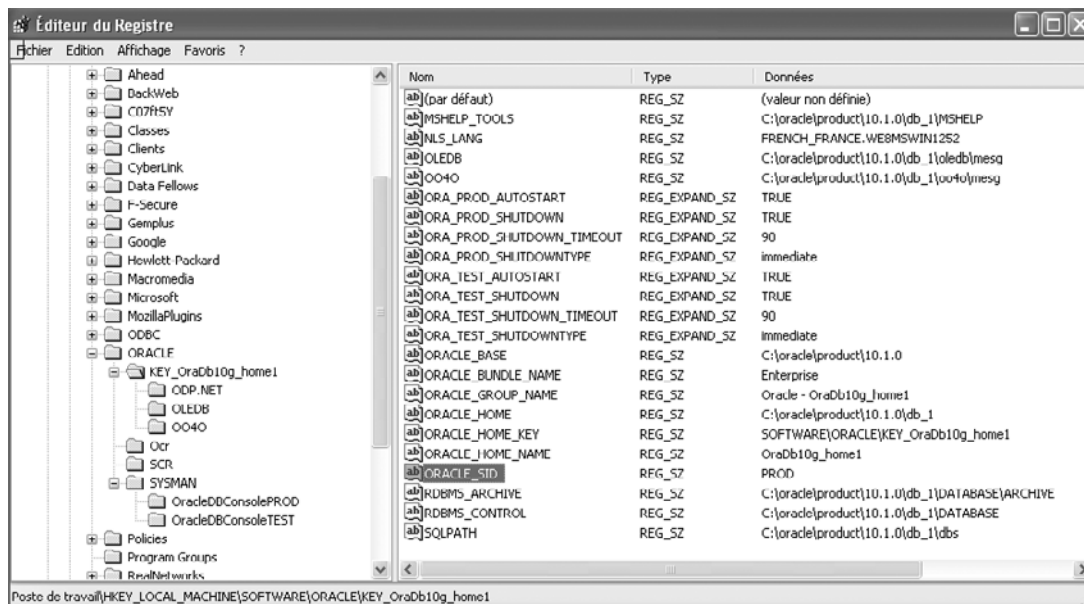


Figure 9-4

Clé de registre "ORACLE_HOMEX"

Dans la figure précédente vous pouvez voir, par exemple, le paramétrage concernant le démarrage et l'arrêt de la base TEST et comment la base de données doit se comporter lors d'un arrêt/démarrage du serveur Windows.

Les paramètres de démarrage des instances TEST et PROD figurent dans la clé de registre relative à KEY_Oradb10g_home1. Cela indique que cette base a été créée avec Oracle version 10. Si une autre version d'Oracle était installée sur ce serveur, les bases qui seraient créées avec cette version figureraient dans la clé correspondante en HKLM\SOFTWARE\ORACLE\ KEY_Oradb10g_home2.

Les valeurs de la base de registre peuvent être modifiées à l'aide de Regedit, mais aussi à l'aide de l'outil graphique *Oracle Administration Assistant for Windows*. On y retrouve ainsi les paramètres présents dans la base de registres.

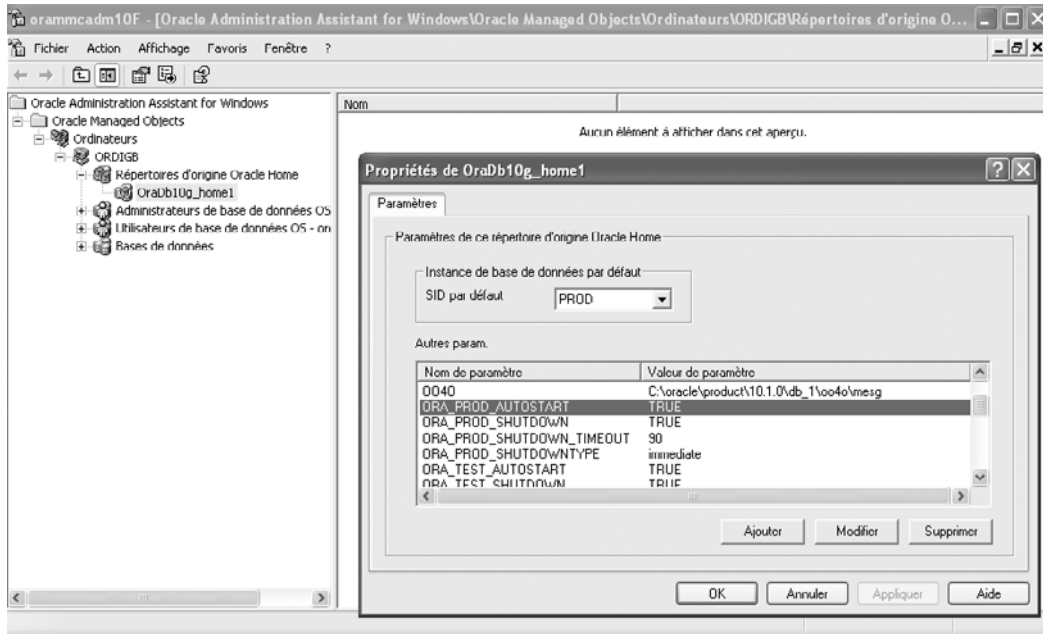


Figure 9-5

L'assistant Oracle pour Windows

L'architecture OFA proposée par Oracle

Les répertoires d'installation de Windows sont bien structurés. Qu'en est-il de ceux d'Oracle ? Pour répondre à cette question, Oracle recommande d'utiliser la structure OFA (*Optimal Flexible Architecture*). C'est un ensemble de règles d'installation et de configuration qui vous donneront des bases Oracle rapides, fiables, faciles à installer et nécessitant peu de maintenance.

Les règles OFA sont le fruit de plus de dix années d'expérience de consultants d'Oracle sous Windows et Unix. Nous vous conseillons fortement de les consulter pour les utiliser avant toute installation Oracle 10g.

Bénéfices procurés par OFA

OFA a été conçu pour :

- faciliter le travail des administrateurs Oracle en standardisant l'installation d'Oracle quel que soit le système d'exploitation utilisé : Windows, Unix, Linux... ;

- organiser un grand nombre de données et d'applications sur des disques en évitant les goulots d'étranglement nuisibles aux performances ;
- faciliter les tâches d'administration comme les sauvegardes des logiciels et des données. Ce sont des points essentiels ;
- installer plusieurs versions d'Oracle sur une même machine et les administrer facilement ;
- permettre le fonctionnement de plusieurs bases Oracle 10g sur une même machine, les gérer facilement ;
- suivre et modifier aisément l'augmentation de la taille des bases ;
- créer des espaces de travail pour les utilisateurs Oracle 10g.

Caractéristiques d'une installation OFA

Le système de fichiers est organisé pour vous offrir une administration simple :

- installer plusieurs versions d'Oracle sur un même serveur ;
- créer de nouvelles bases.

Les performances

Les entrées/sorties sont réparties sur un nombre suffisant de disques pour éviter les problèmes d'engorgement.

En structurant l'installation, les ressources en matériel (CPU, disque) sont minimisées pour un même niveau de performance.

Une panne matérielle affecte le minimum possible d'éléments.

La sécurité

Les sauvegardes sont facilitées, car l'emplacement de tous les composants sensibles est clairement identifié.

Dans chaque base de données, les fichiers qui la composent et ses fichiers d'administration ont tous des répertoires réservés. Ils comportent des noms significatifs qui permettent à un administrateur Windows de les identifier. L'association entre le tablespace et le fichier est contenue dans le nom du fichier.

Évolutivité

Plusieurs versions d'Oracle peuvent fonctionner simultanément, ce qui permet d'installer et de tester une nouvelle version alors que l'ancienne est en production. Le transfert de l'ancienne version vers la nouvelle est simple pour l'administrateur et transparent pour les utilisateurs.

Il est très facile d'ajouter des composants à votre machine ou de changer une base Oracle de machine, car une installation OFA est bien séparée du reste des fichiers Windows.

OFA permet aisément de répartir les fichiers de vos bases de données sur un ou plusieurs disques.

ORACLE_BASE et ORACLE_HOME

Lors de l'installation, Oracle Universal Installer sépare les logiciels nécessaires au fonctionnement d'Oracle des fichiers membres des bases de données.

ORACLE_BASE est le répertoire racine d'une installation Oracle. Si vous suivez les règles OFA, la valeur d'ORACLE_BASE est X:\oracle (par exemple C:\oracle\product\10.1.0).

Oracle Home correspond aux variables nécessaires pour faire fonctionner les **logiciels** Oracle. Ces variables d'environnement regroupent : la variable ORACLE_HOME qui indique l'emplacement où sont installés les logiciels (par exemple C:\oracle\product\10.1.0\db_1). Partant de ce répertoire, Oracle sait que ses exécutables sont situés sous le répertoire BIN.

Si vous suivez les règles OFA, ORACLE_HOME est situé sous ORACLE_BASE. Par exemple :

```
ORACLE_BASE = C:\oracle\product\10.1.0
ORACLE_HOME1 = C:\oracle\product\10.1.0\db_1
ORACLE_HOME_NAME = OraDb10g_home1
Les logiciels Oracle seront alors installés en ORACLE_HOME.
```

Si plusieurs versions Oracle sont installées sur le serveur, il y aura un ensemble de variables Oracle Home par version d'Oracle :

```
ORACLE_BASE = C:\oracle\product\10.1.0
ORACLE_HOME1 = C:\oracle\product\10.1.0\db_1
ORACLE_HOME2 = C:\oracle\product\10.1.0\db_2
Les logiciels Oracle version 10.1 seront installés en ORACLE_HOME1
Les logiciels Oracle version 10.2 seront installés en ORACLE_HOME2
```

Pour chaque version d'Oracle, les autres variables attachées à chaque ORACLE_HOME sont inscrites dans la base de registres.

Où placer les fichiers des bases de données ?

Nous avons vu dans ce chapitre que l'on peut interroger le dictionnaire de données Oracle pour connaître l'ensemble des fichiers qui compose la base. Ils peuvent être placés n'importe où sur votre système. Pour une meilleure organisation, OFA propose de ne pas mélanger les fichiers de vos bases de données avec ceux des logiciels Oracle. De même, il faut placer dans des répertoires différents les fichiers de bases séparées. Ce sont des erreurs répandues qui compliquent les sauvegardes et posent des problèmes lorsque vous souhaitez changer la version d'Oracle.

Pour bien séparer les fichiers de données des logiciels Oracle, les fichiers de données sont placés directement sous ORACLE_BASE\oradata.

Comme plusieurs bases de données peuvent fonctionner simultanément, OFA conseille de créer un répertoire dédié pour contenir les fichiers de chaque base. Par exemple, si les bases TEST et PROD fonctionnent sur un serveur, on trouvera les répertoires :

```
Les fichiers sont placés sous ORACLE_BASE\oradata
C:\oracle\product\10.1.0\oradata\TEST : fichiers de la base TEST
C:\oracle\product\10.1.0\oradata\PROD : fichiers de la base PROD
```

Comme les fichiers d'une base peuvent être répartis sur plusieurs disques,

→ on peut aussi trouver :

```
C:\oracle\product\10.1.0\oradata\TEST : fichiers de TEST sur C:\
D:\oracle\product\10.1.0\oradata\TEST : fichiers de TEST sur D:\
```

Séparer les segments d'utilisations différentes

Pour une meilleure organisation interne de la base de données, les règles OFA recommandent d'affecter un tablespace par type de segment à héberger :

- SYSTEM abritera le dictionnaire de données, procédures, fonctions, etc. ;
- SYSAUX, contiendra les tables, index de toutes les options Oracle (Java, Spatial, OEM...);
- TEMP accueillera les segments temporaires utilisés automatiquement par Oracle lors d'opérations de tri, de création d'index, etc. ;
- UNDOTBS réunira les segments d'annulation ou UNDO ;
- USERS rassemblera les segments (tables, index) de vos utilisateurs ;
- APPLI_A_DATA, APPLI_A_INDEX, APPLI_B... serviront à l'hébergement des données des applications partagées entre plusieurs utilisateurs.

Comment nommer les fichiers des bases de données ?

Pour identifier chaque type de fichiers d'une base, Oracle conseille de les repérer par des extensions spécifiques :

- .ctl pour les fichiers de contrôle ;
- .log pour les fichiers redo-log ;
- .dbf pour les fichiers de données (*Data Base Files*).

Comme les fichiers de contrôle, redo-log et de données peuvent être multiples, utilisez des noms significatifs pour les identifier :

Pour les fichiers de contrôle : controlnn.ctl. Par exemple :

```
Pour la base de SID=TEST
control01.ctl : premier fichier de contrôle
control02.ctl : second fichier de contrôle
control03.ctl : troisième fichier de contrôle
etc.
```

Pour les fichiers redo-log : *redonn.log*. Par exemple :

```
Pour la base de SID=TEST
redo01.log   : premier fichier redo-log
redo02.log   : second fichier redo-log
redo03.log   : troisième fichier redo-log
etc.
```

Pour les fichiers de données : *tablespacenn.dbf*. Par exemple :

```
Pour les fichiers du tablespace SYSTEM
sytem01.dbf  : premier fichier
sytem02.dbf  : second fichier
etc.

Pour les fichiers du tablespace DATA
data01.dbf   : premier fichier
data02.dbf   : second fichier
etc.

Pour les fichiers du tablespace APPLI_A
appli_a01.dbf : premier fichier
appli_a02.dbf : second fichier
etc.
```

Conserver tous ces fichiers dans cette arborescence rend très facile la sauvegarde de vos données.

Par exemple, le premier fichier du tablespace SYSTEM de la base PROD sera : C:\oracle\product\10.1.0\oradata\PROD\system01.dbf ; son identification est simplifiée.

Pour des questions d'optimisation ou de place disque, vous pouvez répartir les fichiers d'une même base de données sur plusieurs disques. Dans ce cas, seuls le disque change (par exemple C :, D :, E :), le reste de l'arborescence est identique.

Ainsi, le premier fichier de données du tablespace USER appartenant à la base PROD sera C:\oracle\product\10.1.0\oradata\PROD\user01.dbf. Le second fichier peut être placé sur un autre disque et se nomme D:\oracle\product\10.1.0\oradata\PROD\user02.dbf.

Où placer les fichiers d'administration des bases ?

Chaque base de données possède ses propres fichiers de configuration, de trace, de log. Oracle conseille que tous les fichiers de création puis à d'exploitation d'une base soient placés dans des répertoires séparés. OFA vous propose de créer un jeu de répertoires pour chacune, avec pour chemin d'accès ORACLE_BASE\admin. Par exemple, pour les deux bases de SID respectif TEST et PROD, leurs fichiers d'administration respectifs seront placés en :

```
Les fichiers sont placés sous ORACLE_BASE\admin\SID
C:\oracle\product\10.1.0\admin\TEST : contient les fichiers d'administration de TEST
C:\oracle\product\10.1.0\admin\PROD : contient les fichiers d'administration de PROD
```

Pour bien structurer les fichiers d'administration de chaque base, Oracle crée sous ORACLE_BASE\admin\SID les répertoires :

- *\bdump* : trace des threads de la base ;
- *\cdump* : fichier d'erreur du « noyau Oracle » ;
- *\create* : fichiers de création de la base ;
- *\exp* : fichiers d'export de la base (optionnel, à ajouter) ;
- *\pfile* : fichier d'initialisation de la base (*initSID.ora*) ;
- *\udump* : fichiers trace d'ordres SQL.

Par exemple, le répertoire qui hébergera le fichier d'alerte de la base PROD, alert-PROD.ora sera C:\oracle\product\10.1.0\admin\PROD\bdump.

Avantages d'OFA

L'organisation des fichiers proposée par OFA rend votre administration très simple. Considérez les exemples ci-dessous qui utilisent le caractère « * » pour manipuler les fichiers :

C :\	premier disque ;
D :\	second disque ;
\oracle	liste de tous les répertoires d'installation des différentes versions Oracle (quel que soit leur disque d'installation) ;
\oracle\admin	liste de toutes les bases de données Oracle dont les fichiers de configuration sont structurés suivant OFA (quel que soit leur disque d'installation) ;
**\oradata	liste de toutes les bases de données Oracle dont les fichiers de données sont structurés suivant OFA (quel que soit leur disque d'installation) ;
**\oradata\TEST*	liste de tous les fichiers de la base de données TEST (quel que soit leur disque d'installation) ;
**\oradata\TEST*.ctl	liste de tous les fichiers de contrôle de la base de données TEST (quel que soit leur disque d'installation).

OFA : schéma d'installation des fichiers Oracle sous Windows

Dans ce premier exemple, seule la version 10 est installée et une seule base de données est en fonction, la base TEST. Nous indiquons aussi l'emplacement des principaux fichiers présentés dans ce chapitre : fichiers d'initialisation, d'administration, etc.

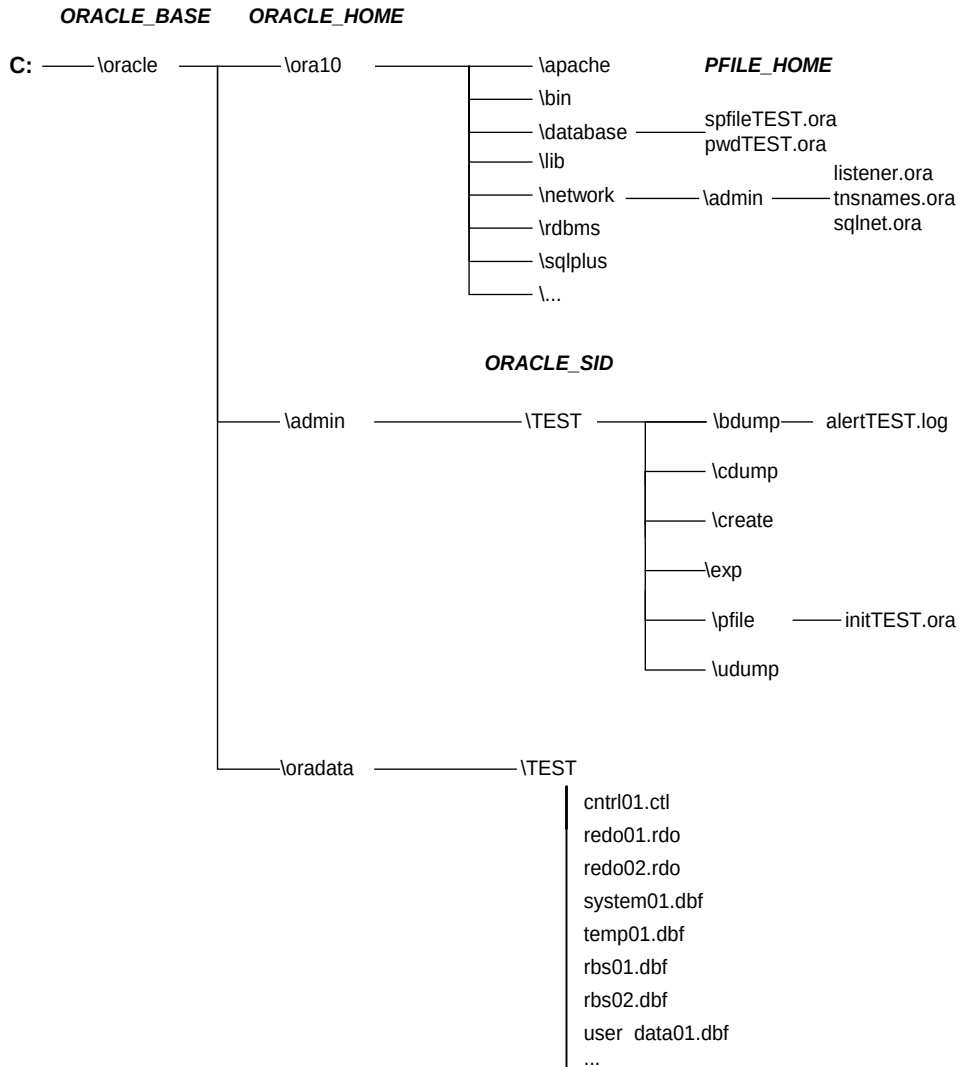


Figure 9-6

OFA : installation d'une base TEST sous Windows

Dans l'exemple suivant, seule la version 10 est installée et les fichiers de la base TEST sont répartis sur deux disques.

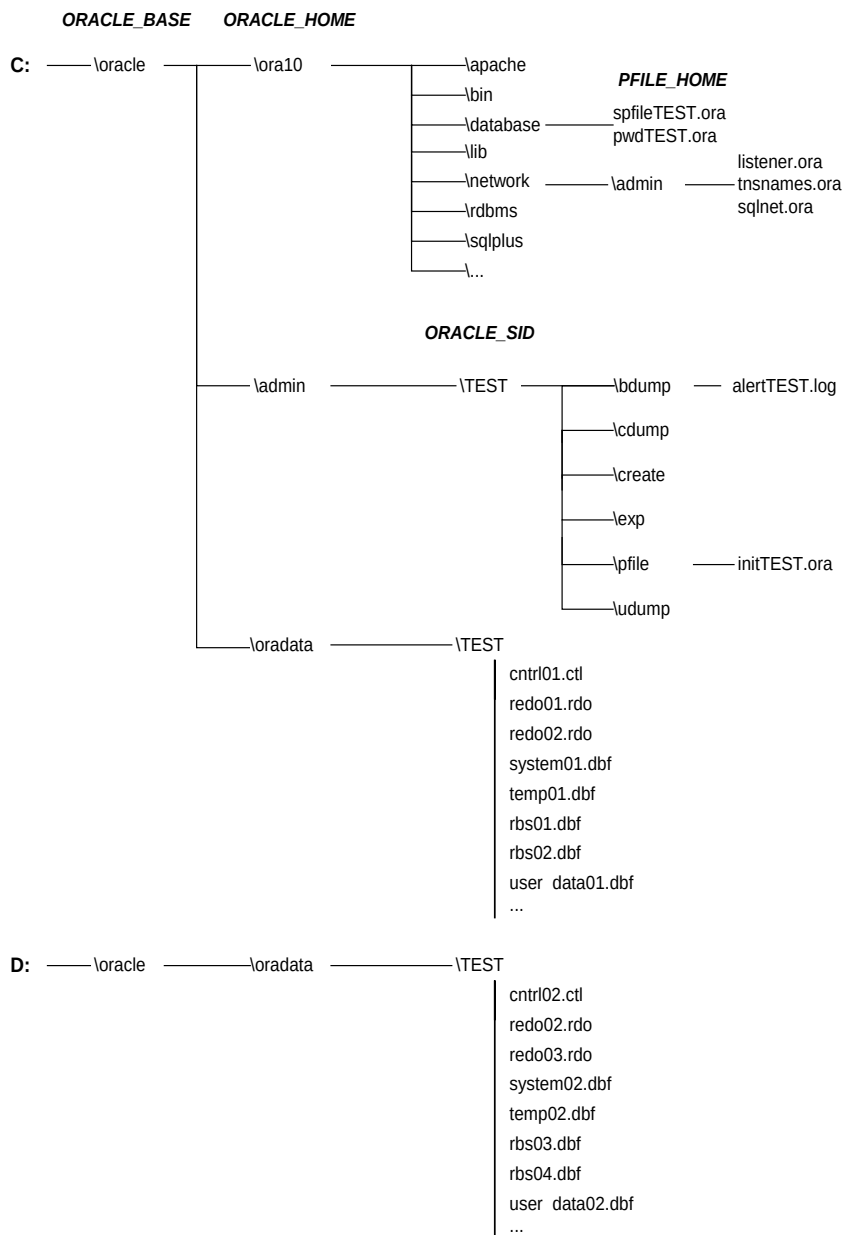


Figure 9-7

OFA : fichiers d'une base répartis sur deux disques

Dans l'exemple suivant, deux bases Oracle sont installées : TEST et PROD.

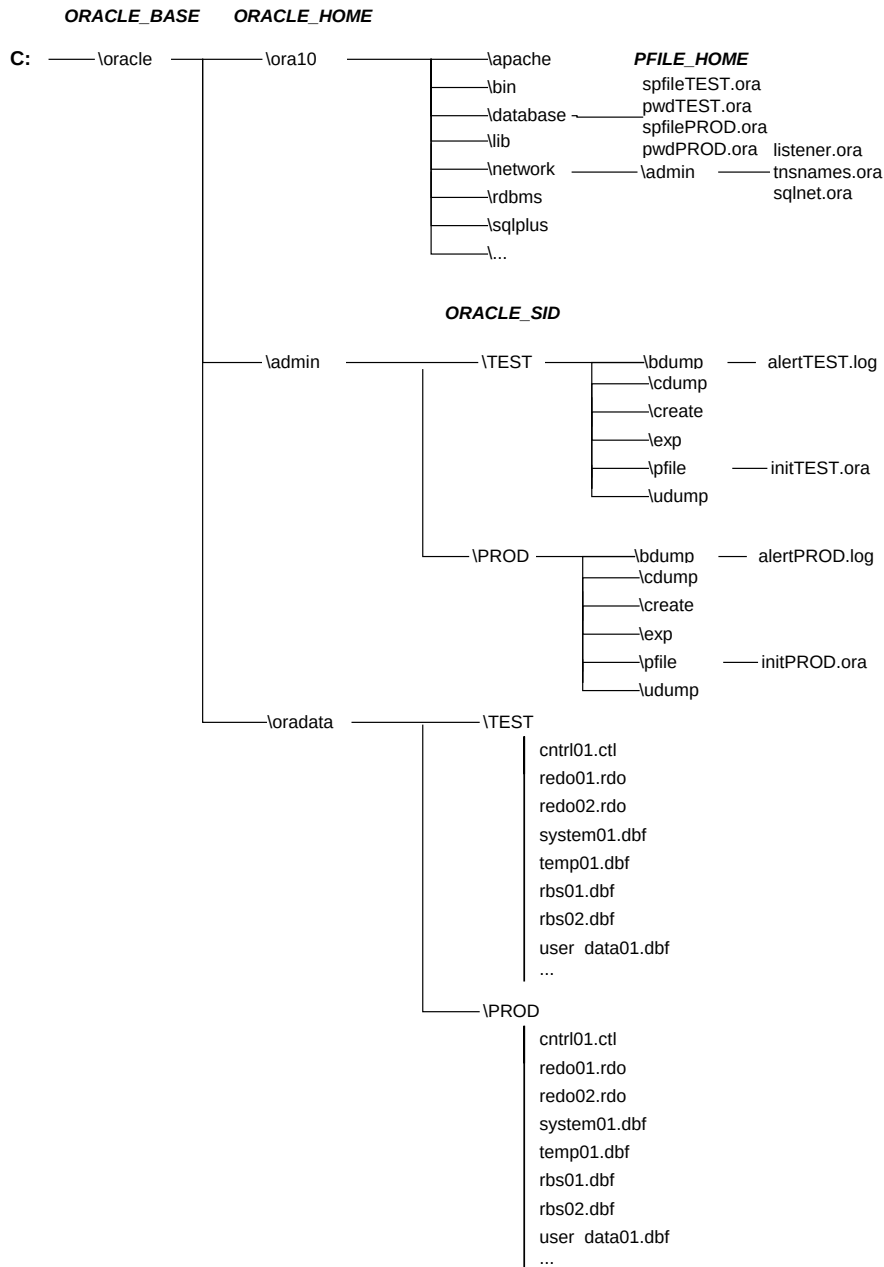


Figure 9-8

OFA : installation de deux bases Oracle sous Windows

Enfin, nous vous présentons comment coexistent deux versions différentes d'Oracle sur un même serveur. La base de production PROD utilisera la version 9.2 alors qu'une base de données TEST utilisera la version 10 avant de « basculer » la base de production de la version 9.2 vers la version 10.

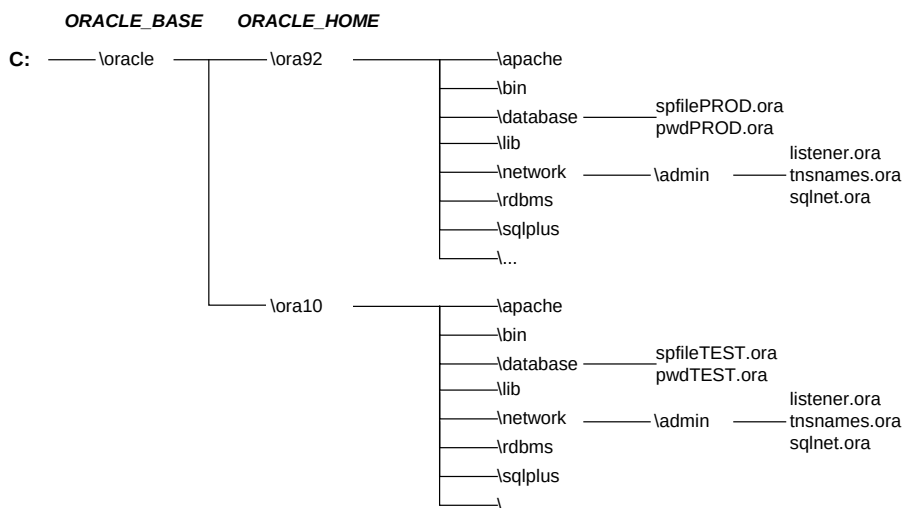


Figure 9-9

Coexistence de deux versions d'Oracle sur un serveur

Ces exemples (et d'autres) sont repris pour mémoire au chapitre 13, *L'installation d'Oracle 10g sous Windows*.

Ces exemples détaillés sont présentés pour insister sur l'importance des règles OFA. Elles sont simples, de bon sens et facilitent le travail des administrateurs Oracle et Windows. Utilisez-les !

Résumé

Ce chapitre a abordé l'utilisation des différents fichiers auxquels recourt Oracle 10g : ceux destinés à conserver les données et ceux réservés au fonctionnement interne de la base ou à son paramétrage.

Pour mieux comprendre l'environnement d'une base Oracle 10g, nous ne nous sommes pas limités aux fichiers qui composent la base de données et qui doivent être sauvegardés, mais nous nous sommes intéressés aux fichiers de paramétrage, trace, log, etc. Ils sont méconnus et pourtant d'une grande utilité pour retrouver l'origine des problèmes.

Nous avons décrit l'OFA (*Optimal Flexible Architecture*) qui propose une méthode d'organisation des fichiers Oracle sur votre système. L'approche OFA, rigoureuse et structurée, simplifie considérablement le travail de l'administrateur Oracle 10g.

10

Démarrer et arrêter une base Oracle 10g

Dans ce chapitre :

- principes de l'installation d'Oracle ;
- étapes d'une création de base Oracle 10g ;
- démarrage d'une base Oracle 10g ;
- arrêt d'une base Oracle 10g.

Ce chapitre aborde les différentes étapes de la vie d'une base Oracle 10g ; il présente tout d'abord les grandes lignes de son installation, puis les étapes en amont et en aval qui encadrent sa création. Ces points ne constituent que des généralités, ils vous fournissent le fil conducteur d'une installation Oracle 10g. Enfin, nous étudions dans le détail les étapes relatives à son démarrage et à son arrêt.

Avant toute installation du logiciel Oracle 10g ou toute création de bases, nous vous recommandons de consulter les règles OFA (*Optimum Flexible Architecture*) préconisées par Oracle et abordées au chapitre 9, *Les fichiers d'une base Oracle 10g*. Les chapitres 13, *L'installation d'Oracle 10g sous Windows*, et 14, *Création d'une base Oracle 10g*, s'intéressent dans le détail aux étapes de sa création.

Étapes d'installation d'Oracle 10g

Toute installation Oracle 10g doit être planifiée. On peut effectuer facilement et rapidement une installation, mais si les choix initiaux ne sont pas judicieux, leur correction risque de s'avérer longue et délicate. Ce point est particulièrement vrai sous Windows où l'installation s'effectue très simplement. Les paragraphes suivants décrivent succinctement les différentes étapes à suivre pour installer Oracle 10g et créer une ou plusieurs bases de données.

S'assurer des prérequis techniques

Les prérequis techniques sous Windows se limitent généralement à la vérification du Service Pack Windows requis par Oracle, de l'espace disque disponible et de la vérification de la mémoire installée.

À ce moment-là, il est important de concevoir l'arborescence des répertoires qui respectera l'architecture OFA. Cela vous permettra d'installer Oracle et de créer vos bases sans aucun souci ultérieur d'espace disque.

L'installation

L'installation s'effectue sous le compte administrateur de Windows, car Oracle a besoin d'accéder à la base de registres ainsi qu'aux services. Comme Oracle 10g dispose d'une « boîte à outils » très importante, vous devez choisir les options à installer.

Après l'installation

Comme pour la vérification des prérequis, seules quelques actions techniques comme l'intégration des bases Oracle dans vos sauvegardes sont nécessaires.

Étapes d'une création de base

Une fois le logiciel Oracle 10g mis en place, on peut créer une ou plusieurs bases au moyen d'outils Oracle 10g, ou « manuellement », par l'ordre `CREATE DATABASE`. Dans les deux cas, les logiciels graphiques qui permettent de créer une base ne font « qu'habiller » un ordre `CREATE DATABASE`.

Quelle que soit l'option choisie, certaines étapes sont à réaliser en amont de la création de la base, d'autres après.

Le rôle du service OracleServiceSID

Sous Windows, une base Oracle nécessite que son service OracleServiceSID (SID correspondant à l'identifiant de la base) soit créé et démarré avant la création ou le démarrage de la base. Ce point est détaillé au chapitre 5, *Oracle 10g sous Windows*.

Étapes préliminaires à la création d'une base

Avant toute création de base, il faut planifier son installation et déterminer ses caractéristiques, dont son nom. Certaines options ne sont plus modifiables une fois la base créée. Elles sont déterminées lors de la création dans le fichier d'initialisation initSID.ora.

D'autres options, comme la détermination des dimensions des fichiers ou le jeu de caractères utilisé pour construire la base, sont spécifiées dans l'ordre SQL de création de la base.

Étapes postérieures à la création d'une base

Une fois la base créée, lancez des scripts SQL pour construire le dictionnaire de la base de données. Ces scripts sont fournis par Oracle. Ils sont nombreux, car chacun est spécialisé et assure une option technique précise. N'installez que les options techniques nécessaires.

Ensuite, il faut créer les tablespaces dédiés aux applications et aux espaces de tri.

Vient ensuite tout ce qui a trait à vos applications : les utilisateurs, leurs droits, les schémas de données, les procédures stockées, les données, etc.

N'oubliez pas d'intégrer chaque nouvelle base de données dans vos procédures de sauvegarde.

Base de données et instance

Chaque *base de données* Oracle 10g « en marche » est associée à une *instance*. Tout d'abord, le service OracleServiceSID doit être démarré. Il lance l'exécutable oracle.exe, qui attend l'ordre de démarrage de l'instance. Lors du démarrage de la base de données, de nombreux threads sont créés et la mémoire SGA (*System Global Area*) est allouée au sein du processus. Cet ensemble SGA et threads est appelé une *instance* Oracle 10g.

Une instance est une base de données « en marche » associant mémoire et threads. On parlera de base de données pour l'ensemble des fichiers qui la composent.

Une base de données Oracle 10g se compose des éléments suivants :

- fichiers : base de données, redo-log, contrôle ;
- threads : pour faire fonctionner la base ;
- mémoire réservée : la SGA.

Un serveur Windows peut accueillir une ou plusieurs instances (bases de données fonctionnant simultanément). Chaque instance possède alors son propre jeu de fichiers, son processus oracle.exe, ses threads et sa SGA. Ces bases de données et instances auront des vies et des administrations indépendantes les unes des autres.

Nous aborderons les avantages et inconvénients à disposer d'une ou plusieurs bases sur un seul serveur au chapitre 14, *Création d'une base Oracle 10g*.

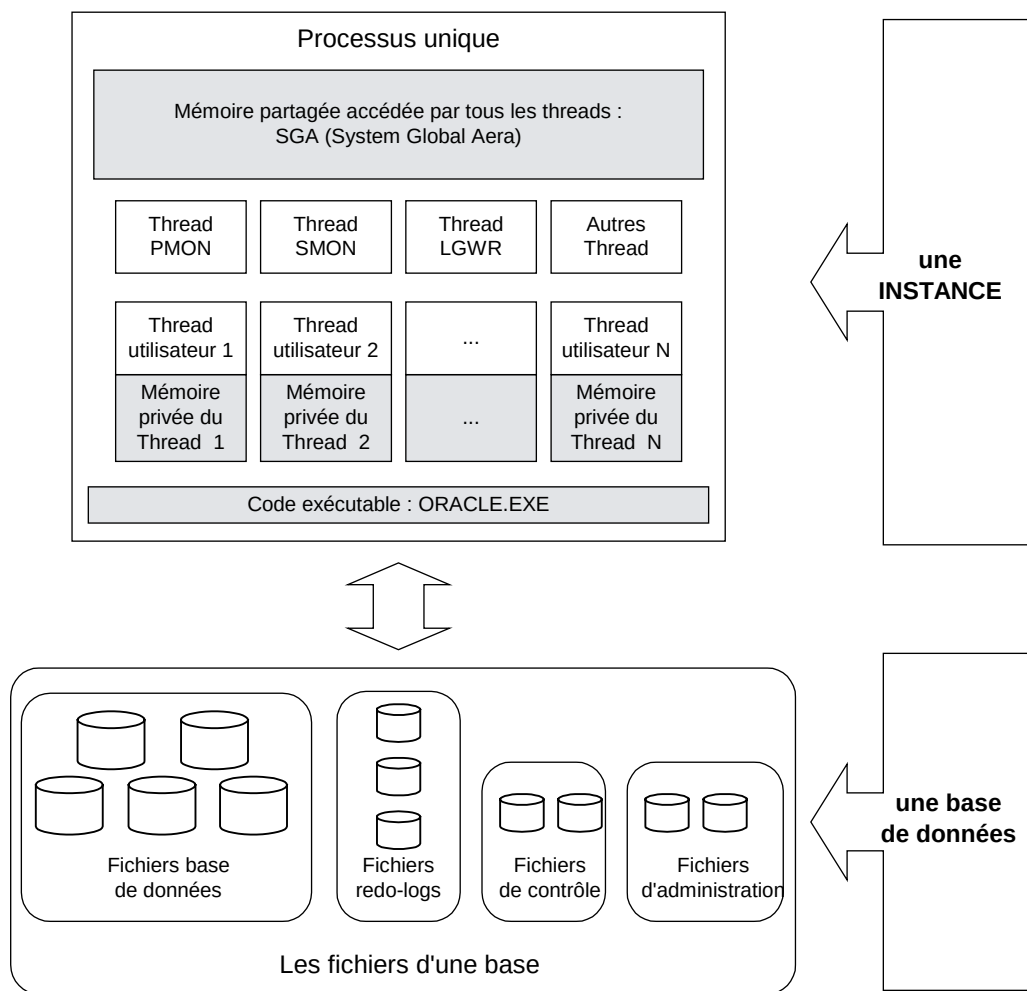


Figure 10-1

Instance et base de données

Les privilèges administrateur

Le démarrage, l'arrêt et la restauration d'une base sont des options très importantes. Par souci de sécurité, elles nécessitent que les utilisateurs qui les effectuent possèdent des privilèges spéciaux. Comme le démarrage s'effectue base fermée, il n'est pas possible de conserver ces privilèges dans la base ; c'est pourquoi ils reposent sur des droits ou des mots de passe vérifiés et stockés hors de la base de données.

Il existe deux méthodes pour attribuer ce privilège à un utilisateur Windows (elles sont décrites au chapitre 24, *Stratégie de sécurité sous Windows* :

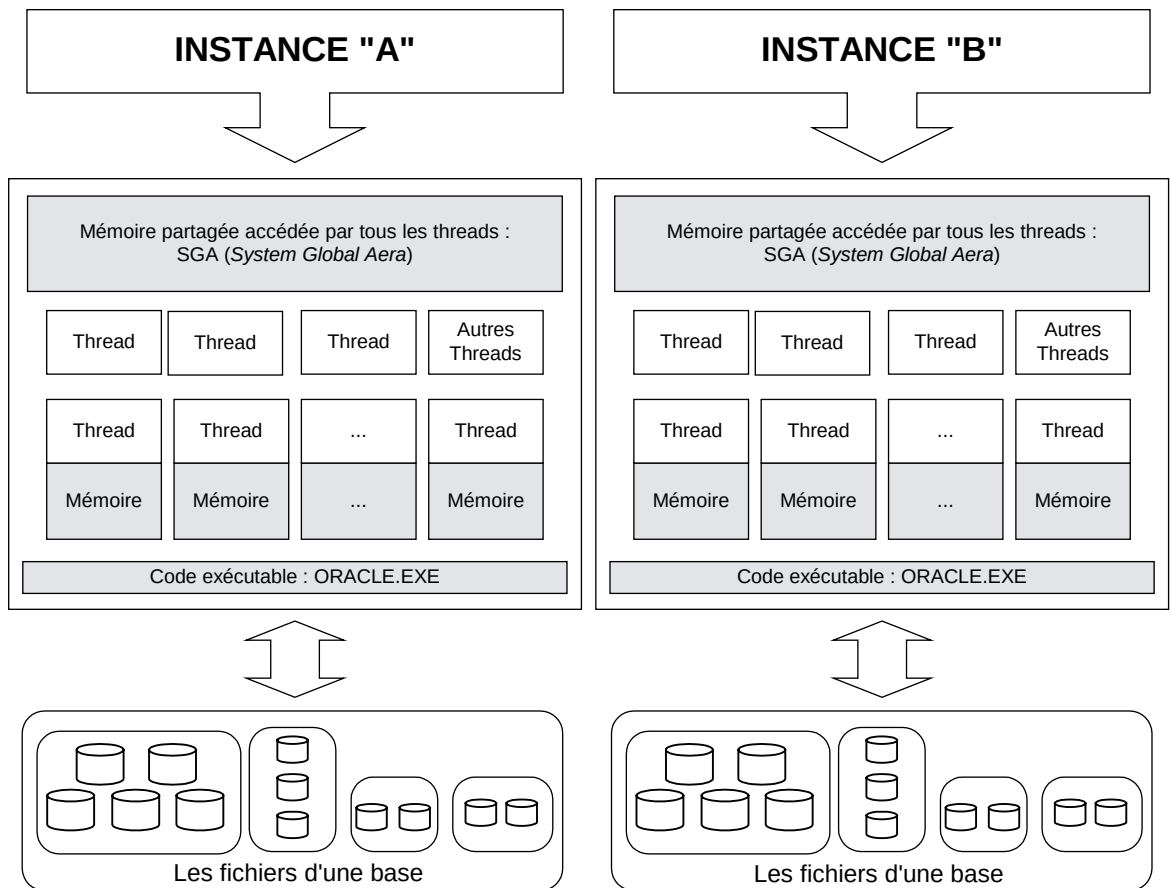


Figure 10-2

Deux instances sur un même serveur

- Un privilège au niveau du système d'exploitation fournit à un utilisateur Windows les droits nécessaires. C'est le cas de l'appartenance au groupe Windows *ORA_DBA*.
- Un utilisateur déclaré dans la base, disposant du privilège Oracle *SYSDBA* ou *SYSOPER*, peut démarrer ou arrêter la base, si sa sécurité d'accès repose sur des fichiers mots de passe stockés dans des fichiers extérieurs à la base.

Nous vous conseillons fortement d'utiliser la première méthode : elle est facile à mettre en œuvre et sa sécurité est optimale.

Le groupe Windows *ORA_DBA* est créé automatiquement lors de l'installation d'Oracle 10g. Lorsque vous êtes connecté à la base avec ce privilège, vous êtes placé dans le schéma propriété de SYS. Cela vous donne accès à tous les objets propriété de l'utilisateur SYS, dont le dictionnaire de données.

Le fichier d'initialisation

Pour démarrer une instance, Oracle 10g a besoin de lire un fichier d'initialisation comprenant une liste de paramètres. Les valeurs sont relatives à cette instance et cette base de données. Son contenu est décrit au chapitre 9, *Les fichiers d'une base Oracle 10g*.

Le fichier d'initialisation recherché par défaut par Oracle 10g se nomme *initSID.ora*, *SID* étant l'identifiant de votre instance. Par exemple, pour l'instance TEST, on aura le fichier *initTEST.ora*. Rien n'empêche de disposer de plusieurs fichiers d'initialisation pour une même instance, il suffit d'indiquer celui utilisé au démarrage.

Certains paramètres figurant dans le fichier d'initialisation peuvent être modifiés une fois l'instance lancée. Si tous ne sont pas présents dans le fichier d'initialisation, des valeurs par défaut sont attribuées automatiquement.

Le fichier d'initialisation peut avoir la forme d'un fichier texte persistant, dans un format géré par Oracle, le fichier *pfileSID.ora*. Ce point est traité au chapitre 23, *Gestion de l'espace disque et des fichiers*.

Démarrer une base Oracle 10g

Afin de la rendre disponible à l'ensemble des utilisateurs, le démarrage d'une base Oracle 10g s'effectue en quatre étapes :

- démarrage du service OracleServiceSID ;
- démarrage de l'instance ;
- montage de la base de données ;
- ouverture de la base de données.

Dans les précédentes versions d'Oracle, l'outil privilégié pour démarrer et arrêter une base était Server Manager. Cet outil a disparu car il est maintenant intégré dans SQL*Plus.

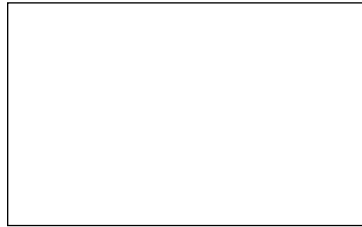
Démarrage du service OracleServiceSID

Le démarrage du service OracleServiceSID s'effectue à partir du gestionnaire de services Windows ou par la commande :

```
■ C:\> net start OracleServiceSID
```

Le démarrage du service peut entraîner celui de l'instance. Ce point est évoqué plus loin dans ce chapitre.

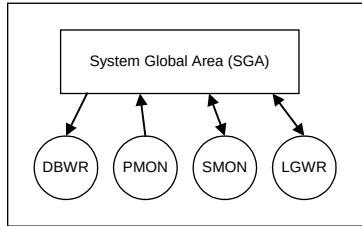
démarrage
du service



Lancer l'exécutable oracle.exe

démarrage du service OracleServiceSID,
SID étant l'identifiant de l'instance.

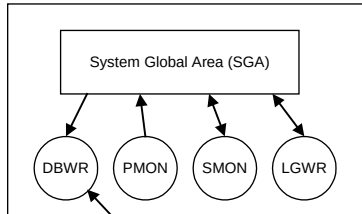
base non
montée



**Étape permettant la création
d'une base**

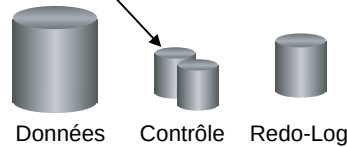
SQL :
sqlplus /nolog
connect / as SYSDBA
startup nomount

base
montée

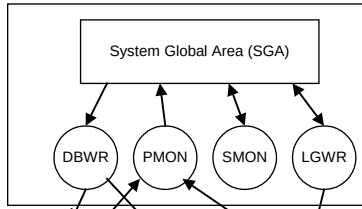
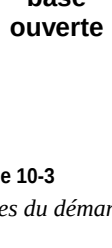


Étape permettant :
- l'ajout, la suppression de fichiers
de données;
- passage au mode ARCHIVELOG
et NOARCHIVELOG.

SQL :
sqlplus /nolog
connect / as SYSDBA
startup mount



base
ouverte



Étape de fonctionnement

SQL :
sqlplus /nolog
connect / as SYSDBA
startup open

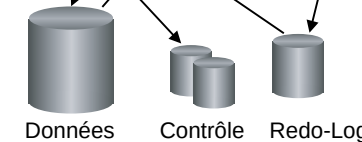


Figure 10-3

Étapes du démarrage d'une instance

Démarrage d'une instance

Pour démarrer une instance, Oracle 10g lit les valeurs présentes dans le fichier d'initialisation (pfileID.ora s'il existe, puis initSID.ora) puis alloue la SGA (*System Global Area*) en mémoire et lance les processus. À ce moment, aucune base de données, c'est-à-dire aucun fichier, n'est associée à cette instance.

```
C:\> sqlplus /nolog

connect / as sysdba

startup nomount pfile=C:\oracle\admin\TEST\scripts\initTEST.ora

Instance ORACLE démarrée
Zone globale système (SGA) totale de 10006476 octets
Fixed Size          65484 octets
Variable Size       8740864 octets
Database Buffers    1126400 octets
Redo Buffers        73728 octets
```

Montage de la base de données

Le montage associe les fichiers de contrôle à l'instance créée. Pour cela, le fichier d'initialisation contient l'emplacement des fichiers de contrôle (précisé par le paramètre `CONTROL_FILES` du fichier `initSID.ora`). Ceux-ci contiennent la liste de tous les autres fichiers de données et redo-log qui composent la base.

```
-- suite de l'exemple précédent
alter database mount ;

Instruction traitée
```

Ouverture de la base de données

L'ouverture de la base de données la rend disponible à l'ensemble des utilisateurs. Tout utilisateur peut s'y connecter et travailler.

```
-- suite de l'exemple précédent
alter database open;

Instruction traitée
```

Démarrage en un seul ordre

Toutes ces étapes peuvent être réalisées par un seul ordre SQL :

```
$ sqlplus /nolog

connect / as sysdba

startup pfile=C:\oracle\admin\TEST\scripts\initTEST.ora
```

```
Instance ORACLE démarrée
Zone globale système (SGA) totale de 10006476 octets
Fixed Size 65484 octets
Variable Size 8740864 octets
Database Buffers 1126400 octets
Redo Buffers 73728 octets
Base de données montée
Base de données ouverte
```

On retrouve les trois actions : le démarrage de l'instance, le montage de la base puis son ouverture.

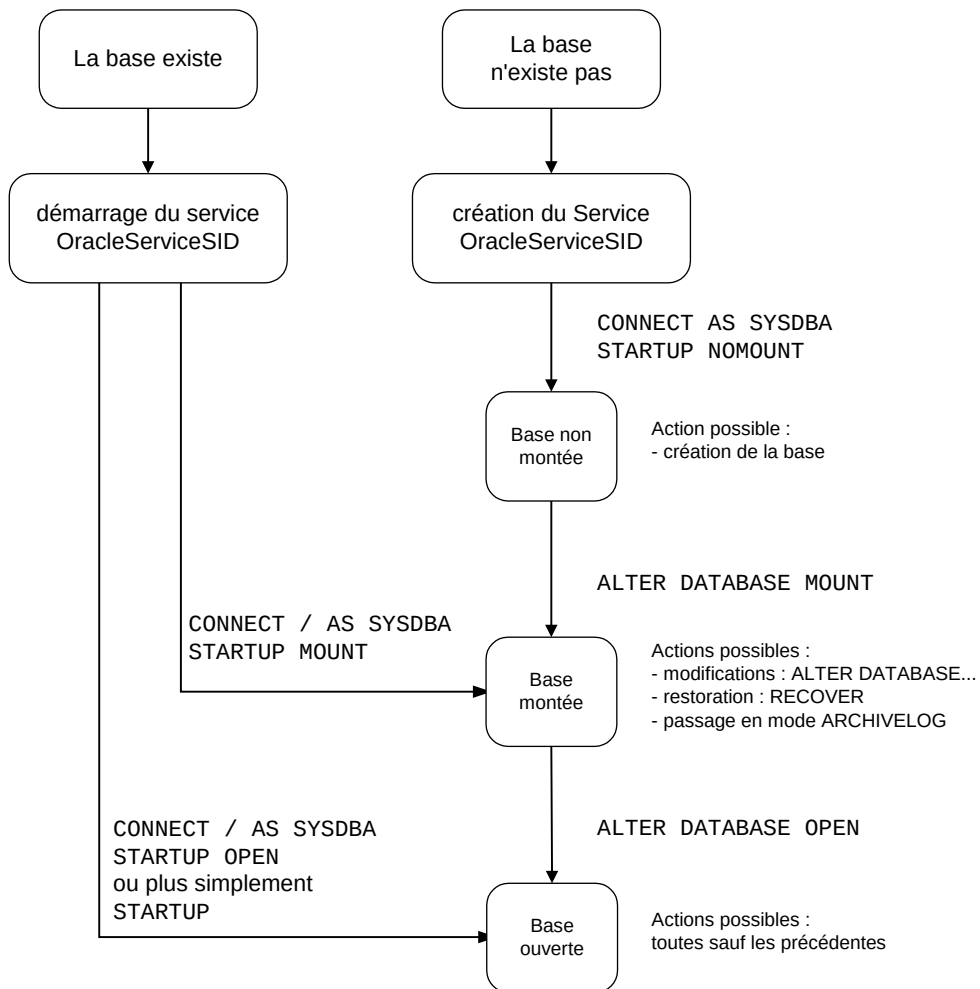


Figure 10-4

Ordres SQL lors du démarrage d'une instance

Arrêter une base Oracle 10g

Lors de l'arrêt d'une base Oracle 10g, les étapes sont identiques à celles rencontrées au démarrage :

- fermeture de la base de données ;
- démontage de la base de données ;
- arrêt de l'instance ;
- arrêt du service OracleServiceSID.

L'arrêt du service OracleServiceSID entraîne plus ou moins brutalement l'arrêt des étapes précédentes, selon le paramétrage choisi.

Les différents types d'arrêts

Lorsque vous fermez une base Oracle 10g, des utilisateurs peuvent être connectés. Comment sont alors traitées les transactions en cours ? Pour cela, il existe différentes options d'arrêts :

- SHUTDOWN NORMAL ;
- SHUTDOWN IMMEDIATE ;
- SHUTDOWN TRANSACTIONAL ;
- SHUTDOWN ABORT.

Le SHUTDOWN NORMAL attend que l'ensemble des utilisateurs soit déconnecté pour fermer la base. L'inconvénient, c'est que l'on peut attendre longtemps si, par exemple, un utilisateur s'est absenté sans fermer son application : sa session est toujours active. C'est pourquoi le SHUTDOWN IMMEDIATE déconnecte automatiquement tous les utilisateurs, puis ferme correctement votre base de données.

Le SHUTDOWN TRANSACTIONAL offre une alternative à ces méthodes : il attend la fin de toutes les transactions en cours en interdisant toute nouvelle transaction.

Pour arrêter votre base avant une sauvegarde, utilisez toujours un SHUTDOWN IMMEDIATE car un SHUTDOWN NORMAL risque de vous mettre en attente sans délai prévisible de fermeture. Même si aucun utilisateur n'est plus connecté à Oracle, il reste souvent des connexions liées au fonctionnement d'agents de surveillance, d'un « pool » de connexion PHP, etc.

Lors d'un SHUTDOWN NORMAL ou IMMEDIATE, Oracle 10g écrit toutes les informations modifiées encore présentes dans la SGA, dans les fichiers de données et redo-log. Ensuite, l'ensemble des fichiers *online* est fermé. (En effet, un fichier *offline* n'a plus de lien avec la base de données. Lorsque vous démarrez la base, un fichier ou un tablespace *offline* reste *offline*). À cet instant, la base est fermée et inaccessible pour des opérations normales. Si elle reste néanmoins montée, les fichiers de contrôle demeurent ouverts.

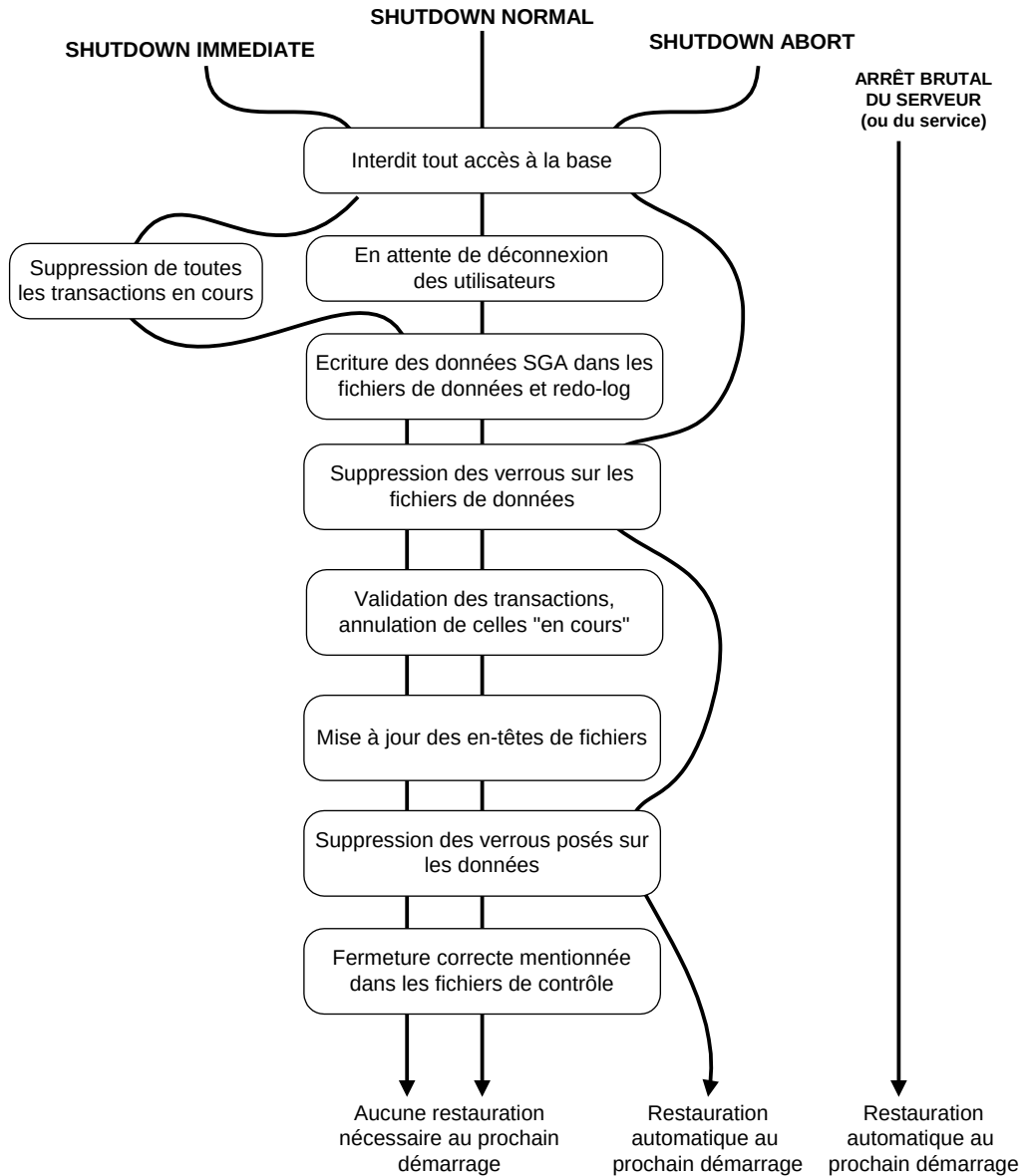


Figure 10-5

Les différents types de fermetures d'une base

Le SHUTDOWN ABORT est un arrêt violent qui correspond à supprimer brutalement le processus oracle.exe, ses threads, la SGA et libérer tous les verrous sur les fichiers. Cette option nécessite une restauration automatique au démarrage suivant. Cette méthode très rapide s'apparente à un arrêt électrique brutal de votre machine. Dans ce cas, il n'y a

aucune écriture des données de la SGA dans les fichiers de données ou redo-log. En conséquence, la prochaine ouverture de base nécessitera une opération de restauration, qu'Oracle 10g effectue automatiquement.

La restauration automatique de l'instance en cas d'arrêt brutal est l'un des points forts d'Oracle.

Le paramétrage du service OracleServiceSID

Des clés de registre coordonnent l'arrêt du service OracleServiceSID avec celui de l'instance. Elles sont situées en HKEY_LOCAL_MACHINE\SOFTWARE\ORACLE\KEY_OraDb10g_home1 et concernent aussi bien le démarrage que l'arrêt de l'instance.

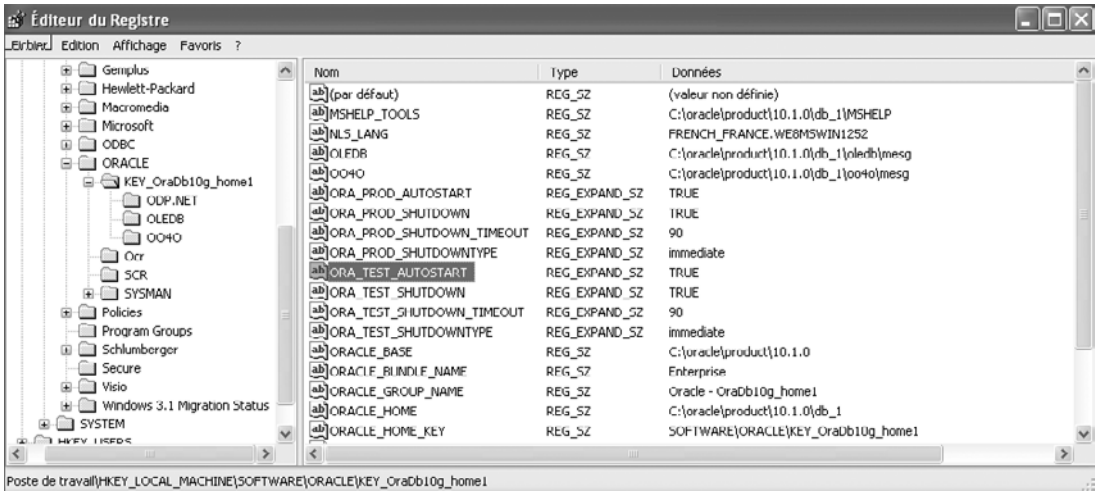


Figure 10-6
Clés du registre liées au service OracleServiceSID

Dans la figure précédente, on retrouve des clés identiques pour deux instances différentes : TEST et PROD.

ORA_SID_AUTOSTART	Indique si la base doit être démarrée en même temps que le service OracleServiceSID. Si le service est paramétré en démarrage automatique, il sera lancé en même temps que l'ordinateur et la base Oracle démarrera aussi.
ORA_SID_SHUTDOWN	Indique si la base doit être arrêtée en même temps que l'arrêt du service OracleServiceSID.
ORA_SID_SHUTDOWN_TIMEOUT	Période de temps accordée par le service pour que la base s'arrête. Au-delà, le service s'arrête, quelle que soit l'étape d'arrêt de la base.
ORA_SID_SHUTDOWNTYPE	Immediate, normal ou abort

Pour éviter l'accès direct à la base de registre, l'utilitaire Oracle Administration Assistant for Windows offre un accès graphique à ces clés de registre. Il est décrit au chapitre 5, *Fonctionnement d'Oracle 10g sous Windows*.

Fermeture de la base de données

La fermeture de la base de données la rend indisponible à l'ensemble des utilisateurs. Dans le cas suivant, la base est fermée, mais reste montée :

```
alter database close immediate;  
  
Instruction traitée
```

Démontage de la base de données

Le démontage de la base de données supprime l'association entre les fichiers de la base et l'instance. Dans le cas suivant, la base fermée est démontée, mais l'instance existe encore :

```
-- suite de l'exemple précédent  
  
alter database dismount ;  
  
Instruction traitée
```

Arrêt de l'instance

Pour arrêter l'instance, effectuez une fermeture « normale » de la base ; elle enchaîne les étapes déjà effectuées puis termine l'instance. Les processus sont supprimés, la SGA libérée.

```
-- suite de l'exemple précédent  
  
shutdown ;  
  
ORA-01507: base de données non montée  
Instance Oracle arrêtée
```

Fermeture en un seul ordre

Toutes ces étapes peuvent être réalisées par un seul ordre SQL :

```
$ sqlplus /nolog  
  
connect as sysdba  
  
shutdown immediate  
  
Base de données fermée  
Base de données démontée  
Instance ORACLE arrêtée
```

On retrouve les trois actions : la fermeture, le démontage de la base puis l'arrêt de l'instance.

Résumé

Ce chapitre a abordé succinctement l'installation d'Oracle 10g ainsi que la création d'une base de données. Puis, nous avons abordé dans le détail les différentes options de démarrage et d'arrêt d'une instance Oracle 10g.

11

Transactions et accès concurrents

Dans ce chapitre :

- la définition d'une transaction ;
- la gestion des accès concurrents ;
- les segments d'annulation ;
- le principe des verrous.

Toute base Oracle a pour objectif de fournir aux utilisateurs un accès simultané aux données. La notion de concurrence d'accès et de verrouillage des données intervient lorsque plusieurs utilisateurs essaient d'atteindre une même information au même instant. Le concept de transaction est différent mais se situe à la base de la gestion des accès concurrents : les données modifiées sont réservées jusqu'à leur validation.

Dans ce chapitre, nous explicitons les termes de concurrence d'accès et de transaction en mettant en évidence leur interaction. Ces mécanismes sont assurés par les segments d'annulation, eux aussi abordés. La mise en œuvre des segments d'annulation est abordée au chapitre 23, *Gestion de l'espace disque et des fichiers*.

Les transactions

Les transactions sont des mécanismes normalisés dans la définition du langage SQL. En revanche, leur implémentation est l'une des caractéristiques qui établit la différence entre les divers fournisseurs de base de données.

Notion de transaction

Les transactions doivent respecter les principes résumés par l'acronyme ACID :

Atomicité : une fois lancée, une transaction doit toujours s'exécuter complètement ou pas du tout. Elle ne peut pas être à moitié exécutée.

Cohérence : une transaction assure l'intégrité des données et réalise le passage des données d'un état cohérent à un autre. En cas de problème, pour respecter le principe d'atomicité, l'état cohérent initial est restauré.

Isolation : les transactions qui s'exécutent simultanément n'interfèrent pas entre elles. Elles s'exécutent comme si elles étaient lancées séquentiellement.

Durabilité : les résultats d'une transaction qui s'est déroulée correctement ne doivent pas être affectés par une panne quelconque du système.

Une transaction n'a que deux issues possibles : le succès « COMMIT » ou l'échec « ROLLBACK ». Il n'y a pas de situation intermédiaire.

Une *transaction* est donc un ensemble d'ordres SQL qui ont pour objectif de faire passer la base de données, *en une seule étape*, d'un état cohérent à un autre état cohérent. Elle comporte un début, une suite d'ordres SQL, puis une fin.

Si elle réussit, la base de données est dans un nouvel état cohérent. Si elle échoue (volontairement ou involontairement), les modifications déjà effectuées dans la base sont annulées, de sorte qu'elle retrouve l'état cohérent antérieur au début de la transaction. C'est Oracle 10g qui se charge entièrement de toute cette gestion.

Une application, quel que soit l'outil de développement, est composée d'un ensemble de transactions. Le choix et le contenu des transactions sont laissés à l'appréciation des développeurs et parfois même à celle de l'utilisateur final. Les ordres SQL sont regroupés au sein d'une transaction en fonction de plusieurs critères :

- les ordres SQL d'une même transaction se complètent souvent pour préserver la *cohérence* de la base. C'est le cas d'opérations comptables de type « débit-crédit », dans lesquelles chaque opération de débit doit obligatoirement s'accompagner d'un crédit sur un autre compte. Les ordres de débit et de crédit sont alors situés dans la même transaction ;
- pour répondre à des impératifs d'*ergonomie*, chaque transaction ne doit pas être composée d'un trop grand nombre d'ordres SQL. En effet, il faut éviter la suppression de nombreuses informations saisies par l'utilisateur en cas d'annulation d'une transaction. Les règles d'*ergonomie* préconisent la validation fréquente de transactions courtes ;

- pour répondre à des impératifs de *performance*, les transactions ne doivent pas être trop « volumineuses », comme le sont par exemple les programmes *batch* qui manipulent de gros volumes de données. Ils ont le choix d'une validation ligne à ligne ou globale à la fin des modifications. Si les critères de cohérence ne sont pas en jeux, les impératifs de performance interviennent alors dans le choix de la solution.

Seuls les ordres SQL qui modifient des données interviennent dans une transaction. Un ordre SELECT qui n'effectue qu'une lecture n'influe pas sur la transaction à laquelle il appartient.

Début et fin d'une transaction

Une transaction commence à la fin de la précédente transaction. La toute première transaction débute au lancement du programme. Il n'existe pas d'ordre implicite de début de transaction.

Elle se termine normalement par l'ordre SQL COMMIT qui la valide ou ROLLBACK qui l'annule.

Certains ordres SQL possèdent un « COMMIT implicite », par exemple, les ordres du langage de définition de données (LDD) tels que CREATE, DROP, GRANT, ALTER, etc. Toutes les données modifiées précédemment sont alors validées par l'ordre contenant le COMMIT implicite.

En cas de déconnexion d'un programme, tout dépend de l'outil utilisé ou de la programmation effectuée. Ainsi, une sortie « normale » de l'outil SQL*Plus entraîne un COMMIT implicite. Une sortie « brutale » (par exemple, une interruption inattendue du processus) assure la validation des transactions déjà validées, mais annule (ROLLBACK) les ordres SQL de celles en cours.

Toute transaction qui se termine anormalement est donc annulée. Dans le pire des cas, s'il produit l'arrêt électrique de la machine qui héberge la base de données, Oracle 10g garantit que toutes les transactions déjà validées par un COMMIT ou un ROLLBACK seront assurées car elles figurent dans les fichiers redo-log. Au redémarrage de l'instance, Oracle 10g consulte ces fichiers et efface toutes les transactions en cours qui n'étaient ni validées, ni supprimées. Ce mécanisme d'annulation automatique ne nécessite aucune intervention de l'administrateur Oracle 10g.

Les savepoints

Les savepoints sont des points de contrôle utilisés dans les transactions pour annuler partiellement l'une d'elles. Dans ce cas, un savepoint est défini par un identifiant et peut être référencé dans la clause ROLLBACK.

L'exemple suivant décrit le principe d'utilisation des savepoints :

```
SQL> -- début de transaction
SQL> ordre SQL 1
SQL> ordre SQL 2
```

```
SQL> SAVEPOINT SP1 ;
SQL> ordre SQL 3
SQL> si condition X alors ROLLBACK TO SP1 ;
SQL> SAVEPOINT SP2 ;
SQL> ordre SQL 4
SQL> ordre SQL 5
SQL> si condition Y alors ROLLBACK TO SP2 ;
SQL> ordre SQL 6
SQL> fin de la transaction par un COMMIT ou ROLLBACK.
```

Si la condition X est vérifiée, l'ordre SQL 3 est annulé et le programme continue. Si la condition Y est vérifiée, il y a annulation des ordres SQL 4 et 5. Si la fin de la transaction est un ROLLBACK, toutes les modifications sont supprimées. S'il s'agit d'un COMMIT, tous les ordres SQL sont validés, sauf ceux annulés par un ROLLBACK TO.

La notion de SAVEPOINT est très utile dans la conception de programmes batch, quel que soit le langage utilisé, dont le PL/SQL.

Les segments d'annulation

Chaque base de données abrite un ou plusieurs segment d'annulation ou segment UNDO. Il contient les anciennes valeurs des enregistrements en cours de modification dans les transactions, qui sont utilisés pour assurer une lecture consistante des données, pour annuler des transactions et en cas de restauration.

Ni les utilisateurs ni l'administrateur ne peuvent accéder ou lire le contenu d'un segment d'annulation ; seul le logiciel Oracle 10g peut l'atteindre. Il est la propriété exclusive de l'utilisateur SYS, quel que soit son créateur.

Principe d'un segment d'annulation

Un segment d'annulation ou UNDO gère toutes les transactions de la base. Entre autres choses, il comporte le numéro de fichier, l'identifiant du block contenant les données modifiées et le block de données dans son état initial. Oracle 10g enregistre toutes les actions d'une transaction dans un même segment. Ainsi, la base dispose de tous les éléments pour effectuer l'annulation d'une transaction, c'est-à-dire son ROLLBACK. Dans ce cas, Oracle 10g copie le bloc de données depuis le segment d'annulation vers son ancien emplacement.

Segment UNDO et rollback segment

Oracle 10g apporte le principe du segment UNDO. Auparavant, seuls les rollback segment existaient. Ils étaient d'une gestion complexe et source de bien des difficultés pour l'administrateur Oracle.

Oracle affectait alors chaque transaction à un rollback segment unique. En revanche, un rollback segment contenait simultanément les transactions de plusieurs utilisateurs.

Pour minimiser les contentions d'accès aux disques, il était très fortement conseillé d'utiliser plusieurs rollback segments. Une méthode simple consistait à disposer de $n/3$ rollback segments, n étant le nombre maximal d'utilisateurs accédant simultanément à la base.

Chaque base Oracle 10g contient encore un rollback segment, propriété de l'utilisateur SYS et situé dans le tablespace SYSTEM. Il est affecté à la gestion du dictionnaire de données et l'administrateur Oracle ne doit effectuer aucune action sur lui.

Par leur simplicité et facilité de gestion, nous vous conseillons d'utiliser les tablespaces de type UNDO décrits au chapitre 23, *Gestion de l'espace disque et des fichiers*.

Les lectures consistantes

Une des caractéristiques d'Oracle 10g est sa capacité à gérer l'accès concurrent aux données, c'est-à-dire l'accès simultané de plusieurs utilisateurs à la *même* donnée. Sans des mécanismes perfectionnés de contrôle des accès, les mises à jour simultanées des données pourraient compromettre leur intégrité. Pour pallier la difficulté, il suffit, pour de multiples utilisateurs, de n'autoriser que des accès successifs à une même information, créant ainsi des temps d'attente. Le rôle d'Oracle 10g sera de réduire cette attente jusqu'à la rendre imperceptible.

La lecture consistante, telle qu'elle est prévue par Oracle 10g assure :

- qu'au sein d'un ordre SQL, les données interrogées ou manipulées ne changeront pas de valeur entre le début et la fin. Tout se passe comme si un « cliché » était effectué sur la totalité de la base au début de l'ordre et que seul ce cliché soit utilisé tout au long de son exécution ;
- les lectures ne seront pas bloquées par des utilisateurs effectuant des modifications sur les mêmes données ;
- les modifications ne seront pas bloquées par des utilisateurs effectuant des lectures sur ces données ;
- un utilisateur ne peut lire les données modifiées par un autre, si elles n'ont pas été validées. Pour lui, elles resteront dans leur état validé précédent ;
- il faudra patienter pour modifier des informations en cours de modification dans une autre transaction.

Pour gérer à la fois les lectures et les mises à jour, Oracle 10g conserve les deux informations : les données mises à jour sont écrites dans les segments de données de la base et les anciennes valeurs sont consignées dans les segments d'annulation ou UNDO segments. Ainsi, l'utilisateur de la transaction modifiant les valeurs lira les données modifiées et tous les autres liront les données non modifiées. Un autre utilisateur accèdera à la fois aux données conservées dans la table (toutes les données n'ont pas été modifiées) et aux anciennes données stockées dans les segments d'annulation.

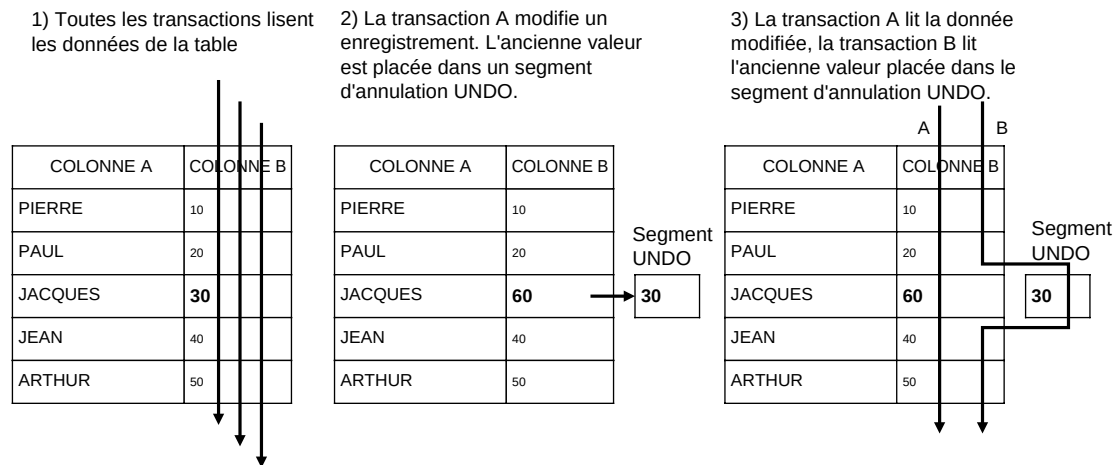


Figure 11-1

Principe de la lecture consistante

Ce n'est qu'au moment de la validation de la transaction (COMMIT) que les modifications effectuées deviennent permanentes. Le segment d'annulation et les verrous posés sur les enregistrements modifiés sont libérés.

Seuls les ordres qui ont débuté après la validation accèdent aux données modifiées. Tous les autres accèdent aux données conservées dans les segments d'annulation.

En cas d'annulation (ROLLBACK), les informations écrites dans le segment d'annulation viennent supprimer les données modifiées, non encore validées. Le segment d'annulation et les verrous posés sur les enregistrements modifiés sont ensuite libérés.

L'utilisateur de la transaction « B » (figure précédente), qui relance son ordre SQL après la validation par « A », lira la donnée modifiée. Si vous souhaitez effectuer de nombreux ordres SQL sans que les informations apparaissent modifiées entre le début du premier ordre et la fin du dernier, utilisez les transactions en lecture seule (read only) . Vous pouvez alors exécuter autant d'ordres SQL que vous voulez, sur toutes les tables. Elles seront comme figées, telles qu'elles étaient au moment du « top chrono » donné par le « read-only ».

```
SQL> -- début de la transaction en mode lecture seule
SQL> -- SET TRANSACTION doit être le premier ordre de la transaction
SQL> commit ; # commencer par un état stable
SQL> set transaction read only ;
SQL>
SQL> select * from tableA ;
...
SQL> select * from tableB ;
...
SQL> commit ; -- indique la fin du " read-only "
```

Les transactions "read only" introduisent une faible baisse des performances de la base de données, car les autres transactions continuent de modifier les données. Elles utilisent donc plus fortement les segments d'annulation.

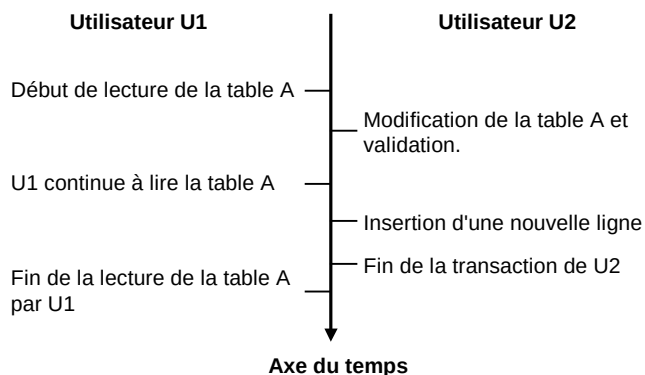
La pose de verrous

Lorsque plusieurs utilisateurs accèdent simultanément à la base de données, nous sommes en présence d'une *concurrency d'accès*. S'ils accèdent en même temps aux mêmes objets, ils créent un conflit d'accès. Ce conflit est résolu par le mécanisme de verrouillage, dont le principe consiste à réserver un objet à un utilisateur et à empêcher les autres de le modifier, tant qu'il n'a pas signifié la fin de sa transaction par un ordre COMMIT ou ROLLBACK.

C'est une surprise pour beaucoup : depuis de nombreuses années, Oracle gère automatiquement tous les verrous d'accès aux données, la concurrence d'accès, etc. Il est rare que l'intervention de l'utilisateur soit requise.

Un ordre de lecture seul (SELECT) **ne pose aucun verrou et ne bloque aucune autre transaction**. L'ordre SELECT visualise uniquement des données couvertes par une transaction (COMMIT ou ROLLBACK) précédente. Cela signifie que, si des données ont été validées par d'autres transactions entre le début et la fin de l'ordre SELECT, il ne les lira pas modifiées mais dans leur état antérieur, même si elles lui sont accessibles. Les données en cours de modification par les autres utilisateurs ne sont jamais accessibles. La figure suivante décrit ce principe : l'utilisateur U1 ne verra aucune des modifications effectuées par l'utilisateur U2, car elles ont lieu après le début de l'ordre SELECT lancé par U1.

Figure 11-2
*Principe d'une lecture
implicite d'Oracle 10g*



Dans la figure précédente, si dès la fin de sa transaction l'utilisateur U1 relance le même ordre SQL de lecture de la table A, il lira les informations corrigées par l'utilisateur U2.

Un ordre SQL de mise à jour (INSERT, UPDATE, DELETE) pose des verrous sur les données modifiées et place les informations nécessaires dans les segments d'annulation.

Seul *l'enregistrement* est verrouillé. Même si 80 % des enregistrements d'une table sont verrouillés, il n'y a pas d'escalade de verrou au niveau du block de données ou sur la totalité de la table.

Il existe deux types de verrous : les verrous exclusifs et les verrous partagés. De nombreux verrous partagés peuvent être placés sur la même ressource : ils ne se bloquent pas mutuellement. En revanche, un seul verrou exclusif peut être placé sur une ressource. L'action qui tente d'en poser un second sur la même ressource sera refusée. C'est le cas de deux utilisateurs désirant modifier un même enregistrement.

Il est possible d'utiliser des ordres SQL pour poser soi-même des verrous sur des données ou des tables. L'un d'entre eux est très intéressant, car il permet de réserver des enregistrements à modifier : c'est le `SELECT ... FOR UPDATE`. Il est largement utilisé dans le langage de programmation PL/SQL. Nous vous déconseillons fortement l'utilisation de `SQL LOCK TABLE` car le verrouillage automatique d'Oracle 10g est plus performant (et surtout plus simple) que la gestion manuelle.

Résumé

Ce chapitre a mis en exergue la puissance et l'utilité des mécanismes internes d'Oracle 10g qui permettent de décharger les programmeurs d'opérations fastidieuses, répétitives et complexes. L'accès simultané aux données et la concurrence d'accès sont ainsi automatiquement gérés. Les mécanismes de verrouillage sont transparents. Enfin, les segments d'annulation offrent la possibilité d'implémenter des fonctionnalités puissantes comme la gestion des transactions et la lecture consistante implicite.

12

Le dictionnaire de données Oracle 10g

Dans ce chapitre :

- le rôle clé du dictionnaire de données d'Oracle 10g ;
- la création du dictionnaire de données ;
- les vues qui permettent d'interroger le dictionnaire de données ;
- la liste commentée des vues du dictionnaire de données.

Nous avons exploré la structure physique de la mémoire, des processus et des fichiers qui composent une instance Oracle 10g. Ce chapitre est consacré à l'exploration du cœur de la base de données, son dictionnaire. Sa connaissance est indispensable à tout administrateur Oracle 10g.

Le dictionnaire Oracle 10g

Le dictionnaire Oracle 10g contient les informations nécessaires au fonctionnement de la base. Il en existe un par base qui comporte tous les détails relatifs à sa structure : les tables, leur propriétaire, le nom et les types de colonnes, le taux de remplissage des tablespaces et bien d'autres choses encore...

Propriété de l'utilisateur SYS, le dictionnaire est créé en même temps que la base. L'utilisateur SYSTEM est l'administrateur, il possède des droits de lecture sur des vues accédant au contenu du dictionnaire. En quelques minutes, un expert ayant accès à votre dictionnaire

de données en connaîtra beaucoup plus que ce que vous auriez pu lui apprendre dans le même intervalle de temps. Il s'agit donc d'un instrument très précieux à sécuriser ; c'est la raison pour laquelle les mots de passe des utilisateurs SYS et SYSTEM devront être changés dès la fin de l'installation et précieusement conservés par la suite.

Les mots de passe initiaux de SYS et SYSTEM doivent être notés et sauvegardés dans une enveloppe de secours mise en sécurité.

Création du dictionnaire Oracle 10g

Le dictionnaire de données est créé d'emblée lors de l'installation de la base. Par la suite, après chaque action, Oracle 10g le met automatiquement à jour. Comme tout le reste de la base Oracle 10g, il est composé de tables et d'objets. Ils sont installés dans le tablespace SYSTEM et sont la propriété de l'utilisateur SYS.

L'utilisateur SYSTEM ne peut pas modifier la structure du dictionnaire, seules les commandes du langage SQL et les processus Oracle peuvent agir sur son contenu. Par contre, de nombreuses vues sont accessibles en lecture pour exploiter le dictionnaire. Elles sont hiérarchisées pour autoriser une vision plus ou moins large des objets décrits dans le dictionnaire.

Les scripts indispensables pour toutes les bases Oracle 10g sont décrits dans le tableau suivant. Ils sont placés en ORACLE_HOME\rdbms\admin.

Nom du script	Utilisé par	Commentaire
<i>sql.bsq</i>	Toutes les bases	Création du dictionnaire de données, lancé automatiquement par l'ordre SQL CREATE DATABASE.
<i>catalog.sql</i>	Toutes les bases	Création des vues, des synonymes et des droits d'accès sur le dictionnaire.
<i>catproc.sql</i>	Toutes les bases	Lance tous les scripts nécessaires à l'utilisation du PL/SQL. Il est indispensable à toutes les bases.
<i>catrep.sql</i>	Toutes les bases	Lance tous les scripts nécessaires à l'utilisation de la réplication avancée.

N'exécutez jamais un script d'une version Oracle sur une autre version de base de données Oracle.

D'autres scripts peuvent être nécessaires, suivant les options installées ou les fonctions que la base doit assurer. Les scripts SQL générés par l'assistant DBCA (*Database Configuration Assistant*) en sont le parfait exemple. Ils sont présentés au chapitre 14, *Création d'une base Oracle 10g*.

Une fois votre base créée, ne relancez ces scripts qu'en cas de passage d'une version d'Oracle à une autre version, en suivant scrupuleusement les indications de migration. Consultez ces scripts, car ils sont riches d'enseignements mais ne les modifiez jamais !

Ajout de produits optionnels au dictionnaire

Oracle 10g fournit d'autres scripts SQL offrant différentes options aux bases de données, par exemple, ceux autorisant la réplication ou l'installation du moteur Java dans Oracle. Ces scripts sont situés en `ORACLE_HOME\rdbms\admin`. Un certain nombre d'entre eux sont lancés automatiquement lors de la création du dictionnaire Oracle 10g, au moyen de `catalog.sql` et `catproc.sql`. Les scripts SQL générés par l'assistant DBCA (*Database Configuration Assistant*) sont un bon exemple. Ils sont présentés au chapitre 14, *Création d'une base Oracle 10g*.

Nom du script	Utilisé par	Lancé par	Commentaire
<i>catblock.sql</i>	Gestion des performances	SYS	Création de vues qui indiquent dynamiquement les verrous.
<i>cath.sql</i>	Services	SYS	Packages d'administration.
<i>catio.sql</i>	Gestion des performances	SYS	Suivi des I/O table par table.
<i>catockt.sql</i>	Sécurité	SYS	Packages utilisés pour gérer le cryptage Oracle 10g.
<i>catparr.sql</i>	Real Application Clusters	SYS	Création des vues utilisées par le Real Application Clusters, le cluster Oracle 10g.
<i>catrep.sql</i>	Réplication avancée	SYS	Exécute tous les scripts SQL nécessaires à l'option réplication avancée.
<i>catrman.sql</i>	Recovery Manager	RMAN	Création des tables et des vues utilisées par RMAN, l'outil de gestion des sauvegardes et restaurations.
<i>dbmspool.sql</i>	Gestion des performances	SYS	Permet au DBA de forcer le maintien dans la SGA de packages PL/SQL.
<i>utlbatat.sql</i> et <i>utlestat.sql</i>	Suivi des performances	SYS	Lance et arrête la collecte d'informations concernant les performances de votre instance.
<i>utlchain1.sql</i>	Gestion du stockage	tout utilisateur	Création de la table contenant les informations de la commande SQL ANALYZE TABLE ... CHAINED ROWS.
<i>utlhttp.sql</i>	Accès Web	SYS	Package PL/SQL destiné à gérer les échanges HTTP avec la base.
<i>utllockt.sql</i>	Suivi des performances	SYS	Affichage des verrous.
<i>utlrp.sql</i>	PL/SQL	SYS	Recompile tous les modules PL/SQL présents dans l'état INVALID.
<i>utlsampl.sql</i>	Exemples	SYS ou utilisateur DBA	Création d'un jeu d'exemple complet : l'utilisateur SCOTT, ses tables et leurs données.
<i>utlxplan.sql</i>	Suivi des performances	tout utilisateur	Création de la table PLAN_TABLE utilisée pour stocker les informations issues de la commande SQL EXPLAIN PLAN.

D'autres scripts existent à l'emplacement `ORACLE_HOME\rdbms\admin`. Nous vous conseillons de les consulter, car ils sont très riches d'enseignements.

Les différents types de vues

Les vues USER, ALL, DBA, V\$ sont des « familles » de vues dont le nom de chaque membre débute par le préfixe l'identifiant.

Pour un DBA Oracle 10g, l'accès au dictionnaire de données par les vues USER, ALL, DBA, V\$ est le moyen privilégié de connaître absolument toutes les caractéristiques de sa base.

DBA : les vues de l'administrateur Oracle 10g

Les vues DBA accèdent au dictionnaire pour fournir les informations dont seul un administrateur peut bénéficier. Logiquement, la liste de ces vues DBA est la plus fournie, car de nombreux aspects de la base relèvent de la compétence exclusive du DBA. Ainsi, les vues traitant de la configuration générale de la base, des segments d'annulation, des droits des différents utilisateurs, etc. n'apparaissent qu'à cet endroit.

ALL : les vues sur tous les objets accessibles

Pour permettre d'atteindre les données autorisées, les vues ALL accèdent aux informations sur les objets propriétés de l'utilisateur et sur ceux pour lesquels des droits d'accès lui ont été fournis.

USER : les vues sur les objets de l'utilisateur

Les vues USER accèdent exclusivement aux objets créés par l'utilisateur et ne fournissent aucune information sur les autres, même si l'utilisateur en possède les droits d'accès.

V\$: les vues dynamiques de la base

Elles fournissent des informations dynamiques sur le fonctionnement et les performances de la base en activité, les résultats obtenus évolueront chaque fois que ces vues seront consultées. Ces vues « attaquent » directement des structures écrites en langage C. Comme elles font partie intégrante du noyau Oracle 10g, elles sont accessibles dès que la base est montée mais non encore ouverte (dans ce cas, les informations proviennent de la lecture des fichiers de contrôle).

C'est pourquoi certaines informations statiques, comme la liste des fichiers (V\$DBFILE) et les informations concernant les fichiers redo-log (V\$LOGHIST et V\$LOGFILE) appartiennent aux vues V\$, car elles sont utilisées en cas de restauration de la base (qui est alors montée mais non ouverte). Les autres vues concernent essentiellement des aspects de performance et de tuning.

GV\$: les vues des bases en cluster

Le fonctionnement d'Oracle en configuration « cluster » a été abordé au chapitre 10, *Démarrer et arrêter une base Oracle 10g*. Dans ce cas de figure, pour surveiller les

performances d'instances placées sur des serveurs différents, il est important de disposer de vues identiques aux vues V\$ mais qui permettent d'identifier l'instance surveillée. C'est ce que permettent les vues GV\$. Elles contiennent toutes une colonne supplémentaire par rapport aux vues V\$ qui identifie l'instance interrogée. Comme le reste de leur contenu est identique, reportez-vous aux vues V\$.

Les autres vues

D'autres existent, comme la très utile vue DICTIONARY (voir paragraphe suivant). Elles proviennent d'anciennes versions d'Oracle 10g, avant la structuration en familles V\$, USER, ALL, DBA, mais elles sont de moins en moins utilisées.

Comment identifier la vue que vous cherchez ?

La vue DICTIONARY permet d'accéder à l'ensemble des vues DBA, ALL, USER, V\$ du dictionnaire. Elle est composée de deux colonnes qu'il suffit d'interroger. Comme Oracle 10g avec toutes les options installées possède plus de 1100 vues, il est important d'utiliser des outils simples pour identifier les vues que vous recherchez. Par exemple, pour retrouver toutes les vues qui traitent des « fichiers », lancez l'interrogation suivante :

```
SQL> desc dictionary
      NomNull ?Type
-----
TABLE_NAMEVARCHAR2(30)
COMMENTS  VARCHAR2(4000)

select table_name, comments from dictionary
       where table_name like '%FILE%' ;

TABLE_NAMECOMMENTS
-----
DBA_DATA_FILESInformation about database data files
DBA_EXP_FILESDescription of export files
DBA_PROFILES  Display all profiles and their limits
DBA_TEMP_FILESInformation about database temp files
V$BACKUP_DATAFILE Synonym for V_$BACKUP_DATAFILE
V$CONTROLFILESynonym for V_$CONTROLFILE
V$CONTROLFILE_RECORD Synonym for V_$CONTROLFILE_REC ORD_SECTION
V$DATAFILESynonym for V_$DATAFILE
V$DATAFILE_COPYSynonym for V_$DATAFILE_COPY
V$DATAFILE_HEADER Synonym for V_$DATAFILE_HEADER
V$DBFILESynonym for V_$DBFILE
V$FILESTATSynonym for V_$FILESTAT
V$FILE_PINGSynonym for V_$FILE_PING
V$LOGFILE Synonym for V_$LOGFILE
V$PROXY_DATAFILESynonym for V_$PROXY_DATAFILE
V$PWFILE_USERSSynonym for V_$PWFILE_USERS
V$RECOVERY_FILE_STAT Synonym for V_$RECOVERY_FILE_STATUS
V$RECOVER_FILESynonym for V_$RECOVER_FILE
V$TEMPFILESynonym for V_$TEMPFILE
```

Cette méthode permet de cibler très rapidement la ou les vues qui vous intéressent. Utilisez-la le plus souvent possible !

Comment obtenir la description des vues que vous cherchez ?

L'exemple précédent permet de récupérer un ensemble de vues dont le nom contient FILE (fichier). Vous pouvez aller plus loin dans la connaissance de ces vues en affichant leurs colonnes. Cette opération peut être effectuée automatiquement par le script suivant ;

```
set echo on
set pagesize 500
spool fichier_desc.sql

select 'desc '||table_name from dictionary
where table_name like upper '%&parametre_a_saisir%' ;

spool off
spool fichir_desc.txt
start fichier_desc.sql
spool off
```

L'exécution de ce fichier de commande donne le résultat suivant :

```
select 'desc '||table_name from dictionary
where table_name like upper '%&parametre_a_saisir%' ;
```

Entrez la valeur pour parametre_a_saisir : FILE

```
desc DBA_DATA_FILES
desc DBA_EXP_FILES
. . . autres vues . . .
```

```
spool off
spool fichir_desc.txt
start fichier_desc.sql
```

```
SQL> desc DBA_DATA_FILES
  NomNULL ?Type
-----
FILE_NAME  VARCHAR2(513)
FILE_IDNUMBER
TABLESPACE_NAME VARCHAR2(30)
BYTESNUMBER
BLOCKS  NUMBER
STATUS  VARCHAR2(9)
RELATIVE_FNUMBER
AUTOEXTENSIBLE VARCHAR2(3)
MAXBYTESNUMBER
MAXBLOCKS  NUMBER
```

```

INCREMENT_BYNUMBER
USER_BYTESNUMBER
USER_BLOCKSNUMBER

SQL> desc DBA_EXP_FILES
NomNULL ?Type
-----
EXP_VERSIONNOT NULL NUMBER(3)
EXP_TYPEVARCHAR2(11)
FILE_NAMENOT NULL VARCHAR2(100)
USER_NAMENOT NULL VARCHAR2(30)
TIMESTAMPNOT NULL DATE

... Suite de la description des vues ...
spool off
# la description des vues est contenue
# dans le fichier fichier_desc.txt

```

Nous ne fournissons pas le descriptif des colonnes de l'ensemble des vues du dictionnaire car cela nécessite environ 120 pages ! Mais, pour vos propres besoins, vous pouvez le réaliser à l'aide de la méthode précédente.

Liste commentée des vues

La majorité des vues porte un nom explicite. Par exemple, DBA_TABLES accède au détail de toutes les tables de la base de données. La liste suivante fournit l'intégralité des vues du dictionnaire de données.

Pour des raisons de place, seul le nom de l'objet figure. Vouloir ajouter le commentaire conduirait à une soixantaine de pages de texte. Si vous désirez ces commentaires, modifiez l'ordre :

```

select table_name from dictionary order by table_name ;

par

select table_name, comments from dictionary order by table_name ;

```

ALL_ALL_TABLES	ALL_ARGUMENTS	ALL_CAPTURE_EXTRA_ATTRIBUTES
ALL_APPLY	ALL_ASSOCIATIONS	ALL_CAPTURE_PARAMETERS
ALL_APPLY_CONFLICT_COLUMNS	ALL_AUDIT_POLICIES	ALL_CAPTURE_PREPARED_DATABASE
ALL_APPLY_DML_HANDLERS	ALL_AUDIT_POLICY_COLUMNS	ALL_CAPTURE_PREPARED_SCHEMAS
ALL_APPLY_ENQUEUE	ALL_AW_NUMBERS	ALL_CAPTURE_PREPARED_TABLES
ALL_APPLY_ERROR	ALL_AW_OBJ	ALL_CATALOG
ALL_APPLY_EXECUTE	ALL_AW_PROP	ALL_CLUSTER_HASH_EXPRESSIONS
ALL_APPLY_KEY_COLUMNS	ALL_AW_PS	ALL_CLUSTERS
ALL_APPLY_PARAMETERS	ALL_AWS	ALL_COL_COMMENTS
ALL_APPLY_PROGRESS	ALL_BASE_TABLE_MVIEWS	ALL_COLL_TYPES
ALL_APPLY_TABLE_COLUMNS	ALL_CAPTURE	ALL_COL_PRIVS

ALL_COL_PRIVS_MADE	ALL_JOBS	ALL_PROCEDURES
ALL_COL_PRIVS_REC'D	ALL_JOIN_IND_COLUMNS	ALL_PROPAGATION
ALL_CONS_COLUMNS	ALL_LIBRARIES	ALL_PUBLISHED_COLUMNS
ALL_CONS_OBJ_COLUMNS	ALL_LOB_PARTITIONS	ALL_QUEUE_PUBLISHERS
ALL_CONSTRAINTS	ALL_LOBS	ALL_QUEUES
ALL_CONTEXT	ALL_LOB_SUBPARTITIONS	ALL_QUEUE_TABLES
ALL_DB_LINKS	ALL_LOB_TEMPLATES	ALL_REFRESH
ALL_DEF_AUDIT_OPTS	ALL_LOG_GROUP_COLUMNS	ALL_REFRESH_CHILDREN
ALL_DEPENDENCIES	ALL_LOG_GROUPS	ALL_REFRESH_DEPENDENCIES
ALL_DIM_ATTRIBUTES	ALL_METHOD_PARAMS	ALL_REFS
ALL_DIM_CHILD_OF	ALL_METHOD_RESULTS	ALL_REGISTERED_MVIEWS
ALL_DIMENSIONS	ALL_MVIEW_AGGREGATES	ALL_REGISTERED_SNAPSHOTS
ALL_DIM_HIERARCHIES	ALL_MVIEW_ANALYSIS	ALL_REGISTRY_BANNERS
ALL_DIM_JOIN_KEY	ALL_MVIEW_COMMENTS	ALL_REPAUDIT_ATTRIBUTE
ALL_DIM_LEVEL_KEY	ALL_MVIEW_DETAIL_RELATIONS	ALL_REPAUDIT_COLUMN
ALL_DIM_LEVELS	ALL_MVIEW_JOINS	ALL_REPCAT
ALL_DIRECTORIES	ALL_MVIEW_KEYS	ALL_REPCATALOG
ALL_ERRORS	ALL_MVIEW_LOGS	ALL_REPCOLUMN
ALL_EVALUATION_CONTEXTS	ALL_MVIEW_REFRESH_TIMES	ALL_REPCOLUMN_GROUP
ALL_EVALUATION_CONTEXT_TABLES	ALL_MVIEWS	ALL_REPCONFLICT
ALL_EVALUATION_CONTEXT_VARS	ALL_NESTED_TABLE_COLS	ALL_REPDDL
ALL_EXTERNAL_LOCATIONS	ALL_NESTED_TABLES	ALL_REPFLAVOR_COLUMNS
ALL_EXTERNAL_TABLES	ALL_OBJ_COLATTRS	ALL_REPFLAVOR_OBJECTS
ALL_HISTOGRAMS	ALL_OBJECTS	ALL_REPFLAVORS
ALL_IND_COLUMNS	ALL_OBJECT_TABLES	ALL_REPGENERATED
ALL_INDEXES	ALL_OLAP_ALTER_SESSION	ALL_REPGENOBJECTS
ALL_IND_EXPRESSIONS	ALL_OPANCILLARY	ALL_REPGROUP
ALL_INDEXTYPE_ARRAYTYPES	ALL_OPARGUMENTS	ALL_REPGROUPED_COLUMN
ALL_INDEXTYPE_COMMENTS	ALL_OPBINDINGS	ALL_REPGROUP_PRIVILEGES
ALL_INDEXTYPE_OPERATORS	ALL_OPERATOR_COMMENTS	ALL_REPKEY_COLUMNS
ALL_INDEXTYPES	ALL_OPERATORS	ALL_REPOBJECT
ALL_IND_PARTITIONS	ALL_OUTLINE_HINTS	ALL_REPPARAMETER_COLUMN
ALL_IND_STATISTICS	ALL_OUTLINES	ALL_REPPRIORITY
ALL_IND_SUBPARTITIONS	ALL_PART_COL_STATISTICS	ALL_REPPRIORITY_GROUP
ALL_INTERNAL_TRIGGERS	ALL_PART_HISTOGRAMS	ALL_REPPROP
ALL_JAVA_ARGUMENTS	ALL_PARTIAL_DROP_TABS	ALL_REPRESOL_STATS_CONTROL
ALL_JAVA_CLASSES	ALL_PART_INDEXES	ALL_REPRESOLUTION
ALL_JAVA_DERIVATIONS	ALL_PART_KEY_COLUMNS	ALL_REPRESOLUTION_METHOD
ALL_JAVA_FIELDS	ALL_PART_LOBS	ALL_REPRESOLUTION_STATISTICS
ALL_JAVA_IMPLEMENTES	ALL_PART_TABLES	ALL_REPSHEMA
ALL_JAVA_INNERS	ALL_PENDING_CONV_TABLES	ALL_REPSITES
ALL_JAVA_LAYOUTS	ALL_PLSQL_OBJECT_SETTINGS	ALL_REWRITE_EQUIVALENCES
ALL_JAVA_METHODS	ALL_POLICIES	ALL_RULES
ALL_JAVA_NCOMPS	ALL_POLICY_CONTEXTS	ALL_RULE_SET_RULES
ALL_JAVA_RESOLVERS	ALL_POLICY_GROUPS	ALL_RULE_SETS
ALL_JAVA_THROWS	ALL_PROBE_OBJECTS	ALL_RULESETS

ALL_SCHEDULER_GLOBAL_ATTRIBUTE	ALL_TAB_COLS	DBA_ADVISOR_JOURNAL
ALL_SCHEDULER_JOB_ARGS	ALL_TAB_COL_STATISTICS	DBA_ADVISOR_LOG
ALL_SCHEDULER_JOB_CLASSES	ALL_TAB_COLUMNS	DBA_ADVISOR_OBJECTS
ALL_SCHEDULER_JOB_LOG	ALL_TAB_COMMENTS	DBA_ADVISOR_OBJECT_TYPES
ALL_SCHEDULER_JOB_RUN_DETAILS	ALL_TAB_HISTOGRAMS	DBA_ADVISOR_PARAMETERS
ALL_SCHEDULER_JOBS	ALL_TABLES	DBA_ADVISOR_PARAMETERS_PROJ
ALL_SCHEDULER_PROGRAM_ARGS	ALL_TAB_MODIFICATIONS	DBA_ADVISOR_RATIONALE
ALL_SCHEDULER_PROGRAMS	ALL_TAB_PARTITIONS	DBA_ADVISOR_RECOMMENDATIONS
ALL_SCHEDULER_SCHEDULES	ALL_TAB_PRIVS	DBA_ADVISOR_SQLA_REC_SUM
ALL_SCHEDULER_WINDOW_DETAILS	ALL_TAB_PRIVS_MADE	DBA_ADVISOR_SQLA_WK_MAP
ALL_SCHEDULER_WINDOW_GROUPS	ALL_TAB_PRIVS_RECD	DBA_ADVISOR_SQLA_WK_STMTS
ALL_SCHEDULER_WINDOW_LOG	ALL_TAB_STATISTICS	DBA_ADVISOR_SQLW_COLVOL
ALL_SCHEDULER_WINDOWS	ALL_TAB_STATS_HISTORY	DBA_ADVISOR_SQLW_JOURNAL
ALL_SCHEDULER_WINGROUP_MEMBERS	ALL_TAB_SUBPARTITIONS	DBA_ADVISOR_SQLW_PARAMETERS
ALL_SECONDARY_OBJECTS	ALL_TRIGGER_COLS	DBA_ADVISOR_SQLW_STMTS
ALL_SEC_RELEVANT_COLS	ALL_TRIGGERS	DBA_ADVISOR_SQLW_SUM
ALL_SEQUENCES	ALL_TYPE_ATTRS	DBA_ADVISOR_SQLW_TABLES
ALL_SERVICES	ALL_TYPE_METHODS	DBA_ADVISOR_SQLW_TABVOL
ALL_SNAPSHOT_LOGS	ALL_TYPES	DBA_ADVISOR_SQLW_TEMPLATES
ALL_SNAPSHOT_REFRESH_TIMES	ALL_TYPE_VERSIONS	DBA_ADVISOR_TASKS
ALL_SNAPSHOTS	ALL_UNUSED_COL_TABS	DBA_ADVISOR_TEMPLATES
ALL_SOURCE	ALL_UPDATABLE_COLUMNS	DBA_ADVISOR_USAGE
ALL_SOURCE_TABLES	ALL_USERS	DBA_ALERT_ARGUMENTS
ALL_SQLJ_TYPE_ATTRS	ALL_USTATS	DBA_ALERT_HISTORY
ALL_SQLJ_TYPE_METHODS	ALL_VARRAYS	DBA_ALL_TABLES
ALL_SQLJ_TYPES	ALL_VIEWS	DBA_ANALYZE_OBJECTS
ALL_STORED_SETTINGS	ALL_WARNING_SETTINGS	DBA_APPLY
ALL_STREAMS_GLOBAL_RULES	ALL_WORKSPACES	DBA_APPLY_CONFLICT_COLUMNS
ALL_STREAMS_MESSAGE_CONSUMERS	ALL_XML_SCHEMAS	DBA_APPLY_DML_HANDLERS
ALL_STREAMS_MESSAGE_RULES	ALL_XML_SCHEMAS2	DBA_APPLY_ENQUEUE
ALL_STREAMS_NEWLY_SUPPORTED	ALL_XML_TAB_COLS	DBA_APPLY_ERROR
ALL_STREAMS_RULES	ALL_XML_TABLES	DBA_APPLY_EXECUTE
ALL_STREAMS_SCHEMA_RULES	ALL_XML_VIEW_COLS	DBA_APPLY_INSTANTIATED_GLOBAL
ALL_STREAMS_TABLE_RULES	ALL_XML_VIEWS	DBA_APPLY_INSTANTIATED_OBJECTS
ALL_STREAMS_TRANSFORM_FUNCTION	AUDIT_ACTIONS	DBA_APPLY_INSTANTIATED_SCHEMAS
ALL_STREAMS_UNSUPPORTED	CAT	DBA_APPLY_KEY_COLUMNS
ALL_SUBPART_COL_STATISTICS	CLU	DBA_APPLY_PARAMETERS
ALL_SUBPART_HISTOGRAMS	COLS	DBA_APPLY_PROGRESS
ALL_SUBPARTITION_TEMPLATES	COLUMN_PRIVILEGES	DBA_APPLY_TABLE_COLUMNS
ALL_SUBPART_KEY_COLUMNS	DATABASE_COMPATIBLE_LEVEL	DBA_AQ_AGENT_PRIVS
ALL_SUBSCRIBED_COLUMNS	DBA_ADVISOR_ACTIONS	DBA_AQ_AGENTS
ALL_SUBSCRIBED_TABLES	DBA_ADVISOR_COMMANDS	DBA_ASSOCIATIONS
ALL_SUBSCRIPTIONS	DBA_ADVISOR_DEFINITIONS	DBA_ATTRIBUTE_TRANSFORMATIONS
ALL_SUMDELTA	DBA_ADVISOR_DEF_PARAMETERS	DBA_AUDIT_EXISTS
ALL_SUMMARIES	DBA_ADVISOR_DIRECTIVES	DBA_AUDIT_OBJECT
ALL_SYNONYMS	DBA_ADVISOR_FINDINGS	DBA_AUDIT_POLICIES

DBA_AUDIT_POLICY_COLUMNS	DBA_DIMENSIONS	DBA_HIST_JAVA_POOL_ADVICE
DBA_AUDIT_SESSION	DBA_DIM_HIERARCHIES	DBA_HIST_LATCH
DBA_AUDIT_STATEMENT	DBA_DIM_JOIN_KEY	DBA_HIST_LATCH_CHILDREN
DBA_AUDIT_TRAIL	DBA_DIM_LEVEL_KEY	DBA_HIST_LATCH_MISSES_SUMMARY
DBA_AW_OBJ	DBA_DIM_LEVELS	DBA_HIST_LATCH_NAME
DBA_AW_PROP	DBA_DIR_DATABASE_ATTRIBUTES	DBA_HIST_LATCH_PARENT
DBA_AW_PS	DBA_DIRECTORIES	DBA_HIST_LIBRARYCACHE
DBA_AWS	DBA_DIR_VICTIM_POLICY	DBA_HIST_LOG
DBA_BASE_TABLE_MVIEWS	DBA_DML_LOCKS	DBA_HIST_METRIC_NAME
DBA_BLOCKERS	DBA_DMT_FREE_SPACE	DBA_HIST_MTTT_TARGET_ADVICE
DBA_CACHEABLE_NONTABLE_OBJECTS	DBA_DMT_USED_EXTENTS	DBA_HISTOGRAMS
DBA_CACHEABLE_OBJECTS	DBA_ENABLED_AGGREGATIONS	DBA_HIST_OPTIMIZER_ENV
DBA_CACHEABLE_OBJECTS_BASE	DBA_ENABLED_TRACES	DBA_HIST_OSSTAT
DBA_CACHEABLE_TABLES	DBA_ERRORS	DBA_HIST_OSSTAT_NAME
DBA_CACHEABLE_TABLES_BASE	DBA_EVALUATION_CONTEXTS	DBA_HIST_PARAMETER
DBA_CAPTURE	DBA_EVALUATION_CONTEXT_TABLES	DBA_HIST_PARAMETER_NAME
DBA_CAPTURE_EXTRA_ATTRIBUTES	DBA_EVALUATION_CONTEXT_VARS	DBA_HIST_PGASTAT
DBA_CAPTURE_PARAMETERS	DBA_EXP_FILES	DBA_HIST_PGA_TARGET_ADVICE
DBA_CAPTURE_PREPARED_DATABASE	DBA_EXP_OBJECTS	DBA_HIST_RESOURCE_LIMIT
DBA_CAPTURE_PREPARED_SCHEMAS	DBA_EXPORT_OBJECTS	DBA_HIST_ROWCACHE_SUMMARY
DBA_CAPTURE_PREPARED_TABLES	DBA_EXP_VERSION	DBA_HIST_SEG_STAT
DBA_CATALOG	DBA_EXTENTS	DBA_HIST_SEG_STAT_OBJ
DBA_CL_DIR_INSTANCE_ACTIONS	DBA_EXTERNAL_LOCATIONS	DBA_HIST_SERVICE_NAME
DBA_CLU_COLUMNS	DBA_EXTERNAL_TABLES	DBA_HIST_SERVICE_STAT
DBA_CLUSTER_HASH_EXPRESSIONS	DBA_FEATURE_USAGE_STATISTICS	DBA_HIST_SERVICE_WAIT_CLASS
DBA_CLUSTERS	DBA_FGA_AUDIT_TRAIL	DBA_HIST_SESSMETRIC_HISTORY
DBA_COL_COMMENTS	DBA_FREE_SPACE	DBA_HIST_SGA
DBA_COLL_TYPES	DBA_FREE_SPACE_COALESCED	DBA_HIST_SGASTAT
DBA_COL_PRIVS	DBA_GLOBAL_CONTEXT	DBA_HIST_SHARED_POOL_ADVICE
DBA_COMMON_AUDIT_TRAIL	DBA_HIGH_WATER_MARK_STATISTICS	DBA_HIST_SNAP_ERROR
DBA_CONS_COLUMNS	DBA_HIST_ACTIVE_SESS_HISTORY	DBA_HIST_SNAPSHOT
DBA_CONS_OBJ_COLUMNS	DBA_HIST_BASELINE	DBA_HIST_SQLBIND
DBA_CONSTRAINTS	DBA_HIST_BG_EVENT_SUMMARY	DBA_HIST_SQL_PLAN
DBA_CONTEXT	DBA_HIST_BUFFER_POOL_STAT	DBA_HIST_SQLSTAT
DBA_DATA_FILES	DBA_HIST_CLASS_CACHE_TRANSFER	DBA_HIST_SQL_SUMMARY
DBA_DATAPUMP_JOBS	DBA_HIST_CR_BLOCK_SERVER	DBA_HIST_SQLTEXT
DBA_DATAPUMP_SESSIONS	DBA_HIST_CURRENT_BLOCK_SERVER	DBA_HIST_SQL_WORKAREA_HSTGRM
DBA_DB_DIR_ESCALATE_ACTIONS	DBA_HIST_DATABASE_INSTANCE	DBA_HIST_STAT_NAME
DBA_DB_DIR QUIESCE_ACTIONS	DBA_HIST_DATAFILE	DBA_HIST_SYSMETRIC_HISTORY
DBA_DB_DIR_SERVICE_ACTIONS	DBA_HIST_DB_CACHE_ADVICE	DBA_HIST_SYSMETRIC_SUMMARY
DBA_DB_DIR_SESSION_ACTIONS	DBA_HIST_DLM_MISC	DBA_HIST_SYSSTAT
DBA_DB_LINKS	DBA_HIST_ENQUEUE_STAT	DBA_HIST_SYSTEM_EVENT
DBA_DDL_LOCKS	DBA_HIST_EVENT_NAME	DBA_HIST_SYS_TIME_MODEL
DBA_DEPENDENCIES	DBA_HIST_FILEMETRIC_HISTORY	DBA_HIST_TABLESPACE_STAT
DBA_DIM_ATTRIBUTES	DBA_HIST_FILESTATXS	DBA_HIST_TBSPC_SPACE_USAGE
DBA_DIM_CHILD_OF	DBA_HIST_INSTANCE_RECOVERY	DBA_HIST_TEMPFILE

DBA_HIST_TEMPSTATXS	DBA_LOB_PARTITIONS	DBA_OUTLINE_HINTS
DBA_HIST_THREAD	DBA_LOBS	DBA_OUTLINES
DBA_HIST_UNDOSTAT	DBA_LOB_SUBPARTITIONS	DBA_OUTSTANDING_ALERTS
DBA_HIST_WAITCLASSMET_HISTORY	DBA_LOB_TEMPLATES	DBA_PART_COL_STATISTICS
DBA_HIST_WAITSTAT	DBA_LOCK	DBA_PART_HISTOGRAMS
DBA_HIST_WR_CONTROL	DBA_LOCK_INTERNAL	DBA_PARTIAL_DROP_TABS
DBA_IAS_CONSTRAINT_EXP	DBA_LOCKS	DBA_PART_INDEXES
DBA_IAS_GEN_STMTS	DBA_LOG_GROUP_COLUMNS	DBA_PART_KEY_COLUMNS
DBA_IAS_GEN_STMTS_EXP	DBA_LOG_GROUPS	DBA_PART_LOBS
DBA_IAS_OBJECTS	DBA_LOGMNR_LOG	DBA_PART_TABLES
DBA_IAS_OBJECTS_BASE	DBA_LOGMNR_PURGED_LOG	DBA_PENDING_CONV_TABLES
DBA_IAS_OBJECTS_EXP	DBA_LOGMNR_SESSION	DBA_PENDING_TRANSACTIONS
DBA_IAS_POSTGEN_STMTS	DBA_LOGSTDBY_EVENTS	DBA_PLSQL_OBJECT_SETTINGS
DBA_IAS_PREGEN_STMTS	DBA_LOGSTDBY_HISTORY	DBA_POLICIES
DBA_IAS_SITES	DBA_LOGSTDBY_LOG	DBA_POLICY_CONTEXTS
DBA_IAS_TEMPLATES	DBA_LOGSTDBY_NOT_UNIQUE	DBA_POLICY_GROUPS
DBA_IND_COLUMNS	DBA_LOGSTDBY_PARAMETERS	DBA_PRIV_AUDIT_OPTS
DBA_INDEXES	DBA_LOGSTDBY_PROGRESS	DBA_PROCEEDURES
DBA_IND_EXPRESSIONS	DBA_LOGSTDBY_SKIP	DBA_PROFILES
DBA_INDEXTYPE_ARRAYTYPES	DBA_LOGSTDBY_SKIP_TRANSACTION	DBA_PROPAGATION
DBA_INDEXTYPE_COMMENTS	DBA_LOGSTDBY_UNSUPPORTED	DBA_PROXIES
DBA_INDEXTYPE_OPERATORS	DBA_METHOD_PARAMS	DBA_PUBLISHED_COLUMNS
DBA_INDEXTYPES	DBA_METHOD_RESULTS	DBA_QUEUE_PUBLISHERS
DBA_IND_PARTITIONS	DBA_MVIEW_AGGREGATES	DBA_QUEUES
DBA_IND_STATISTICS	DBA_MVIEW_ANALYSIS	DBA_QUEUE_SCHEDULES
DBA_IND_SUBPARTITIONS	DBA_MVIEW_COMMENTS	DBA_QUEUE_TABLES
DBA_INTERNAL_TRIGGERS	DBA_MVIEW_DETAIL_RELATIONS	DBA_RCHILD
DBA_JAVA_ARGUMENTS	DBA_MVIEW_JOINS	DBA_RECYCLEBIN
DBA_JAVA_CLASSES	DBA_MVIEW_KEYS	DBA_REFRESH
DBA_JAVA_DERIVATIONS	DBA_MVIEW_LOG_FILTER_COLS	DBA_REFRESH_CHILDREN
DBA_JAVA_FIELDS	DBA_MVIEW_LOGS	DBA_REFS
DBA_JAVA_IMPLEMENTES	DBA_MVIEW_REFRESH_TIMES	DBA_REGISTERED_ARCHIVED_LOG
DBA_JAVA_INNERS	DBA_MVIEWS	DBA_REGISTERED_MVIEW_GROUPS
DBA_JAVA_LAYOUTS	DBA_NESTED_TABLE_COLS	DBA_REGISTERED_MVIEWS
DBA_JAVA_METHODS	DBA_NESTED_TABLES	DBA_REGISTERED_SNAPSHOT_GROUPS
DBA_JAVA_NCOMPS	DBA_OBJ_AUDIT_OPTS	DBA_REGISTERED_SNAPSHOTS
DBA_JAVA_POLICY	DBA_OBJ_COLATTRS	DBA_REGISTRY
DBA_JAVA_RESOLVERS	DBA_OBJECTS	DBA_REGISTRY_HIERARCHY
DBA_JAVA_THROWS	DBA_OBJECT_SIZE	DBA_REPAUDIT_ATTRIBUTE
DBA_JOBS	DBA_OBJECT_TABLES	DBA_REPAUDIT_COLUMN
DBA_JOBS_RUNNING	DBA_OPANCILLARY	DBA_REPCAT
DBA_JOIN_IND_COLUMNS	DBA_OPARGUMENTS	DBA_REPCAT_EXCEPTIONS
DBA_KGLLOCK	DBA_OPBINDINGS	DBA_REPCATALOG
DBA_LIBRARIES	DBA_OPERATOR_COMMENTS	DBA_REPCOLUMN
DBA_LMT_FREE_SPACE	DBA_OPERATORS	DBA_REPCOLUMN_GROUP
DBA_LMT_USED_EXTENTS	DBA_OPTSTAT_OPERATIONS	DBA_REPCONFLICT

DBA_REPDDL	DBA_SCHEDULER_PROGRAM_ARGS	DBA_SUBPARTITION_TEMPLATES
DBA_REPEXTENSIONS	DBA_SCHEDULER_PROGRAMS	DBA_SUBPART_KEY_COLUMNS
DBA_REPFLAVOR_COLUMNS	DBA_SCHEDULER_SCHEDULES	DBA_SUBSCRIBED_COLUMNS
DBA_REPFLAVOR_OBJECTS	DBA_SCHEDULER_WINDOW_DETAILS	DBA_SUBSCRIBED_TABLES
DBA_REPFLAVORS	DBA_SCHEDULER_WINDOW_GROUPS	DBA_SUBSCRIPTIONS
DBA_REPGENERATED	DBA_SCHEDULER_WINDOW_LOG	DBA_SUMMARIES
DBA_REPGENOBJECTS	DBA_SCHEDULER_WINDOWS	DBA_SYNONYMS
DBA_REPGROUP	DBA_SCHEDULER_WINGROUP_MEMBERS	DBA_SYS_PRIVS
DBA_REPGROUPED_COLUMN	DBA_SECONDARY_OBJECTS	DBA_TAB_COLS
DBA_REPGROUP_PRIVILEGES	DBA_SEC_RELEVANT_COLS	DBA_TAB_COL_STATISTICS
DBA_REPKEY_COLUMNS	DBA_SEGMENTS	DBA_TAB_COLUMNS
DBA_REPOBJECT	DBA_SEGMENTS_OLD	DBA_TAB_COMMENTS
DBA_REPPARAMETER_COLUMN	DBA_SEQUENCES	DBA_TAB_HISTOGRAMS
DBA_REPPRIORITY	DBA_SERVER_REGISTRY	DBA_TABLES
DBA_REPPRIORITY_GROUP	DBA_SNAPSHOT_LOG_FILTER_COLS	DBA_TABLESPACE_GROUPS
DBA_REPPROP	DBA_SNAPSHOT_LOGS	DBA_TABLESPACES
DBA_REPRESOL_STATS_CONTROL	DBA_SNAPSHOT_REFRESH_TIMES	DBA_TABLESPACE_USAGE_METRICS
DBA_REPRESOLUTION	DBA_SNAPSHOTS	DBA_TAB_MODIFICATIONS
DBA_REPRESOLUTION_METHOD	DBA_SOURCE	DBA_TAB_PARTITIONS
DBA_REPRESOLUTION_STATISTICS	DBA_SOURCE_TABLES	DBA_TAB_PRIVS
DBA_REPSHEMA	DBA_SQLJ_TYPE_ATTRS	DBA_TAB_STATISTICS
DBA_REPSITES	DBA_SQLJ_TYPE_METHODS	DBA_TAB_STATS_HISTORY
DBA_REPSITES_NEW	DBA_SQLJ_TYPES	DBA_TAB_SUBPARTITIONS
DBA_RESUMABLE	DBA_SQLSET	DBA_TEMP_FILES
DBA_REWRITE_EQUIVALENCES	DBA_SQLSET_BINDS	DBA_TEMPLATE_REFGROUPS
DBA_RGROUP	DBA_SQLSET_DEFINITIONS	DBA_TEMPLATE_TARGETS
DBA_ROLE_PRIVS	DBA_SQLSET_REFERENCES	DBA_THRESHOLDS
DBA_ROLES	DBA_SQLSET_STATEMENTS	DBA_TRANSFORMATIONS
DBA_ROLLBACK_SEGS	DBA_SQLTUNE_BINDS	DBA_TRIGGER_COLS
DBA_RSRC_CONSUMER_GROUP_PRIVS	DBA_SQLTUNE_PLANS	DBA_TRIGGERS
DBA_RSRC_CONSUMER_GROUPS	DBA_SQLTUNE_RATIONALE_PLAN	DBA_TSM_HISTORY
DBA_RSRC_GROUP_MAPPINGS	DBA_SQLTUNE_STATISTICS	DBA_TS_QUOTAS
DBA_RSRC_MANAGER_SYSTEM_PRIVS	DBA_STMT_AUDIT_OPTS	DBA_TUNE_MVIEW
DBA_RSRC_MAPPING_PRIORITY	DBA_STORED_SETTINGS	DBA_TYPE_ATTRS
DBA_RSRC_PLAN_DIRECTIVES	DBA_STREAMS_ADMINISTRATOR	DBA_TYPE_METHODS
DBA_RSRC_PLANS	DBA_STREAMS_GLOBAL_RULES	DBA_TYPES
DBA_RULES	DBA_STREAMS_MESSAGE_CONSUMERS	DBA_TYPE_VERSIONS
DBA_RULE_SET_RULES	DBA_STREAMS_MESSAGE_RULES	DBA_UNDO_EXTENTS
DBA_RULE_SETS	DBA_STREAMS_NEWLY_SUPPORTED	DBA_UNUSED_COL_TABS
DBA_RULESETS	DBA_STREAMS_RULES	DBA_UPDATABLE_COLUMNS
DBA_SCHEDULER_GLOBAL_ATTRIBUTE	DBA_STREAMS_SCHEMA_RULES	DBA_USERS
DBA_SCHEDULER_JOB_ARGS	DBA_STREAMS_TABLE_RULES	DBA_USTATS
DBA_SCHEDULER_JOB_CLASSES	DBA_STREAMS_TRANSFORM_FUNCTION	DBA_VARRAYS
DBA_SCHEDULER_JOB_LOG	DBA_STREAMS_UNSUPPORTED	DBA_VIEWS
DBA_SCHEDULER_JOB_RUN_DETAILS	DBA_SUBPART_COL_STATISTICS	DBA_WAITERS
DBA_SCHEDULER_JOBS	DBA_SUBPART_HISTOGRAMS	DBA_WARNING_SETTINGS

DBA_WORKSPACES	GV\$BACKUP_CORRUPTION	GV\$DLM_CONVERT_LOCAL
DBA_WORKSPACE_SESSIONS	GV\$BACKUP_DATAFILE	GV\$DLM_CONVERT_REMOTE
DBA_XML_SCHEMAS	GV\$BACKUP_DEVICE	GV\$DLM_LATCH
DBA_XML_TAB_COLS	GV\$BACKUP_PIECE	GV\$DLM_LOCKS
DBA_XML_TABLES	GV\$BACKUP_REDOLOG	GV\$DLM_MISC
DBA_XML_VIEW_COLS	GV\$BACKUP_SET	GV\$DLM_RESS
DBA_XML_VIEWS	GV\$BACKUP_SFILE	GV\$DLM_TRAFFIC_CONTROLLER
DBA_2PC_NEIGHBORS	GV\$BACKUP_SYNC_IO	GV\$ENABLEDPRIVS
DBA_2PC_PENDING	GV\$BGPROCESS	GV\$ENQUEUE_LOCK
DBMS_ALERT_INFO	GV\$BH	GV\$ENQUEUE_STAT
DBMS_LOCK_ALLOCATED	GV\$BSP	GV\$ENQUEUE_STATISTICS
DICT	GV\$BUFFERED_PUBLISHERS	GV\$EVENT_HISTOGRAM
DICT_COLUMNS	GV\$BUFFERED_QUEUES	GV\$EVENTMETRIC
DICTIONARY	GV\$BUFFERED_SUBSCRIBERS	GV\$EVENT_NAME
DUAL	GV\$BUFFER_POOL	GV\$EXECUTION
GLOBAL_NAME	GV\$BUFFER_POOL_STATISTICS	GV\$FAST_START_SERVERS
GV\$ACCESS	GV\$CIRCUIT	GV\$FAST_START_TRANSACTIONS
GV\$ACTIVE_INSTANCES	GV\$CLASS_CACHE_TRANSFER	GV\$FILE_CACHE_TRANSFER
GV\$ACTIVE_SERVICES	GV\$CLASS_PING	GV\$FILE_HISTOGRAM
GV\$ACTIVE_SESSION_HISTORY	GV\$CLIENT_STATS	GV\$FILEMETRIC
GV\$ACTIVE_SESS_POOL_MTH	GV\$CONTEXT	GV\$FILEMETRIC_HISTORY
GV\$ADVISOR_PROGRESS	GV\$CONTROLFILE	GV\$FILE_PING
GV\$ALERT_TYPES	GV\$CONTROLFILE_RECORD_SECTION	GV\$FILESPACE_USAGE
GV\$AQ	GV\$COPY_CORRUPTION	GV\$FILESTAT
GV\$AQ1	GV\$CR_BLOCK_SERVER	GV\$FIXED_TABLE
GV\$ARCHIVE	GV\$CURRENT_BLOCK_SERVER	GV\$FIXED_VIEW_DEFINITION
GV\$ARCHIVE_DEST	GV\$DATABASE	GV\$FLASHBACK_DATABASE_LOG
GV\$ARCHIVE_DEST_STATUS	GV\$DATABASE_BLOCK_CORRUPTION	GV\$FLASHBACK_DATABASE_LOGFILE
GV\$ARCHIVED_LOG	GV\$DATABASE_INCARNATION	GV\$FLASHBACK_DATABASE_STAT
GV\$ARCHIVE_GAP	GV\$DATAFILE	GV\$GC_ELEMENT
GV\$ARCHIVE_PROCESSES	GV\$DATAFILE_COPY	GV\$GC_ELEMENTS_WITH_COLLISIONS
GV\$ASM_ALIAS	GV\$DATAFILE_HEADER	GV\$GCSPFMASTER_INFO
GV\$ASM_CLIENT	GV\$DATAGUARD_CONFIG	GV\$GCSPFMASTER_INFO
GV\$ASM_DISK	GV\$DATAGUARD_STATUS	GV\$GES_BLOCKING_ENQUEUE
GV\$ASM_DISKGROUP	GV\$DATAPUMP_JOB	GV\$GES_ENQUEUE
GV\$ASM_FILE	GV\$DATAPUMP_SESSION	GV\$GLOBAL_BLOCKED_LOCKS
GV\$ASM_OPERATION	GV\$DB_CACHE_ADVICE	GV\$GLOBALCONTEXT
GV\$ASM_TEMPLATE	GV\$DBFILE	GV\$GLOBAL_TRANSACTION
GV\$AW_AGGREGATE_OP	GV\$DBLINK	GV\$HS_AGENT
GV\$AW_ALLOCATE_OP	GV\$DB_OBJECT_CACHE	GV\$HS_PARAMETER
GV\$AW_CALC	GV\$DB_PIPES	GV\$HS_SESSION
GV\$AW_LONGOPS	GV\$DELETED_OBJECT	GV\$HVMMASTER_INFO
GV\$AW_OLAP	GV\$DISPATCHER	GV\$INDEXED_FIXED_COLUMN
GV\$AW_SESSION_INFO	GV\$DISPATCHER_CONFIG	GV\$INSTANCE
GV\$BACKUP	GV\$DISPATCHER_RATE	GV\$INSTANCE_CACHE_TRANSFER
GV\$BACKUP_ASYNC_IO	GV\$DLM_ALL_LOCKS	GV\$INSTANCE_LOG_GROUP

GV\$INSTANCE_RECOVERY	GV\$MAP_FILE_IO_STACK	GV\$RECOVERY_STATUS
GV\$JAVA_LIBRARY_CACHE_MEMORY	GV\$MAP_LIBRARY	GV\$REPLPROP
GV\$JAVAPOL	GV\$MAP_SUBELEMENT	GV\$REPLQUEUE
GV\$JAVA_POOL_ADVICE	GV\$MAX_ACTIVE_SESS_TARGET_MTH	GV\$REQDIST
GV\$LATCH	GV\$METRIC	GV\$RESERVED_WORDS
GV\$LATCH_CHILDREN	GV\$METRICGROUP	GV\$RESOURCE
GV\$LATCHHOLDER	GV\$METRIC_HISTORY	GV\$RESOURCE_LIMIT
GV\$LATCH_MISSES	GV\$METRICNAME	GV\$RESUMABLE
GV\$LATCHNAME	GV\$MTR_TARGET_ADVICE	GV\$RMAN_CONFIGURATION
GV\$LATCH_PARENT	GV\$MVREFRESH	GV\$RMAN_OUTPUT
GV\$LIBRARYCACHE	GV\$MYSTAT	GV\$ROLLSTAT
GV\$LIBRARY_CACHE_MEMORY	GV\$NLS_PARAMETERS	GV\$ROWCACHE
GV\$LICENSE	GV\$NLS_VALID_VALUES	GV\$ROWCACHE_PARENT
GV\$LOADISTAT	GV\$OBJECT_DEPENDENCY	GV\$ROWCACHE_SUBORDINATE
GV\$LOADPSTAT	GV\$OBSOLETE_PARAMETER	GV\$RSRC_CONSUMER_GROUP
GV\$_LOCK	GV\$OFFLINE_RANGE	GV\$RSRC_CONSUMER_GROUP_CPU_MTH
GV\$LOCK	GV\$OPEN_CURSOR	GV\$RSRC_PLAN
GV\$LOCK_ACTIVITY	GV\$OPTION	GV\$RSRC_PLAN_CPU_MTH
GV\$LOCKED_OBJECT	GV\$OSSTAT	GV\$RULE
GV\$LOCK_ELEMENT	GV\$PARALLEL_DEGREE_LIMIT_MTH	GV\$RULE_SET
GV\$LOCKS_WITH_COLLISIONS	GV\$PARAMETER	GV\$RULE_SET_AGGREGATE_STATS
GV\$LOCK_TYPE	GV\$PARAMETER2	GV\$SCHEDULER_RUNNING_JOBS
GV\$LOG	GV\$PGASTAT	GV\$SEGMENT_STATISTICS
GV\$LOGFILE	GV\$PGA_TARGET_ADVICE	GV\$SEGSTAT
GV\$LOGHIST	GV\$PGA_TARGET_ADVICE_HISTOGRAM	GV\$SEGSTAT_NAME
GV\$LOG_HISTORY	GV\$PQ_SESSTAT	GV\$SERVICE_EVENT
GV\$LOGMNR_CALLBACK	GV\$PQ_SLAVE	GV\$SERVICEMETRIC
GV\$LOGMNR_CONTENTS	GV\$PQ_SYSSTAT	GV\$SERVICEMETRIC_HISTORY
GV\$LOGMNR_DICTIONARY	GV\$PQ_TQSTAT	GV\$SERVICES
GV\$LOGMNR_LATCH	GV\$PROCESS	GV\$SERVICE_STATS
GV\$LOGMNR_LOGFILE	GV\$PROPAGATION_RECEIVER	GV\$SERVICE_WAIT_CLASS
GV\$LOGMNR_LOGS	GV\$PROPAGATION_SENDER	GV\$SERV_MOD_ACT_STATS
GV\$LOGMNR_PARAMETERS	GV\$PROXY_ARCHIVEDLOG	GV\$SES_OPTIMIZER_ENV
GV\$LOGMNR_PROCESS	GV\$PROXY_DATAFILE	GV\$SESS_IO
GV\$LOGMNR_REGION	GV\$PWFILERS_USERS	GV\$SESSION
GV\$LOGMNR_SESSION	GV\$PX_BUFFER_ADVICE	GV\$SESSION_CONNECT_INFO
GV\$LOGMNR_STATS	GV\$PX_PROCESS	GV\$SESSION_CURSOR_CACHE
GV\$LOGMNR_TRANSACTION	GV\$PX_PROCESS_SYSSTAT	GV\$SESSION_EVENT
GV\$LOGSTDBY	GV\$PX_SESSION	GV\$SESSION_LONGOPS
GV\$LOGSTDBY_STATS	GV\$PX_SESSTAT	GV\$SESSION_OBJECT_CACHE
GV\$MANAGED_STANDBY	GV\$QUEUE	GV\$SESSION_WAIT
GV\$MAP_COMP_LIST	GV\$QUEUEING_MTH	GV\$SESSION_WAIT_CLASS
GV\$MAP_ELEMENT	GV\$RECOVER_FILE	GV\$SESSION_WAIT_HISTORY
GV\$MAP_EXT_ELEMENT	GV\$RECOVERY_FILE_STATUS	GV\$SESSMETRIC
GV\$MAP_FILE	GV\$RECOVERY_LOG	GV\$SESSTAT
GV\$MAP_FILE_EXTENT	GV\$RECOVERY_PROGRESS	GV\$SESS_TIME_MODEL

GV\$SGA	GV\$SYSTEM_CURSOR_CACHE	SM\$VERSION
GV\$SGA_CURRENT_RESIZE_OPS	GV\$SYSTEM_EVENT	SYN
GV\$SGA_DYNAMIC_COMPONENTS	GV\$SYSTEM_PARAMETER	TABLE_PRIVILEGES
GV\$SGA_DYNAMIC_FREE_MEMORY	GV\$SYSTEM_PARAMETER2	TABS
GV\$SGAINFO	GV\$SYSTEM_WAIT_CLASS	USER_ADVISOR_ACTIONS
GV\$SGA_RESIZE_OPS	GV\$SYS_TIME_MODEL	USER_ADVISOR_DIRECTIVES
GV\$SGASTAT	GV\$TABLESPACE	USER_ADVISOR_FINDINGS
GV\$SHARED_POOL_ADVICE	GV\$TEMP_CACHE_TRANSFER	USER_ADVISOR_JOURNAL
GV\$SHARED_POOL_RESERVED	GV\$TEMP_EXTENT_MAP	USER_ADVISOR_LOG
GV\$SHARED_SERVER	GV\$TEMP_EXTENT_POOL	USER_ADVISOR_OBJECTS
GV\$SHARED_SERVER_MONITOR	GV\$TEMPFILE	USER_ADVISOR_PARAMETERS
GV\$SORT_SEGMENT	GV\$TEMP_HISTOGRAM	USER_ADVISOR_RATIONALE
GV\$SORT_USAGE	GV\$TEMPORARY_LOBS	USER_ADVISOR_RECOMMENDATIONS
GV\$SPPARAMETER	GV\$TEMP_PING	USER_ADVISOR_SQLA_REC_SUM
GV\$SQL	GV\$TEMPSEG_USAGE	USER_ADVISOR_SQLA_WK_MAP
GV\$SQLAREA	GV\$TEMP_SPACE_HEADER	USER_ADVISOR_SQLA_WK_STMTS
GV\$SQL_BIND_CAPTURE	GV\$TEMPSTAT	USER_ADVISOR_SQLW_COLVOL
GV\$SQL_BIND_DATA	GV\$THREAD	USER_ADVISOR_SQLW_JOURNAL
GV\$SQL_BIND_METADATA	GV\$THRESHOLD_TYPES	USER_ADVISOR_SQLW_PARAMETERS
GV\$SQL_CURSOR	GV\$TIMER	USER_ADVISOR_SQLW_STMTS
GV\$SQL_OPTIMIZER_ENV	GV\$TIMEZONE_FILE	USER_ADVISOR_SQLW_SUM
GV\$SQL_PLAN	GV\$TIMEZONE_NAMES	USER_ADVISOR_SQLW_TABLES
GV\$SQL_PLAN_STATISTICS	GV\$TRANSACTION	USER_ADVISOR_SQLW_TABVOL
GV\$SQL_PLAN_STATISTICS_ALL	GV\$TRANSACTION_ENQUEUE	USER_ADVISOR_SQLW_TEMPLATES
GV\$SQL_REDIRECTION	GV\$TYPE_SIZE	USER_ADVISOR_TASKS
GV\$SQL_SHARED_CURSOR	GV\$UNDOSTAT	USER_ADVISOR_TEMPLATES
GV\$SQL_SHARED_MEMORY	GV\$VERSION	USER_ALL_TABLES
GV\$SQLTEXT	GV\$VPD_POLICY	USER_AQ_AGENT_PRIVS
GV\$SQLTEXT_WITH_NEWLINES	GV\$WAITCLASSMETRIC	USER_ARGUMENTS
GV\$SQL_WORKAREA	GV\$WAITCLASSMETRIC_HISTORY	USER_ASSOCIATIONS
GV\$SQL_WORKAREA_ACTIVE	GV\$WAITSTAT	USER_ATTRIBUTE_TRANSFORMATIONS
GV\$SQL_WORKAREA_HISTOGRAM	IND	USER_AUDIT_OBJECT
GV\$STANDBY_LOG	INDEX_HISTOGRAM	USER_AUDIT_POLICIES
GV\$STATISTICS_LEVEL	INDEX_STATS	USER_AUDIT_POLICY_COLUMNS
GV\$STATNAME	NLS_DATABASE_PARAMETERS	USER_AUDIT_SESSION
GV\$STREAMS_APPLY_COORDINATOR	NLS_INSTANCE_PARAMETERS	USER_AUDIT_STATEMENT
GV\$STREAMS_APPLY_READER	NLS_SESSION_PARAMETERS	USER_AUDIT_TRAIL
GV\$STREAMS_APPLY_SERVER	OBJ	USER_AW_OBJ
GV\$STREAMS_CAPTURE	RECYCLEBIN	USER_AW_PROP
GV\$SUBCACHE	RESOURCE_COST	USER_AW_PS
GV\$SYSAUX_OCCUPANTS	ROLE_ROLE_PRIVS	USER_AWS
GV\$SYSMETRIC	ROLE_SYS_PRIVS	USER_BASE_TABLE_MVIEWS
GV\$SYSMETRIC_HISTORY	ROLE_TAB_PRIVS	USER_CATALOG
GV\$SYSMETRIC_SUMMARY	SEQ	USER_CLU_COLUMNS
GV\$SYS_OPTIMIZER_ENV	SESSION_PRIVS	USER_CLUSTER_HASH_EXPRESSIONS
GV\$SYSSTAT	SESSION_ROLES	USER_CLUSTERS

USER_COL_COMMENTS	USER_JAVA_NCOMPS	USER_PENDING_CONV_TABLES
USER_COLL_TYPES	USER_JAVA_POLICY	USER_PLSQL_OBJECT_SETTINGS
USER_COL_PRIVS	USER_JAVA_RESOLVERS	USER_POLICIES
USER_COL_PRIVS_MADE	USER_JAVA_THROWS	USER_POLICY_CONTEXTS
USER_COL_PRIVS_RECD	USER_JOBS	USER_POLICY_GROUPS
USER_CONS_COLUMNS	USER_JOIN_IND_COLUMNS	USER_PROCEDURES
USER_CONS_OBJ_COLUMNS	USER_LIBRARIES	USER_PROXIES
USER_CONSTRAINTS	USER_LOB_PARTITIONS	USER_PUBLISHED_COLUMNS
USER_DATAPUMP_JOBS	USER_LOBS	USER_QUEUE_PUBLISHERS
USER_DB_LINKS	USER_LOB_SUBPARTITIONS	USER_QUEUES
USER_DEPENDENCIES	USER_LOB_TEMPLATES	USER_QUEUE_SCHEDULES
USER_DIM_ATTRIBUTES	USER_LOG_GROUP_COLUMNS	USER_QUEUE_TABLES
USER_DIM_CHILD_OF	USER_LOG_GROUPS	USER_RECYCLEBIN
USER_DIMENSIONS	USER_METHOD_PARAMS	USER_REFRESH
USER_DIM_HIERARCHIES	USER_METHOD_RESULTS	USER_REFRESH_CHILDREN
USER_DIM_JOIN_KEY	USER_MVIEW_AGGREGATES	USER_REFS
USER_DIM_LEVEL_KEY	USER_MVIEW_ANALYSIS	USER_REGISTERED_MVIEWS
USER_DIM_LEVELS	USER_MVIEW_COMMENTS	USER_REGISTERED_SNAPSHOTS
USER_ERRORS	USER_MVIEW_DETAIL_RELATIONS	USER_REGISTRY
USER_EVALUATION_CONTEXTS	USER_MVIEW_JOINS	USER_REPAUDIT_ATTRIBUTE
USER_EVALUATION_CONTEXT_TABLES	USER_MVIEW_KEYS	USER_REPAUDIT_COLUMN
USER_EVALUATION_CONTEXT_VARS	USER_MVIEW_LOGS	USER_REPCAT
USER_EXTENTS	USER_MVIEW_REFRESH_TIMES	USER_REPCATLOG
USER_EXTERNAL_LOCATIONS	USER_MVIEWS	USER_REPCOLUMN
USER_EXTERNAL_TABLES	USER_NESTED_TABLE_COLS	USER_REPCOLUMN_GROUP
USER_FREE_SPACE	USER_NESTED_TABLES	USER_REPCONFLICT
USER_HISTOGRAMS	USER_OBJ_AUDIT_OPTS	USER_REPDDL
USER_IND_COLUMNS	USER_OBJ_COLATTRS	USER_REPFLAVOR_COLUMNS
USER_INDEXES	USER_OBJECTS	USER_REPFLAVOR_OBJECTS
USER_IND_EXPRESSIONS	USER_OBJECT_SIZE	USER_REPFLAVORS
USER_INDEXTYPE_ARRAYTYPES	USER_OBJECT_TABLES	USER_REPGENERATED
USER_INDEXTYPE_COMMENTS	USER_OPANCILLARY	USER_REPGENOBJECTS
USER_INDEXTYPE_OPERATORS	USER_OPARGUMENTS	USER_REPGROUP
USER_INDEXTYPES	USER_OPBINDINGS	USER_REPGROUPED_COLUMN
USER_IND_PARTITIONS	USER_OPERATOR_COMMENTS	USER_REPGROUP_PRIVILEGES
USER_IND_STATISTICS	USER_OPERATORS	USER_REPKEY_COLUMNS
USER_IND_SUBPARTITIONS	USER_OUTLINE_HINTS	USER_REPOBJECT
USER_INTERNAL_TRIGGERS	USER_OUTLINES	USER_REPPARAMETER_COLUMN
USER_JAVA_ARGUMENTS	USER_PART_COL_STATISTICS	USER_REPPRIORITY
USER_JAVA_CLASSES	USER_PART_HISTOGRAMS	USER_REPPRIORITY_GROUP
USER_JAVA_DERIVATIONS	USER_PARTIAL_DROP_TABS	USER_REPPROP
USER_JAVA_FIELDS	USER_PART_INDEXES	USER_REPRESOL_STATS_CONTROL
USER_JAVA_IMPLEMENTES	USER_PART_KEY_COLUMNS	USER_REPRESOLUTION
USER_JAVA_INNERS	USER_PART_LOBS	USER_REPRESOLUTION_METHOD
USER_JAVA_LAYOUTS	USER_PART_TABLES	USER_REPRESOLUTION_STATISTICS
USER_JAVA_METHODS	USER_PASSWORD_LIMITS	USER_REPSHEMA

USER_REPSITES	USER_SUBSCRIPTIONS	V\$ADVISOR_PROGRESS
USER_RESOURCE_LIMITS	USER_SUMMARIES	V\$ALERT_TYPES
USER_RESUMABLE	USER_SYNONYMS	V\$AQ
USER_REWRITE_EQUIVALENCES	USER_SYS_PRIVS	V\$AQ1
USER_ROLE_PRIVS	USER_TAB_COLS	V\$ARCHIVE
USER_RSRC_CONSUMER_GROUP_PRIVS	USER_TAB_COL_STATISTICS	V\$ARCHIVE_DEST
USER_RSRC_MANAGER_SYSTEM_PRIVS	USER_TAB_COLUMNS	V\$ARCHIVE_DEST_STATUS
USER_RULES	USER_TAB_COMMENTS	V\$ARCHIVED_LOG
USER_RULE_SET_RULES	USER_TAB_HISTOGRAMS	V\$ARCHIVE_GAP
USER_RULE_SETS	USER_TABLES	V\$ARCHIVE_PROCESSES
USER_RULESETS	USER_TABLESPACES	V\$ASM_ALIAS
USER_SCHEDULER_JOB_ARGS	USER_TAB_MODIFICATIONS	V\$ASM_CLIENT
USER_SCHEDULER_JOB_LOG	USER_TAB_PARTITIONS	V\$ASM_DISK
USER_SCHEDULER_JOB_RUN_DETAILS	USER_TAB_PRIVS	V\$ASM_DISKGROUP
USER_SCHEDULER_JOBS	USER_TAB_PRIVS_MADE	V\$ASM_FILE
USER_SCHEDULER_PROGRAM_ARGS	USER_TAB_PRIVS_RECD	V\$ASM_OPERATION
USER_SCHEDULER_PROGRAMS	USER_TAB_STATISTICS	V\$ASM_TEMPLATE
USER_SCHEDULER_SCHEDULES	USER_TAB_STATS_HISTORY	V\$AW_AGGREGATE_OP
USER_SECONDARY_OBJECTS	USER_TAB_SUBPARTITIONS	V\$AW_ALLOCATE_OP
USER_SEC_RELEVANT_COLS	USER_TRANSFORMATIONS	V\$AW_CALC
USER_SEGMENTS	USER_TRIGGER_COLS	V\$AW_LONGOPS
USER_SEQUENCES	USER_TRIGGERS	V\$AW_OLAP
USER_SNAPSHOT_LOGS	USER_TS_QUOTAS	V\$AW_SESSION_INFO
USER_SNAPSHOT_REFRESH_TIMES	USER_TUNE_MVIEW	V\$BACKUP
USER_SNAPSHOTS	USER_TYPE_ATTRS	V\$BACKUP_ASYNC_IO
USER_SOURCE	USER_TYPE_METHODS	V\$BACKUP_CORRUPTION
USER_SOURCE_TABLES	USER_TYPES	V\$BACKUP_DATAFILE
USER_SQLJ_TYPE_ATTRS	USER_TYPE_VERSIONS	V\$BACKUP_DEVICE
USER_SQLJ_TYPE_METHODS	USER_UNUSED_COL_TABS	V\$BACKUP_FILES
USER_SQLJ_TYPES	USER_UPDATABLE_COLUMNS	V\$BACKUP_PIECE
USER_SQLSET	USER_USERS	V\$BACKUP_REDOLOG
USER_SQLSET_BINDS	USER_USTATS	V\$BACKUP_SET
USER_SQLSET_DEFINITIONS	USER_VARRAYS	V\$BACKUP_SPFIL
USER_SQLSET_REFERENCES	USER_VIEWS	V\$BACKUP_SYNC_IO
USER_SQLSET_STATEMENTS	USER_WARNING_SETTINGS	V\$BGPROCESS
USER_SQLTUNE_BINDS	USER_WORKSPACES	V\$BH
USER_SQLTUNE_PLANS	USER_XML_SCHEMAS	V\$BLOCK_CHANGE_TRACKING
USER_SQLTUNE_RATIONALE_PLAN	USER_XML_TAB_COLS	V\$BSP
USER_SQLTUNE_STATISTICS	USER_XML_TABLES	V\$BUFFERED_PUBLISHERS
USER_STORED_SETTINGS	USER_XML_VIEW_COLS	V\$BUFFERED_QUEUES
USER_SUBPART_COL_STATISTICS	USER_XML_VIEWS	V\$BUFFERED_SUBSCRIBERS
USER_SUBPART_HISTOGRAMS	V\$ACCESS	V\$BUFFER_POOL
USER_SUBPARTITION_TEMPLATES	V\$ACTIVE_INSTANCES	V\$BUFFER_POOL_STATISTICS
USER_SUBPART_KEY_COLUMNS	V\$ACTIVE_SERVICES	V\$CIRCUIT
USER_SUBSCRIBED_COLUMNS	V\$ACTIVE_SESSION_HISTORY	V\$CLASS_CACHE_TRANSFER
USER_SUBSCRIBED_TABLES	V\$ACTIVE_SESS_POOL_MTH	V\$CLASS_PING

V\$CLIENT_STATS	V\$FILEMETRIC	V\$LOCKED_OBJECT
V\$CONTEXT	V\$FILEMETRIC_HISTORY	V\$LOCK_ELEMENT
V\$CONTROLFILE	V\$FILE_PING	V\$LOCKS_WITH_COLLISIONS
V\$CONTROLFILE_RECORD_SECTION	V\$FILESPACE_USAGE	V\$LOCK_TYPE
V\$COPY_CORRUPTION	V\$FILESTAT	V\$LOG
V\$SCR_BLOCK_SERVER	V\$FILESTATXS	V\$LOGFILE
V\$CURRENT_BLOCK_SERVER	V\$FIXED_TABLE	V\$LOGHIST
V\$DATABASE	V\$FIXED_VIEW_DEFINITION	V\$LOG_HISTORY
V\$DATABASE_BLOCK_CORRUPTION	V\$FLASHBACK_DATABASE_LOG	V\$LOGMNR_CALLBACK
V\$DATABASE_INCARNATION	V\$FLASHBACK_DATABASE_LOGFILE	V\$LOGMNR_CONTENTS
V\$DATAFILE	V\$FLASHBACK_DATABASE_STAT	V\$LOGMNR_DICTIONARY
V\$DATAFILE_COPY	V\$GC_ELEMENT	V\$LOGMNR_LATCH
V\$DATAFILE_HEADER	V\$GC_ELEMENTS_WITH_COLLISIONS	V\$LOGMNR_LOGFILE
V\$DATAGUARD_CONFIG	V\$GCSHVMaster_INFO	V\$LOGMNR_LOGS
V\$DATAGUARD_STATUS	V\$GCSPFMaster_INFO	V\$LOGMNR_PARAMETERS
V\$DATAPUMP_JOB	V\$GES_BLOCKING_ENQUEUE	V\$LOGMNR_PROCESS
V\$DATAPUMP_SESSION	V\$GES_ENQUEUE	V\$LOGMNR_REGION
V\$DB_CACHE_ADVICE	V\$GLOBAL_BLOCKED_LOCKS	V\$LOGMNR_SESSION
V\$DBFILE	V\$GLOBALCONTEXT	V\$LOGMNR_STATS
V\$DBLINK	V\$GLOBAL_TRANSACTION	V\$LOGMNR_TRANSACTION
V\$DB_OBJECT_CACHE	V\$HS_AGENT	V\$LOGSTDBY
V\$DB_PIPES	V\$HS_PARAMETER	V\$LOGSTDBY_STATS
V\$DELETED_OBJECT	V\$HS_SESSION	V\$MANAGED_STANDBY
V\$DISPATCHER	V\$HVMaster_INFO	V\$MAP_COMP_LIST
V\$DISPATCHER_CONFIG	V\$INDEXED_FIXED_COLUMN	V\$MAP_ELEMENT
V\$DISPATCHER_RATE	V\$INSTANCE	V\$MAP_EXT_ELEMENT
V\$DLM_ALL_LOCKS	V\$INSTANCE_CACHE_TRANSFER	V\$MAP_FILE
V\$DLM_CONVERT_LOCAL	V\$INSTANCE_LOG_GROUP	V\$MAP_FILE_EXTENT
V\$DLM_CONVERT_REMOTE	V\$INSTANCE_RECOVERY	V\$MAP_FILE_IO_STACK
V\$DLM_LATCH	V\$JAVA_LIBRARY_CACHE_MEMORY	V\$MAP_LIBRARY
V\$DLM_LOCKS	V\$JAVAPool	V\$MAP_SUBELEMENT
V\$DLM_MISC	V\$JAVA_POOL_ADVICE	V\$MAX_ACTIVE_SESS_TARGET_MTH
V\$DLM_RESS	V\$LATCH	V\$METRIC
V\$DLM_TRAFFIC_CONTROLLER	V\$LATCH_CHILDREN	V\$METRICGROUP
V\$ENABLEDPRIVS	V\$LATCHHOLDER	V\$METRIC_HISTORY
V\$ENQUEUE_LOCK	V\$LATCH_MISSES	V\$METRICNAME
V\$ENQUEUE_STAT	V\$LATCHNAME	V\$MTTR_TARGET_ADVICE
V\$ENQUEUE_STATISTICS	V\$LATCH_PARENT	V\$MVREFRESH
V\$EVENT_HISTOGRAM	V\$LIBRARYCACHE	V\$MYSTAT
V\$EVENTMETRIC	V\$LIBRARY_CACHE_MEMORY	V\$NLS_PARAMETERS
V\$EVENT_NAME	V\$LICENSE	V\$NLS_VALID_VALUES
V\$EXECUTION	V\$LOADISTAT	V\$OBJECT_DEPENDENCY
V\$FAST_START_SERVERS	V\$LOADPSTAT	V\$OBSOLETE_PARAMETER
V\$FAST_START_TRANSACTIONS	V\$_LOCK	V\$OFFLINE_RANGE
V\$FILE_CACHE_TRANSFER	V\$LOCK	V\$OPEN_CURSOR
V\$FILE_HISTOGRAM	V\$LOCK_ACTIVITY	V\$OPTION

V\$OSSTAT	V\$RSRC_CONSUMER_GROUP_CPU_MTH	V\$SQLAREA
V\$PARALLEL_DEGREE_LIMIT_MTH	V\$RSRC_PLAN	V\$SQL_BIND_CAPTURE
V\$PARAMETER	V\$RSRC_PLAN_CPU_MTH	V\$SQL_BIND_DATA
V\$PARAMETER2	V\$RULE	V\$SQL_BIND_METADATA
V\$PGASTAT	V\$RULE_SET	V\$SQL_CURSOR
V\$PGA_TARGET_ADVICE	V\$RULE_SET_AGGREGATE_STATS	V\$SQL_OPTIMIZER_ENV
V\$PGA_TARGET_ADVICE_HISTOGRAM	V\$SCHEDULER_RUNNING_JOBS	V\$SQL_PLAN
V\$PQ_SESSTAT	V\$SEGMENT_STATISTICS	V\$SQL_PLAN_STATISTICS
V\$PQ_SLAVE	V\$SEGSTAT	V\$SQL_PLAN_STATISTICS_ALL
V\$PQ_SYSTAT	V\$SEGSTAT_NAME	V\$SQL_REDIRECTION
V\$PQ_TQSTAT	V\$SERVICE_EVENT	V\$SQL_SHARED_CURSOR
V\$PROCESS	V\$SERVICEMETRIC	V\$SQL_SHARED_MEMORY
V\$PROPAGATION_RECEIVER	V\$SERVICEMETRIC_HISTORY	V\$SQLTEXT
V\$PROPAGATION_SENDER	V\$SERVICES	V\$SQLTEXT_WITH_NEWLINES
V\$PROXY_ARCHIVEDLOG	V\$SERVICE_STATS	V\$SQL_WORKAREA
V\$PROXY_DATAFILE	V\$SERVICE_WAIT_CLASS	V\$SQL_WORKAREA_ACTIVE
V\$PWFILE_USERS	V\$SERV_MOD_ACT_STATS	V\$SQL_WORKAREA_HISTOGRAM
V\$PX_BUFFER_ADVICE	V\$SES_OPTIMIZER_ENV	V\$SQLXS
V\$PX_PROCESS	V\$SESS_IO	V\$STANDBY_LOG
V\$PX_PROCESS_SYSTAT	V\$SESSION	V\$STATISTICS_LEVEL
V\$PX_SESSION	V\$SESSION_CONNECT_INFO	V\$STATNAME
V\$PX_SESSTAT	V\$SESSION_CURSOR_CACHE	V\$STREAMS_APPLY_COORDINATOR
V\$QUEUE	V\$SESSION_EVENT	V\$STREAMS_APPLY_READER
V\$QUEUEING_MTH	V\$SESSION_LONGOPS	V\$STREAMS_APPLY_SERVER
V\$RECOVER_FILE	V\$SESSION_OBJECT_CACHE	V\$STREAMS_CAPTURE
V\$RECOVERY_FILE_DEST	V\$SESSION_WAIT	V\$SUBCACHE
V\$RECOVERY_FILE_STATUS	V\$SESSION_WAIT_CLASS	V\$SYSAUX_OCCUPANTS
V\$RECOVERY_LOG	V\$SESSION_WAIT_HISTORY	V\$SYSMETRIC
V\$RECOVERY_PROGRESS	V\$SESSMETRIC	V\$SYSMETRIC_HISTORY
V\$RECOVERY_STATUS	V\$SESSTAT	V\$SYSMETRIC_SUMMARY
V\$REPLPROP	V\$SESS_TIME_MODEL	V\$SYS_OPTIMIZER_ENV
V\$REPLQUEUE	V\$SGA	V\$SYSSTAT
V\$REQDIST	V\$SGA_CURRENT_RESIZE_OPS	V\$SYSTEM_CURSOR_CACHE
V\$RESERVED_WORDS	V\$SGA_DYNAMIC_COMPONENTS	V\$SYSTEM_EVENT
V\$RESOURCE	V\$SGA_DYNAMIC_FREE_MEMORY	V\$SYSTEM_PARAMETER
V\$RESOURCE_LIMIT	V\$SGAINFO	V\$SYSTEM_PARAMETER2
V\$RESUMABLE	V\$SGA_RESIZE_OPS	V\$SYSTEM_WAIT_CLASS
V\$RMAN_CONFIGURATION	V\$SGASTAT	V\$SYS_TIME_MODEL
V\$RMAN_OUTPUT	V\$SHARED_POOL_ADVICE	V\$TABLESPACE
V\$RMAN_STATUS	V\$SHARED_POOL_RESERVED	V\$TEMP_CACHE_TRANSFER
V\$ROLLNAME	V\$SHARED_SERVER	V\$TEMP_EXTENT_MAP
V\$ROLLSTAT	V\$SHARED_SERVER_MONITOR	V\$TEMP_EXTENT_POOL
V\$ROWCACHE	V\$SORT_SEGMENT	V\$TEMPFILE
V\$ROWCACHE_PARENT	V\$SORT_USAGE	V\$TEMP_HISTOGRAM
V\$ROWCACHE_SUBORDINATE	V\$SPPARAMETER	V\$TEMPORARY_LOBS
V\$RSRC_CONSUMER_GROUP	V\$SQL	V\$TEMP_PING

V\$TEMPSEG_USAGE	V\$TIMEZONE_NAMES	V\$WAITCLASSMETRIC
V\$TEMP_SPACE_HEADER	V\$TRANSACTION	V\$WAITCLASSMETRIC_HISTORY
V\$TEMPSTAT	V\$TRANSACTION_ENQUEUE	V\$WAITSTAT
V\$TEMPSTATXS	V\$TRANSPORTABLE_PLATFORM	X\$KCBFWAIT
V\$THREAD	V\$TYPE_SIZE	X\$KSLLT
V\$THRESHOLD_TYPES	V\$UNDOSTAT	X\$KSPPi
V\$TIMER	V\$VERSION	X\$KSPPSV
V\$TIMEZONE_FILE	V\$VPD_POLICY	X\$KSQST

Résumé

Le dictionnaire de données Oracle 10g est souvent obscur, y compris pour de nombreux administrateurs Oracle. Ce chapitre traite à la fois de sa création « minimale », des différents scripts SQL additionnels et des vues qui permettent d'y accéder. L'accès aux vues DBA, ALL, USER, V\$ est commenté. Leur connaissance générale est indispensable à tous les administrateurs Oracle, c'est pourquoi nous vous en fournissons la liste complète.

Partie III

Installer et paramétrer Oracle 10g sous Windows

Chapitre 13 : L'installation d'Oracle 10g sous Windows

Chapitre 14 : Création d'une base Oracle 10g

Chapitre 15 : Oracle Net, le middleware Oracle

Chapitre 16 : Accéder à Oracle depuis Windows

13

L'installation d'Oracle 10g sous Windows

Ce chapitre a pour but de vous guider dans l'installation d'Oracle 10g sur un serveur et des postes client Windows. Bonne nouvelle, l'installation est conforme aux habitudes des utilisateurs Windows : peu de prérequis sont nécessaires, l'installation est sans surprise, les logiciels et une base de données initiale très rapidement installés. Ces étapes sont détaillées pas à pas, notamment le paramétrage d'Oracle.

Nous abordons la configuration de logiciels livrés avec Oracle 10g dans des chapitres spécifiques. C'est le cas, par exemple, pour Oracle Net Services et ses interfaces Java JDBC. Chaque fois qu'un point est détaillé dans cet ouvrage, nous indiquons le chapitre correspondant.

Pendant l'installation, nous créerons une base Oracle par défaut. Le chapitre suivant, *Création d'une base Oracle 10g*, aborde dans le détail les étapes de création d'une nouvelle base Oracle 10g.

Enfin, je vous conseille de consulter les sites Web :

- <http://perso.club-Internet.fr/gbriard> : site Web de l'auteur contenant les mises à jour du livre, de nombreuses informations et des liens indispensables ;
- <http://metalink.oracle.com> : si vous bénéficiez du support Oracle, même pour une autre plate-forme ;
- <http://otn.oracle.com> : OTN (*Oracle Technology Network*) est le site technique d'Oracle. Il offre de très nombreuses ressources, dont des forums de discussion, et permet de télécharger les derniers logiciels Oracle. Les notes techniques les plus récentes qui accompagnent les produits se trouvent sur ce site ;

- <http://otn.oracle.com/docs> : la documentation Oracle en ligne ;
- <http://www.microsoft.fr> : indispensable pour obtenir des informations techniques ainsi que les Services Packs.

Les versions d'Oracle Database 10g

Comme pour les versions précédentes, Oracle 10g est disponible en version d'évaluation (non bridée) téléchargeable sur Internet. Compte tenu de la diffusion des réseaux à haut débit, c'est la façon la plus simple et la plus rapide de récupérer Oracle 10g.

Oracle propose plusieurs versions commerciales de son produit phare, la base de données. Autour d'un socle unique, différents packages commerciaux existent et les prix varient fortement.

Avant tout achat, regardez attentivement les caractéristiques techniques des différentes offres commerciales sur <http://otn.oracle.com/technology> et leurs tarifs sur le site de vente en ligne <http://www.oracle.com>. Vous constaterez que très, très souvent, ce n'est pas la version Oracle la plus intéressante financièrement qui vous est proposée... Oracle fait des efforts tarifaires : profitez-en !

Oracle propose actuellement quatre versions commerciales d'Oracle 10g. Les différences concernent les droits d'usages : ces versions sont présentes sur le même CD-Rom et ne constituent qu'un choix à effectuer durant l'installation :

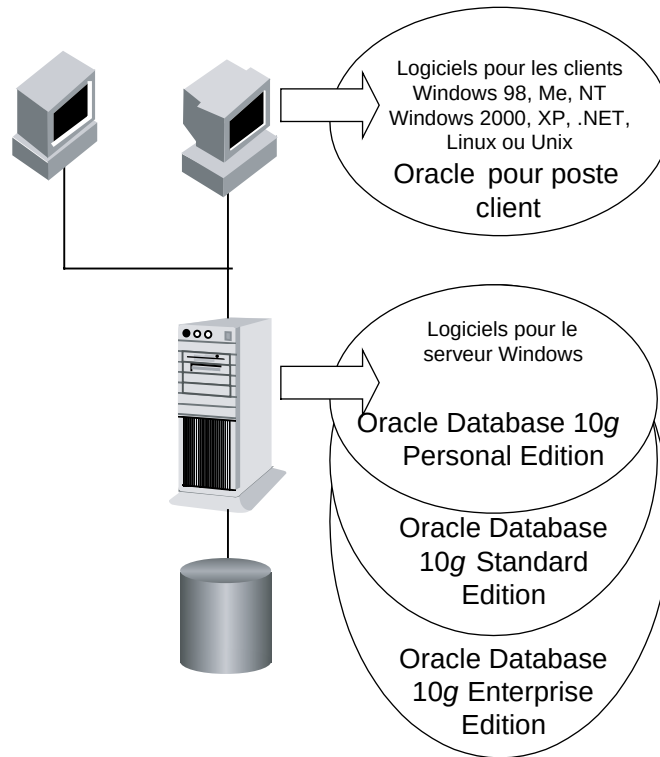
- « Oracle Database 10g Personal Edition » : pour un seul utilisateur sur un poste de travail ou un serveur de développement. Utile pour disposer officiellement d'une version avec le support à un prix très modique !
- « Oracle Database 10g Standard Edition One » : son utilisation n'est autorisée que sur des serveurs acceptant un maximum de deux processeurs.
- « Oracle Database 10g Standard Edition (SE) » : son utilisation est autorisée pour des serveurs limités à deux processeurs ou mis en cluster (avec un maximum de quatre processeurs pour le cluster).
- « Oracle Database 10g Enterprise Edition (EE) » : les limites de la version précédente sont levées mais le prix de cette version est bien supérieur à la version SE. La version EE est imposée comme base pour l'achat d'options supplémentaires
- « Oracle Database 10g Express Edition » : une version Oracle gratuite, assortie de quelques limites mineures, présentées à l'annexe 8.

Compte tenu de la pression que constituent des bases de données comme MySQL, SQL Server, etc., d'autres packages à prix agressifs arriveront. La concurrence a du bon !

Les différences entre ces versions concernent essentiellement la puissance du serveur hôte (nombre de processeurs « possibles » sur le serveur, virtualisation, cluster, etc.). Le coût des logiciels est proportionnel. Pour le reste, il s'agit de produits identiques.

Figure 13-1

*Logiciels nécessaires à
l'installation d'Oracle 10g
pour Windows*



Les différents CD-Rom disponibles

La base de données et les produits annexes sont maintenant livrés sur des CD-Rom différents. Voici la liste actuelle :

- Oracle Database 10g : tous les logiciels nécessaires à l'installation d'une base Oracle regroupés sur un CD-Rom ;
- Cluster Ready Service : regroupe les composants nécessaires à l'installation d'une configuration cluster. Il doit alors être présent avant l'installation d'Oracle Real Application Clusters ;
- Oracle Database Client : à partir de cette version d'Oracle, l'installation d'un client Oracle « seul » n'est plus possible depuis le CD-Rom contenant la base de données mais s'effectue à partir de ce CD-Rom ;
- Oracle Database Companion CD Components : regroupe les modules :
 - Jpublisher
 - Legatoo Single Server Version : pour les sauvegardes avec l'outil Legato ;
 - Natively Compiled Java Libraries ;

- Oracle Database Exemples : les jeux d'exemples ;
- Oracle Application Express : l'outil de développement d'applications Web présenté dans cet ouvrage ;
- Oracle HTTP Server : le serveur Apache adapté à Oracle ;
- Oracle Text Supplied Knowledge Base ;
- Oracle Migration Workbench : des utilitaires de migration d'autres bases de données vers Oracle.

Toutes les images de ces CD-Rom sont téléchargeables depuis <http://otn.oracle.com>.

Utilisation de l'aide Oracle

Pour un logiciel comme Oracle, accéder à la documentation est indispensable. Il y a d'abord la documentation officielle, au format papier ou électronique, qui intègre souvent des volumes non encore imprimés. Vous pouvez y accéder en ligne à l'adresse <http://otn.oracle.com/docs>.

Diverses autres documentations sont également disponibles. Et là, il faut être curieux ! Nous vous proposons quelques pistes à suivre lorsque la documentation ne suffit pas à répondre à vos questions.

La documentation sur CD-Rom

La documentation officielle des logiciels Oracle existe imprimée mais aussi au format HTML ou PDF. Pour les versions électroniques, Oracle propose des moyens simples et rapides pour trouver l'information recherchée.

Le support technique

Si vous disposez d'un contrat de support à Oracle Metalink, le site <http://metalink.oracle.com> est incontournable. Les forums du site <http://otn.oracle.com> sont aussi d'une aide précieuse... et gratuite !

Les autres documentations

D'autres documentations existent. Il faut être curieux, faire des recherches : le résultat est souvent très intéressant.

Les documentations « en ligne »

Les produits Oracle sont accompagnés de documentations en ligne. N'hésitez pas à les utiliser. Elles comportent souvent des informations qui précisent la version papier.

Les fichiers accompagnant les produits

La majorité des produits Oracle sont accompagnés d'un fichier Read-me présent dans l'arborescence de la distribution. Les fichiers Read-me contiennent les dernières informations et restrictions, les bugs rencontrés, etc.

Les textes supplémentaires

De nombreux textes supplémentaires non répertoriés et non indiqués sont livrés avec les produits. Il s'agit souvent de documents complémentaires non encore intégrés à la documentation officielle. Ils concernent les derniers « bugs », des recommandations, des conseils, etc. Recherchez dans l'arborescence des produits Oracle des fichiers de type `read*.*`, `*.doc`, `*.wri`, etc., vous y trouverez de véritables mines d'or.

Les étapes de l'installation d'Oracle 10g

L'installation d'Oracle 10g pour Windows est simple et rapide. Elle nécessite de suivre avec précision quelques étapes afin d'éviter des problèmes rencontrés suite à des choix effectués dans une trop grande précipitation.

L'installation sous Windows s'avère être plus simple et plus rapide qu'une installation sous Unix/Linux où les étapes de vérification et de préparation de l'environnement sont plus exigeantes.

1. **Vérification des prérequis** : assurez-vous que votre système Windows possède toutes les caractéristiques matérielles et logicielles ainsi qu'un espace disque suffisant pour les produits que vous souhaitez installer. Ces recommandations sont décrites dans ce chapitre.
2. **Adapter l'environnement Windows** : Oracle recommande de vérifier certaines caractéristiques du système Windows. Nous indiquons l'intérêt que vous avez à effectuer (ou non) ces modifications.
3. **Installation des logiciels Oracle** : le nouveau logiciel Oracle Universal Installer (écrit en Java) permet d'effectuer graphiquement l'installation de vos logiciels Oracle sur Windows. Nous détaillons cette étape.
4. **Post-installation** : concerne toutes les étapes postérieures à l'installation pour rendre la configuration opérationnelle, comme la création d'objets de la base, la configuration de l'environnement client-serveur, la mise en œuvre d'un environnement utilisateur, etc.
5. **Installation des clients** : si vous souhaitez que des stations distantes accèdent à la base Oracle en mode client-serveur, le processus d'installation est à reproduire pour ces stations, le CD-Rom à utiliser est toutefois différent.

Étape 1 : vérification des prérequis

L'installation d'Oracle 10g pour Windows ne nécessite que quelques vérifications préalables. Toutes les recommandations suivantes concernent Oracle 10g R1 pour Windows. Elles peuvent évoluer, mais l'expérience prouve que les conditions à satisfaire sont stables dans le temps.

Niveau matériel requis

Matériel	Prérequis
Processeur « minimum »	Un processeur minimum de type Intel Pentium II ou équivalent AMD est demandé. Windows est « gourmand » et Oracle l'est aussi : prévoyez large !
Mémoire	Un minimum de 256 Mo est demandé. Si vous disposez de moins, vous serez pénalisé durant l'installation, car l'outil Java est gourmand. Il faudra alors compenser le manque par de la mémoire virtuelle sur disque. Vous avez aussi besoin de beaucoup d'espace disque si vous souhaitez tester les options Java du noyau Oracle 10g. Une base de données prenant en charge beaucoup d'utilisateurs simultanés peut nécessiter plus de 1 Go de mémoire à elle seule.
Mémoire virtuelle	Une mémoire virtuelle égale au double de la capacité en mémoire vive est conseillée.
CD-Rom	Pour ceux qui n'ont pas téléchargé Oracle 10g, un lecteur de CD-Rom est nécessaire.

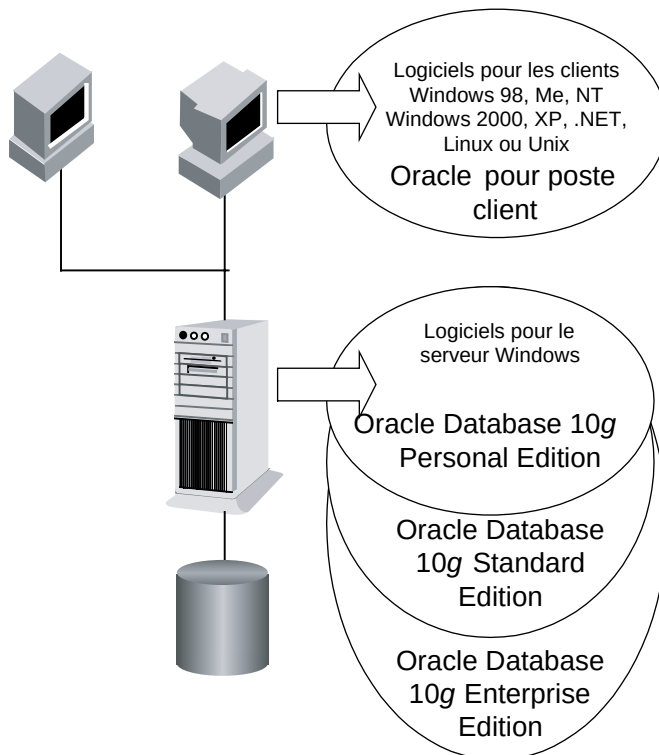
Comment vérifier ces paramètres sur un serveur Windows ?

Pour déterminer la mémoire RAM installée sur le système, utilisez l'onglet *Performances* du Gestionnaire de tâches.

Parmi les informations affichées sur cet écran, figurent la mémoire physique du serveur et la mémoire virtuelle allouée au serveur.

Figure 13-2

Gestionnaire de tâches
Windows



Espace disque requis

Pour installer Oracle 10g « confortablement », il faut disposer au minimum de :

- 2 Go d'espace disque pour installer tous les logiciels Oracle ;
- 700 Mo par base Oracle version Enterprise créée, plus le volume de données à y stocker.

Nous vous conseillons très fortement d'organiser cet espace disque en consultant les règles OFA (*Oracle Optimal Flexible Architecture*) décrites au chapitre 9, *Les fichiers d'une base Oracle 10g*.

Les installations décrites dans ce livre suivent les règles OFA.

Nous vous conseillons aussi de lire le chapitre 27, *Configurer les disques pour Oracle*, pour choisir la configuration disque **avant** d'installer Oracle. Il est facile d'effectuer ces choix maintenant mais, une fois les logiciels installés et les bases créées, les modifications disque sont bien plus complexes.

Versions Windows supportées par Oracle Database 10g

Oracle 10g fonctionne sur les principaux systèmes d'exploitation Microsoft. Ces caractéristiques étant susceptibles de changer, nous vous conseillons de consulter le support Oracle. Comme vous le constaterez, Windows NT et l'édition Standard de XP ne sont pas supportées par Oracle 10g.

Système d'exploitation	9i R2	10g Release 1	10g Release 2
Windows NT (Service Pack 6)	OUI	OUI	NON
Windows 2000	OUI	OUI	OUI
Windows XP Standard	NON	NON	NON
Windows XP Professional	OUI	OUI	OUI
Windows Server 2003, 32 bits	OUI	OUI	OUI
Windows XP et Windows Server 2003 pour processeur 64 bits Itanium	OUI	OUI	OUI
Windows XP et Windows Server 2003 pour processeur 64 bits (Intel EM64T et AMD Opteron)	OUI	OUI	OUI
Service Pack	Vérifiez sur le site http://metalink.oracle.com . En général, le dernier niveau de service pack est conseillé.		

Environnement Java nécessaire

Un environnement Java est fourni avec Oracle 10g comme partie intégrante de son installation. Il permet de faire fonctionner le logiciel d'installation Oracle Universal Installer, application écrite en Java, ainsi que les modules Web accédant à Oracle (iSQL*Plus, Oracle Enterprise Manager, etc.).

Environnement réseau indispensable

Une carte réseau doit être installée et le réseau doit être configuré avant de commencer l'installation d'Oracle 10g.

Les systèmes d'exploitation de Microsoft ne démarrent pas les couches « réseau » si la carte réseau n'est pas reliée par un câble à un élément actif (hub). Comme Oracle a besoin de la présence active des couches réseau lors de certaines étapes de l'installation, il faut impérativement relier le serveur à un hub lors de l'installation, sous peine de rencontrer des blocages. Cette liaison n'est plus nécessaire lors du fonctionnement courant.

Vérification de la configuration réseau du serveur

Ces paramétrages et vérifications sont indispensables pour accéder aux interfaces Web proposées par Oracle : Oracle Enterprise Manager, iSQL*Plus, etc.

Tout d'abord, vérifiez que le nom du serveur hébergeant la base Oracle est connu du serveur DNS :

```
# Depuis Démarrer > Exécuter, lancez la commande cmd
# depuis la fenêtre de commande, lancez
# ping <nom du serveur>
ping ordigb
# en cas d'insuccès, demandez à votre administrateur
# d'ajouter le nom du serveur dans le DNS.
# en attendant, accédez au serveur en utilisant son adresse IP
# ping <adresse IP du serveur>
ping 129.128.34.2
```

Si vous disposez d'un serveur de test sur une station isolée, donc sans DNS présent, modifiez le fichier <X>:\WINDOWS\system32\drivers\etc\hosts à l'aide du Bloc-notes (et non Wordpad qui insère des caractères parasites dans la mise en page).

```
# modifiez la ligne
# 127.0.0.1      localhost
# par
# 127.0.0.1      localhost <nom de l'ordinateur>
                localhost ORDIGB
# sauvegardez et sortez
```

Étape 2 : adapter et préparer Windows

A l'inverse des systèmes Unix et Linux qui demandent de nombreuses étapes de préparation, Windows n'en nécessite presque pas. Il vous suffit d'avoir suffisamment de mémoire et d'espace disque pour que l'installation se déroule sans problème.

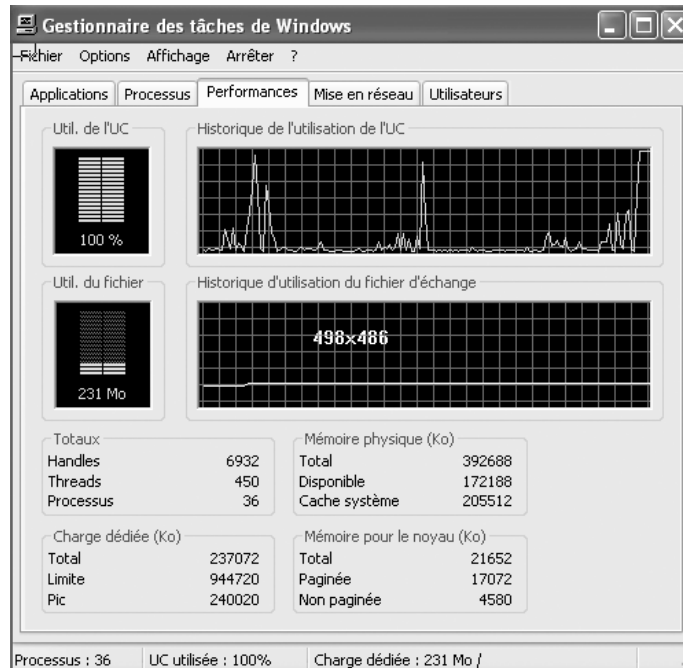
Ces préparatifs concernent aussi le respect des règles OFA (*Optimal Flexible Architecture*) avant de commencer l'installation des produits.

Le respect des règles OFA (Optimal Flexible Architecture)

L'architecture OFA permet une organisation souple et rigoureuse de l'installation des logiciels Oracle et des bases de données. Les règles OFA sont détaillées au chapitre 9, *Les fichiers d'une base Oracle 10g*. Nous vous conseillons d'en prendre connaissance car l'installation que nous allons effectuer respecte les règles OFA. Pour mémoire, l'installation d'une distribution Oracle 10g et d'une base nommée TEST sur votre système utilise les répertoires suivants :

Figure 13-2

*Arborescence OFA
d'une installation
Oracle 10g sous Windows*



L'exemple précédent d'une architecture OFA concerne une base de données TEST, avec une seule version Oracle installée sur la machine.

Étape 3 : installation d'Oracle 10g pour Windows

Avant de débiter une installation Oracle 10g, vérifiez sur le site <http://otn.oracle.com> et sur le site de l'auteur <http://perso.club-internet.fr/gbriard> que votre version est la dernière sortie et que des correctifs supplémentaires ne sont pas requis.

Comment télécharger Oracle 10g ?

Vous pouvez télécharger Oracle 10g à partir du site Web Oracle : <http://otn.oracle.com/technology>. Les versions pour Windows sont proposées mais aussi celles pour d'autres systèmes d'exploitation Unix et Linux dont IBM AIX, Hewlett Packard HP-UX et SunOS.

Pour ces systèmes, je vous conseille de vous référer à mes autres ouvrages abordant la configuration d'Oracle sous Linux. Ils sont eux aussi publiés aux éditions Eyrolles.

Toutes les versions téléchargeables n'embarquent actuellement aucun dispositif limitant leur durée d'utilisation : c'est la confiance totale. La version Express Edition, gratuite, vient renforcer cette volonté de rendre Oracle accessible à tous.

Installer les logiciels Oracle

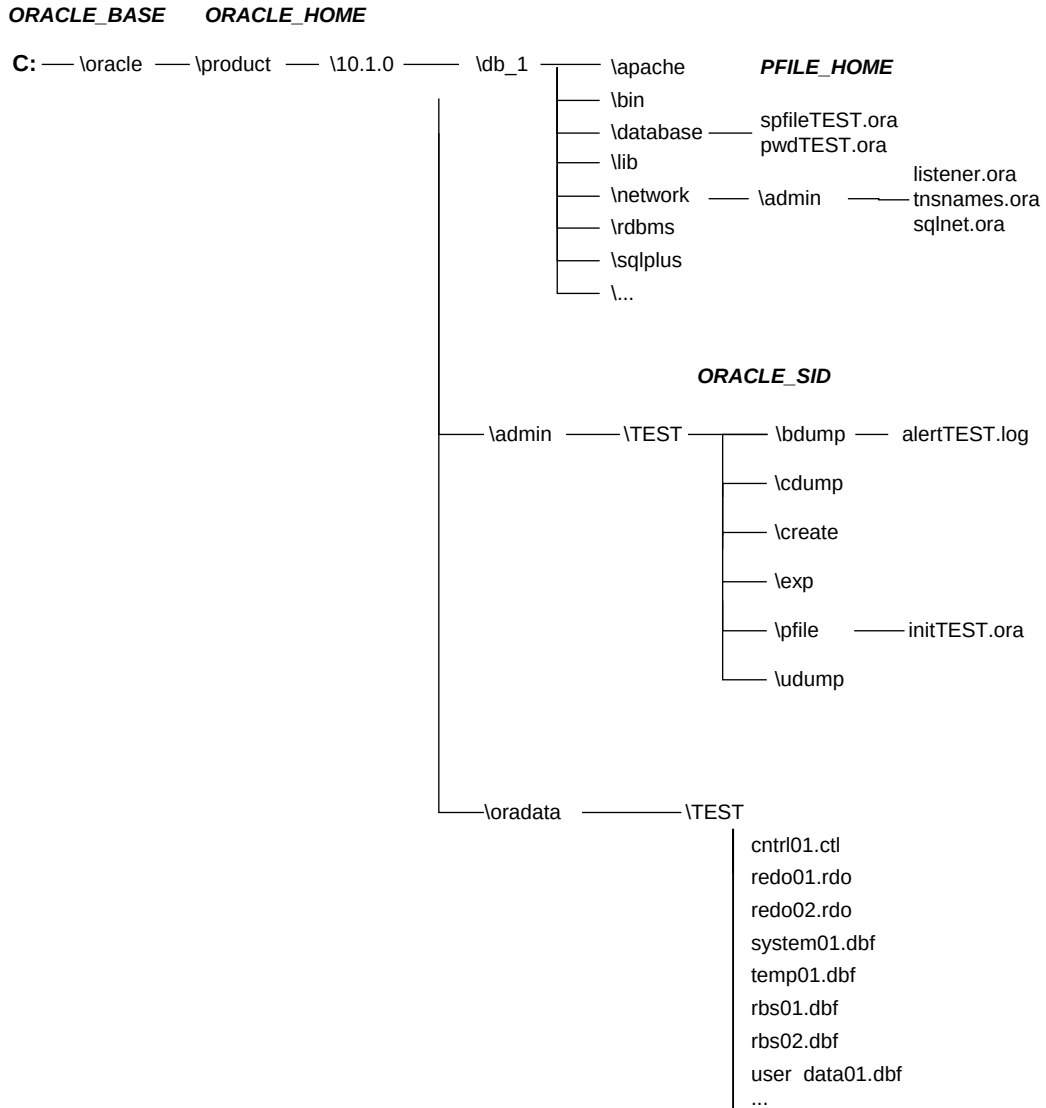
Pour les habitués d'Oracle, il n'est plus possible d'installer les logiciels Oracle à partir de l'ancien outil « mode caractère ». Depuis Oracle8i, un logiciel d'aspect identique pour l'ensemble des plates-formes (Windows, Linux, Unix,...) est utilisé. Il s'agit de l'OUI (*Oracle Universal Installer*), écrit en langage Java.

Si vous installez Oracle 10g sous Windows XP Professional **Service Pack 2**, consultez le site de l'auteur : <http://gbriard.club-internet.fr>. Des recommandations concernant cette version y figurent.

Lancer Oracle Universal Installer

Pour lancer Oracle Universal Installer :

1. Connectez-vous sous un utilisateur Windows ayant le privilège **Administrateur**. Dans cet exemple, nous sommes connectés sous le compte *Administrateur*.
2. Dès que le CD-Rom est inséré dans le lecteur, l'écran suivant apparaît.

**Figure 13-3***Lancement automatique de l'installation*

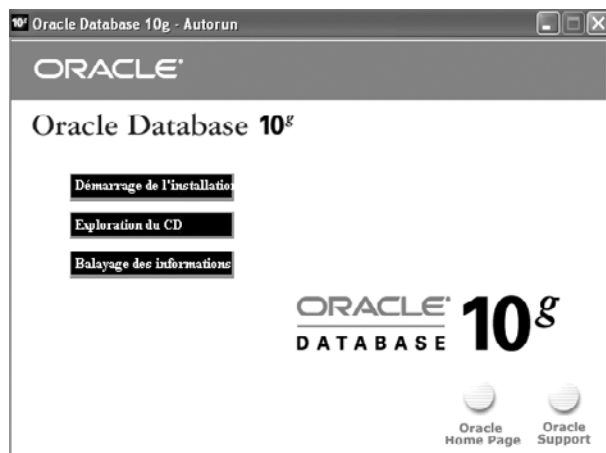
Si vous avez téléchargé les logiciels depuis Internet et s'ils sont présents sur le disque, lancez le programme `autorun\autorun.exe` depuis la racine.

3. Sélectionnez **Démarrage de l'installation**.

Installation pas à pas des logiciels Oracle Database 10g

Figure 13-4

Lancement de l'Installer



Deux options d'installation sont proposées : l'installation de base et l'installation avancée. L'installation de base est quasi automatique. Elle présente l'avantage de ne pas se poser de question... mais c'est aussi son principal inconvénient ! Pour vous permettre de découvrir les options d'installation d'Oracle, nous allons effectuer pas à pas une installation avancée.

Une aide en ligne efficace vous guide durant le processus d'installation. Pour y accéder, cliquez sur *Aide* :

Figure 13-5

Aide en ligne de l'Installer

Installation d'Oracle Database 10g

Bienvenue dans l'installation d'Oracle Database 10g

Sélectionnez la méthode d'installation que vous souhaitez utiliser.

☐ Installation de base

Emplacement du répertoire d'origine Oracle Home :

Type d'installation :

☒ Créer une base de données de départ (720MB) supplémentaire

Nom global de base de données :

Mot de passe de base de données : Confirmer :

Mot de passe pour les comptes SYS, SYSTEM, SYSMAN, DBSNMP.

☒ Installation avancée

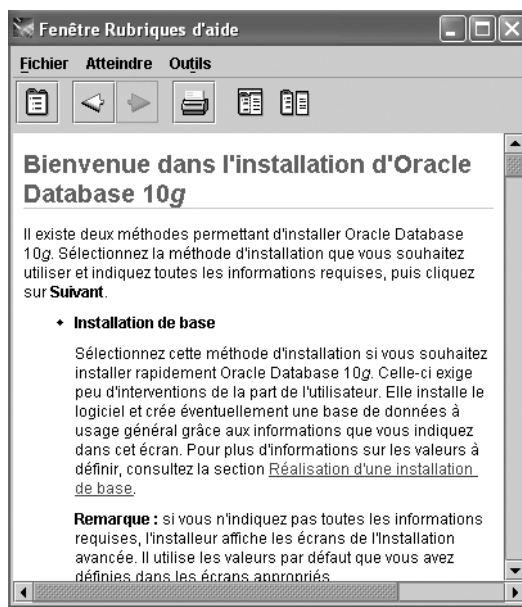
Permet d'effectuer des sélections avancées, telles que des mots de passe différents pour les comptes SYS, SYSTEM, SYSMAN et DBSNMP, un jeu de caractères de base de données, des langues de produit, les sauvegardes automatiques, l'installation personnalisée et d'autres options de stockage, telles qu'Automatic Storage Management (ASM).

ORACLE

Pour commencer l'installation, retournez à l'écran d'accueil et cliquez sur *Suivant* :

Figure 13-6

Emplacement de
l'Oracle_Home



Il faut tout d'abord indiquer l'emplacement du fichier listant les logiciels contenus sur le CD-Rom. Il s'agit du fichier <X>:\stage\products.xml. <X> représente un volume disque, un lecteur de CD-Rom ou un DVD-Rom.

Ensuite, il faut indiquer l'emplacement (ou Oracle_Home) où seront installés les logiciels Oracle (les bases de données seront placées à un autre endroit). L'Oracle_Home regroupe comme paramètres :

- le répertoire où seront installés les logiciels Oracle ;
- la modification de la variable PATH du système pour y ajouter le chemin vers les exécutables ;
- des entrées dans la base de registre ;
- des groupes de programmes placés dans les menus.

L'identifiant de l'Oracle_Home sera OraDb10g_homeN, N étant une valeur s'incrémentant correspondant à un répertoire racine. Le répertoire racine où seront installés tous les logiciels sera <X>:\oracle\product\10.1.0\db_<N>. Dans l'exemple, Oracle_Home se nommera OraDb10g_home1 et pointera vers le répertoire C:\oracle\product\10.1.0\db_1.

Choisissez un répertoire Oracle_Home où vous disposez d'au moins 2 Go d'espace disque. Prévoyez large : changer le répertoire Oracle_Home est une opération lourde.

Tous les paramètres liés à cet Oracle Home sont conservés dans la base de registre en HKEY_LOCAL_MACHINE\SOFTWARE\ORACLE\KEY_OraDb10g_home1. Le contenu des valeurs de la base de registre est abordé au chapitre 5, *Fonctionnement d'Oracle 10g sous Windows*.

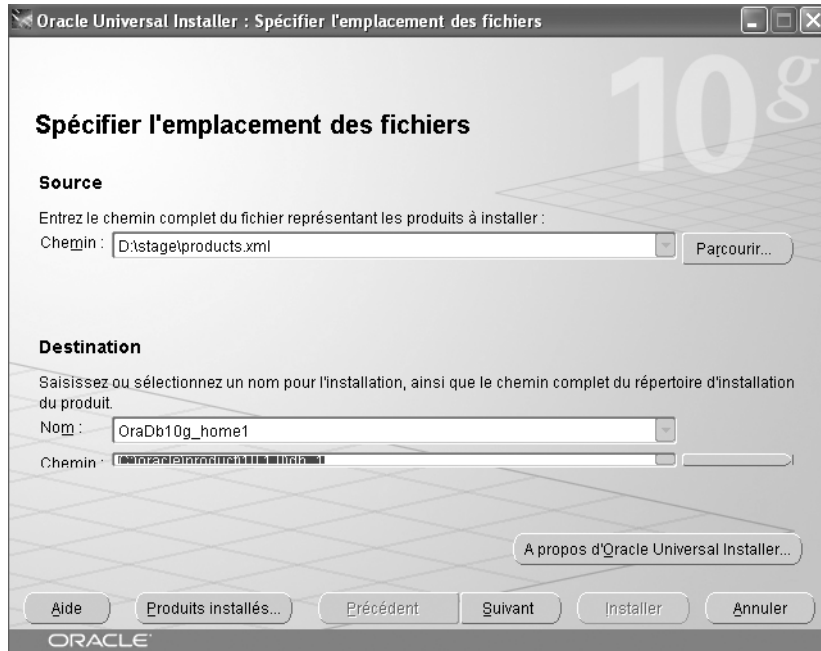


Figure 13-7
Produits disponibles

OUI propose l'installation des trois familles de produits : les versions d'Oracle Database 10g (Enterprise, Standard, Personal) et une installation personnalisée permettant de sélectionner et d'installer un seul composant. C'est l'option personnalisée qu'il vous faudra utiliser si vous souhaitez installer un composant logiciel supplémentaire.

Afin de montrer le plus d'options possibles, l'installation de la version Enterprise est présentée.

C'est aussi cet écran qui permet de choisir les langues dans lesquelles s'afficheront les messages. Pour installer les messages en français, cliquez sur l'option *Langues du produit*.

Pour l'installation, les langues anglaise (toujours présente) et française sont présentées par défaut. Oracle s'appuie sur le langage activé au niveau du système d'exploitation pour proposer le français. Cliquez sur OK pour valider et revenir à l'écran précédent.

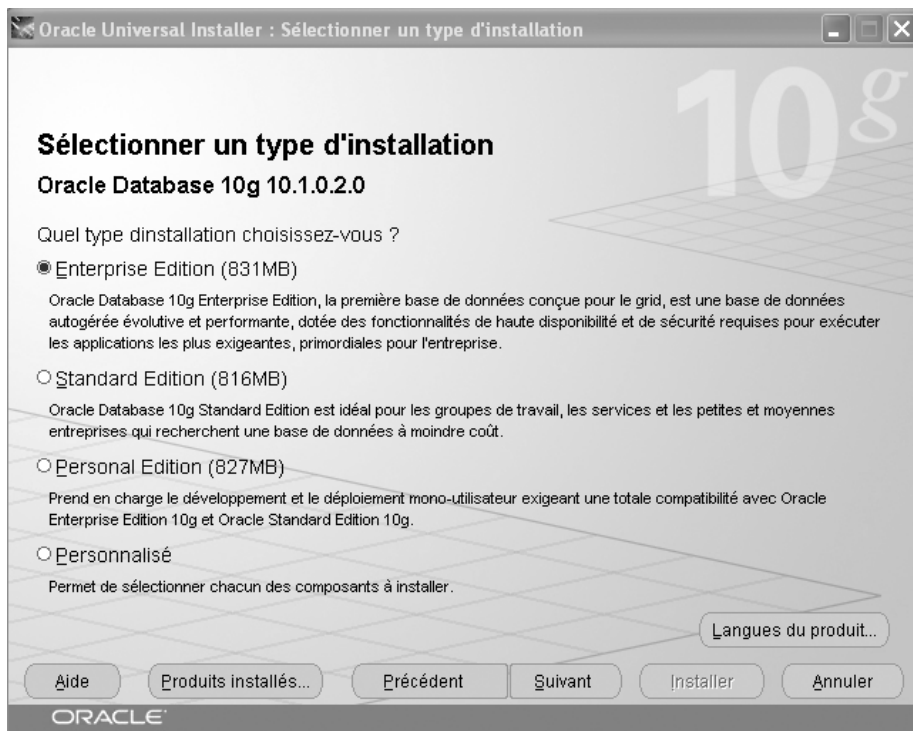


Figure 13-8

Langues choisies pour installer les messages

Figure 13-9

Choix du type de base de données



Le choix du type de base de données à créer constitue une étape importante de l'installation :

- les trois premières options (*Universel*, *Traitement des transactions*, *Data Warehouse*) proposent d'installer les logiciels plus une base préconfigurée. Ce type d'installation est rapide et consomme peu de ressources système. Au lieu de suivre le processus classique de création d'une base puis de construction des catalogues, les bases de données sont copiées depuis le CD-Rom puis adaptées ;
- l'option *Avancé* installe les logiciels Oracle puis lance l'assistant graphique DBCA (*DataBase Configuration Assistant*) qui permet de choisir toutes les options de création de la nouvelle base. L'assistant DBCA est détaillé au chapitre suivant, il ne sera pas utilisé à cette étape ;
- enfin, il est possible d'installer uniquement les logiciels, sans créer de base de données.

Si vous êtes un administrateur Oracle débutant, je vous conseille de choisir la première option. Les logiciels, ainsi qu'une base de données initiale, seront créés.

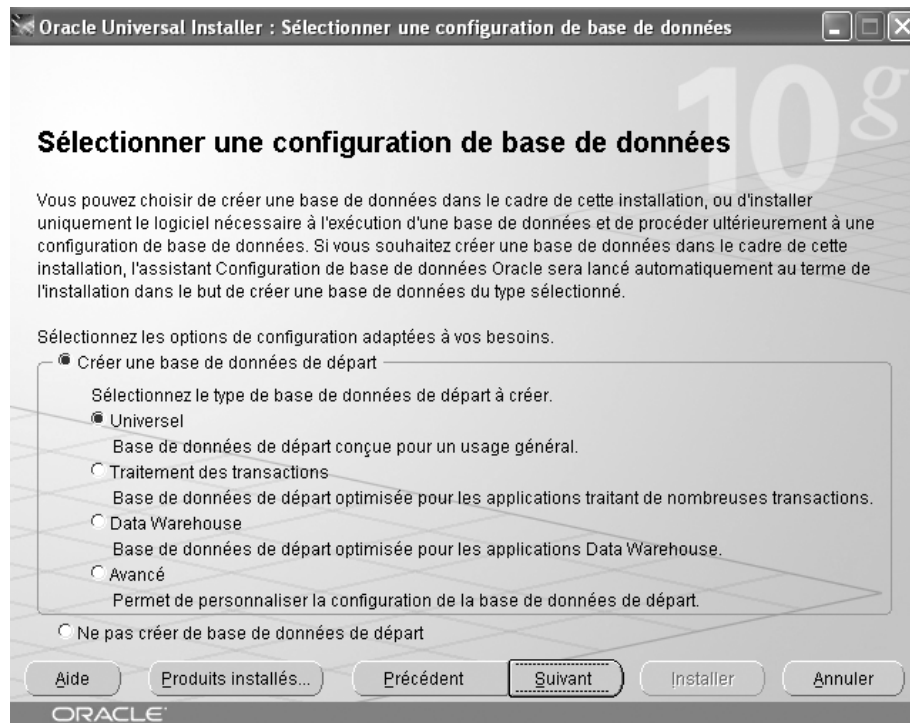


Figure 13-10

Identification de la nouvelle base de données

Les variables concernées par cet écran d'installation sont :

Variable Oracle	Commentaire	Exemple de valeur
Global Database Name	Nom global de la base de données. C'est la concaténation du nom de la base de données et du domaine de la base de données.	TEST.gilles.fr
SID	System Identifier : identifie de manière unique l'instance.	TEST
DB_NAME	Identifie de manière unique la base de données. Dans le cas d'un cluster, une base de données est composée de plusieurs instances. Dans notre installation, le DB_NAME et le SID seront identiques.	TEST
DB_DOMAIN	Identifie le domaine de la base de données.	gilles.fr

Pour supporter le français ET le caractère Euro (€), les seuls jeux de caractères compatibles sont : WE8ISO8859P15, WE8MSWIN1252, Unicode (AL32UTF8 ou UTF8). Pour plus d'informations, reportez-vous à l'annexe 5 *NLS, le support des langues par Oracle et l'euro*. Sous Windows, nous vous conseillons de choisir WE8MSWIN1252.

Enfin, la base de données sera créée avec des exemples.

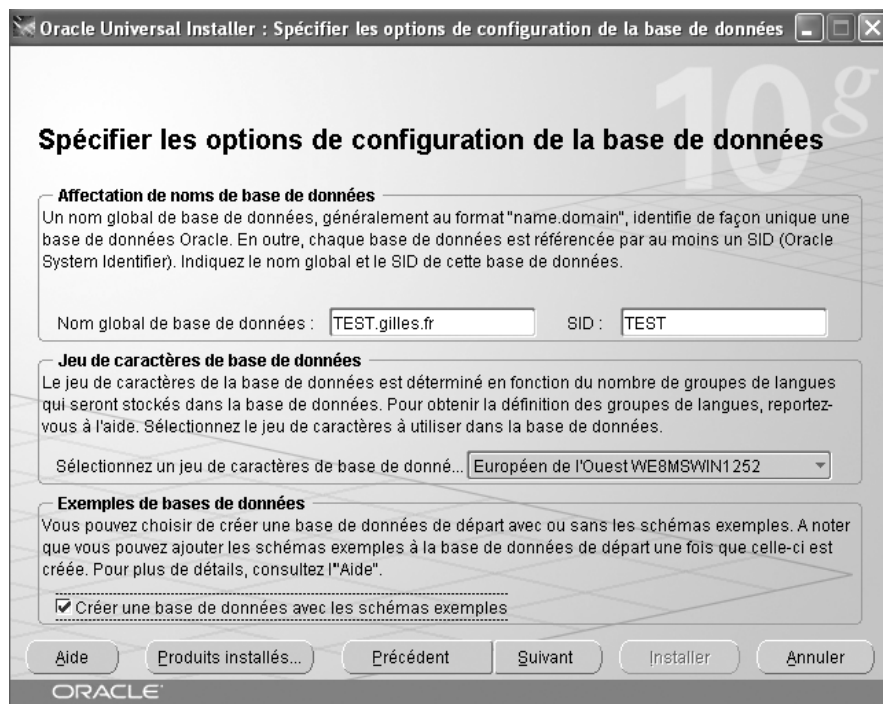


Figure 13-11

Gestion de la base de données

Pour simplifier l'administration d'une base, Oracle propose l'interface Web Oracle Enterprise Manager. Deux options sont possibles :

- utiliser la version locale d'OEM (*Oracle Enterprise Manager*), qui permet l'administration distante par un navigateur Web. Dans ce cas, la configuration OEM sera répétée pour chaque serveur où sera installé Oracle ;
- installer la version centralisée d'OEM : Oracle Enterprise Manager Grid Control qui communiquera avec toutes les bases distantes ou locales à administrer. La communication s'effectuera via des agents (logiciels relais installés sur chaque serveur Oracle).

Dans l'exemple, aucun agent n'a été détecté. La version *locale* d'OEM sera installée et les alertes seront signalées par e-mail.

OEM est présenté au chapitre 25, *Oracle Enterprise Manager*.

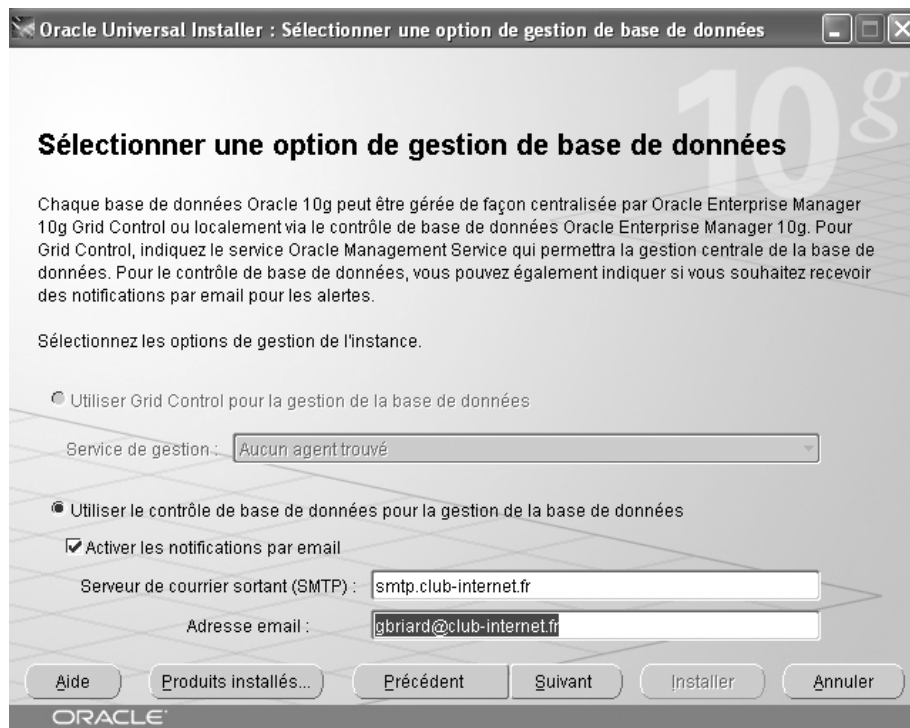


Figure 13-12

Option de stockage de la base de données

Les fichiers composant la base de données peuvent être placés sur des systèmes de stockage différents :

- un système de fichier Windows « normal », qu'il soit de type FAT (déconseillé), NTFS (conseillé), avec des disques en configuration miroir, RAID ou encore un SAN ;

- un système de stockage propre à Oracle : ASM (*Automatic Storage Management*) ;
- un accès direct aux disques : le raw device.

Le choix du système de stockage est fondamental. Il conditionne une partie des performances d'Oracle. S'il est facile d'implémenter tel ou tel choix lors de la configuration du système Windows et de l'installation d'Oracle, il est toujours lourd et complexe de revenir sur cette décision. Faites le bon choix ! Pour vous aider, ces options sont présentées plus en détail au chapitre 27, *Configurer les disques pour Oracle*.

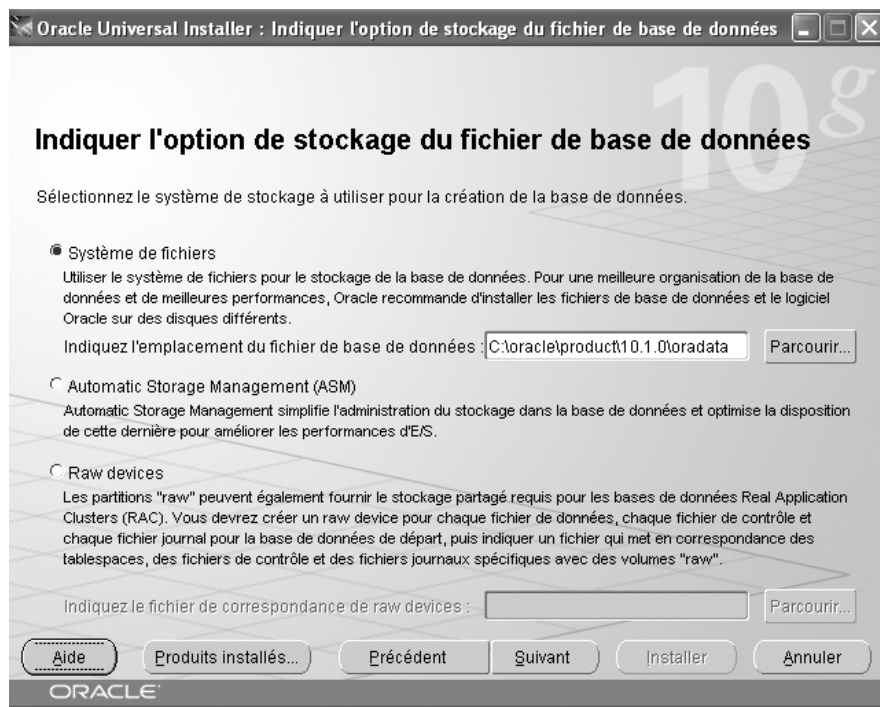


Figure 13-13

Options de sauvegarde et de restauration

L'Installer propose de sauvegarder automatiquement la base de données dans un répertoire nommé le *flash recovery area*. Dans ce cas, Oracle Enterprise Manager exécutera chaque nuit un script Oracle Recovery Manager pour copier :

- lors de la première exécution l'ensemble de la base de données ;
- à chaque nouvelle exécution le « delta » des différences sous forme d'une sauvegarde incrémentale.

Le choix effectué à cette étape peut être modifié par la suite à partir d'Oracle Enterprise Manager.

Tout système de sauvegarde doit être choisi, mis en place et **obligatoirement vérifié**. Le chapitre 26, *La sauvegarde d'une base Oracle 10g* aborde ces points.

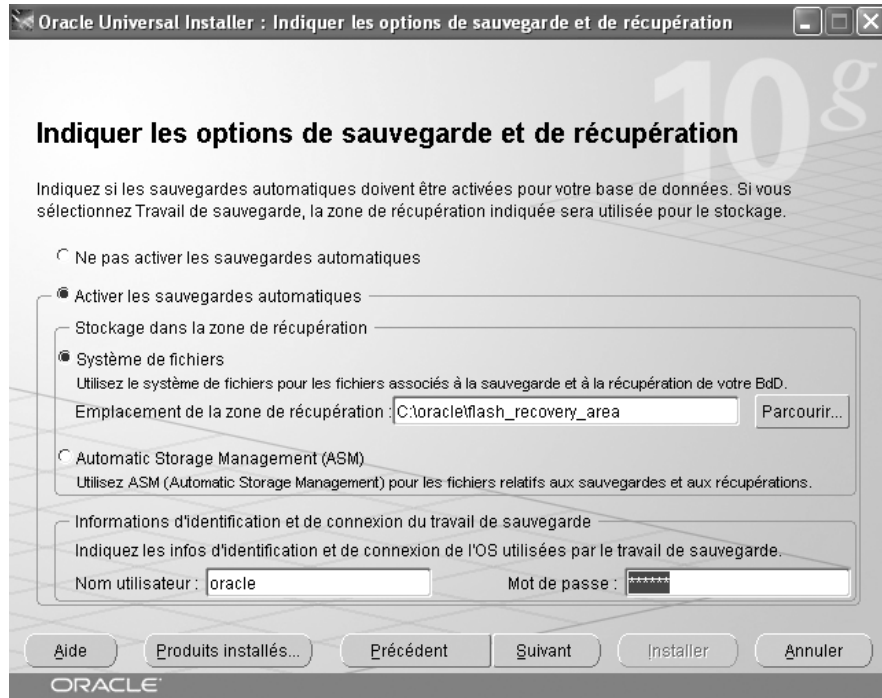


Figure 13-14

Indiquer les mots de passe

Pour des raisons de sécurité, Oracle impose maintenant le changement des mots de passe des comptes SYS, SYSTEM et SYSMAN durant la création de la base.

Dans les versions précédentes d'Oracle, les utilisateurs SYS et SYSTEM possédaient par défaut les mots de passe CHANGE_ON_INSTALL et MANAGER. C'est une bonne chose pour la sécurité d'imposer leur changement à cette étape car 80 % des bases Oracle installées possédaient les mots de passe d'origine.

Définissez des mots de passe « sérieux » et conservez soigneusement leurs valeurs dans une enveloppe de secours scellée, accessible aux seuls utilisateurs autorisés.

Tous les utilisateurs autres que SYS et SYSTEM qui n'ont pas changé de mot de passe seront verrouillés.

Par facilité, j'ai essayé de tricher : l'écran suivant m'a rappelé à l'ordre ! C'est un point très positif pour la sécurité.

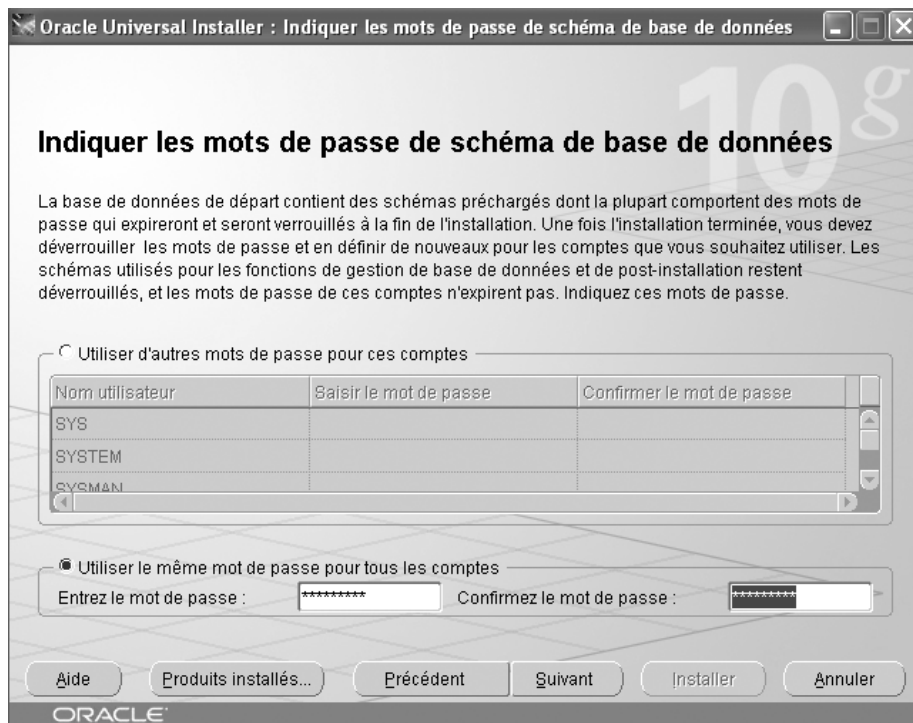


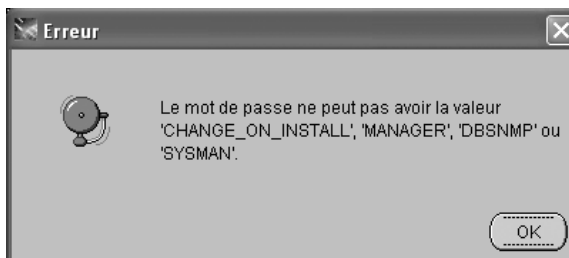
Figure 13-15

Il faut un « vrai » mot de passe

Tous les choix sont maintenant effectués.

Figure 13-16

Résumé de l'installation



L'écran précédent résume toutes les options choisies. Il ne reste plus qu'à lancer l'installation.

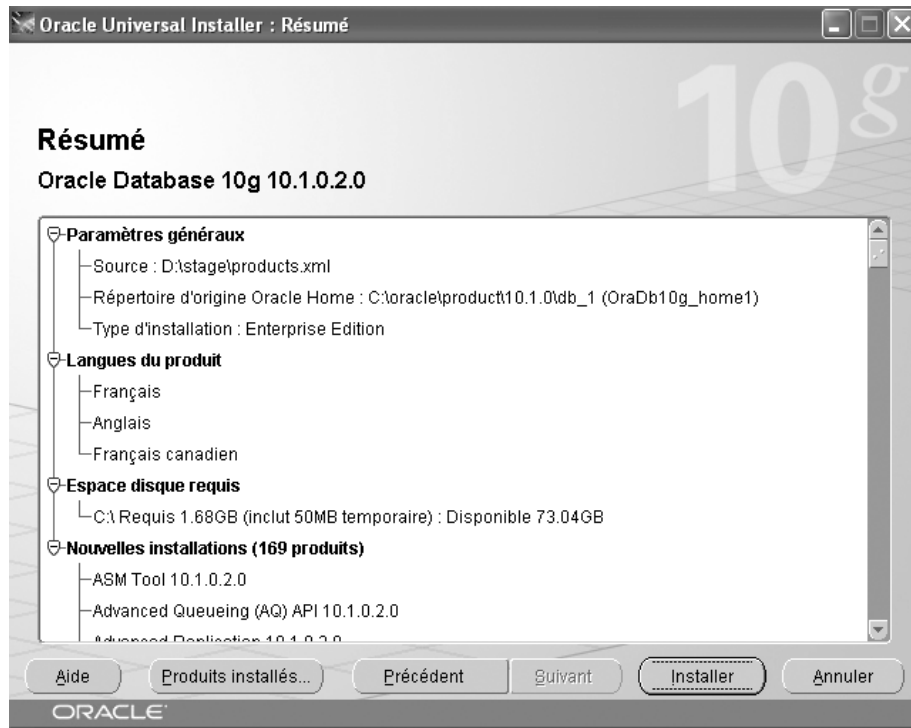


Figure 13-17
Installation des logiciels

L'installation des logiciels d'Oracle Database 10g, leur configuration et la création de la base TEST s'exécutent. La durée de cette opération dépend de la puissance de l'ordinateur. Un journal de l'installation est créé en <X> :Program Files\Oracle\Inventory\logs.

Si vous êtes sous Windows XP Professional, les couches logicielles nécessaires à l'installation d'un cluster ne sont pas disponibles. Vous rencontrerez alors un message d'erreur. N'en tenez pas compte et poursuivez. Le service OracleCSService n'étant pas supporté sous Windows XP Professional, il convient de ne pas lancer ce service en automatique depuis l'interface de gestion des services.

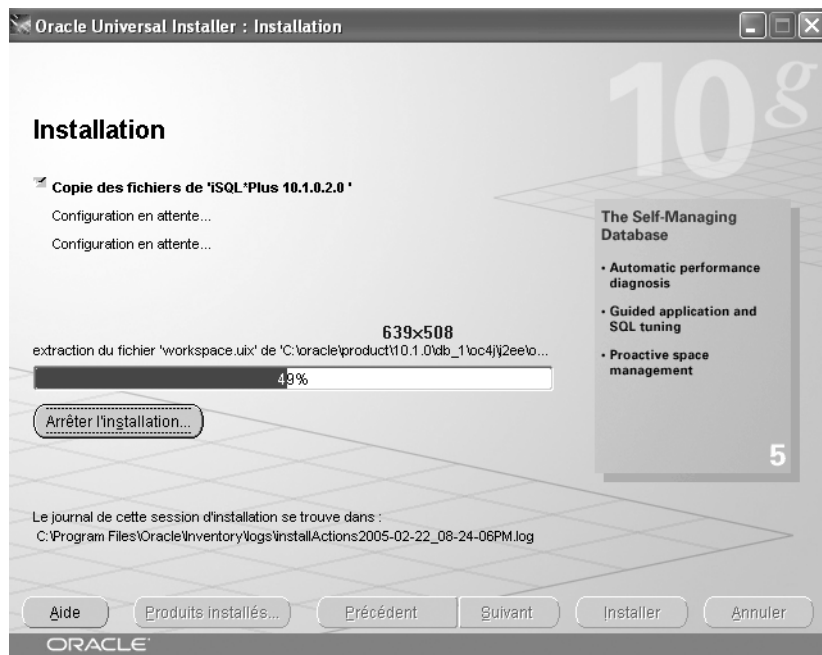


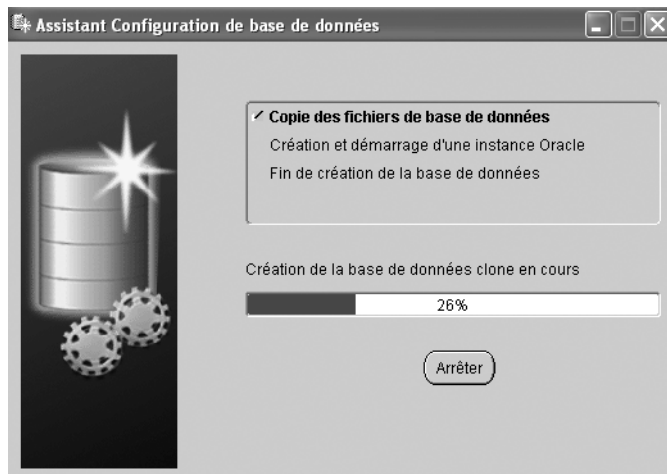
Figure 13-18

Création de la base de données

Nous avons choisi précédemment une base « préinstallée ». Les fichiers de la base sont donc copiés depuis le CD-Rom vers le disque dur, puis la configuration est automatiquement adaptée.

Figure 13-19

Assistants de configuration



Une fois les logiciels Oracle installés, les opérations de configuration s'enchaînent automatiquement :

- un assistant de configuration d'Oracle Net est lancé (il est détaillé au chapitre 15, *Oracle Net, le middleware Oracle*) ;
- l'assistant Oracle Database Configuration Assistant est lancé pour compléter automatiquement la configuration de la base.

Suivant le type d'installation (logiciel seul, base de données préinstallée, etc.), les étapes présentées sur cette figure seront différentes.

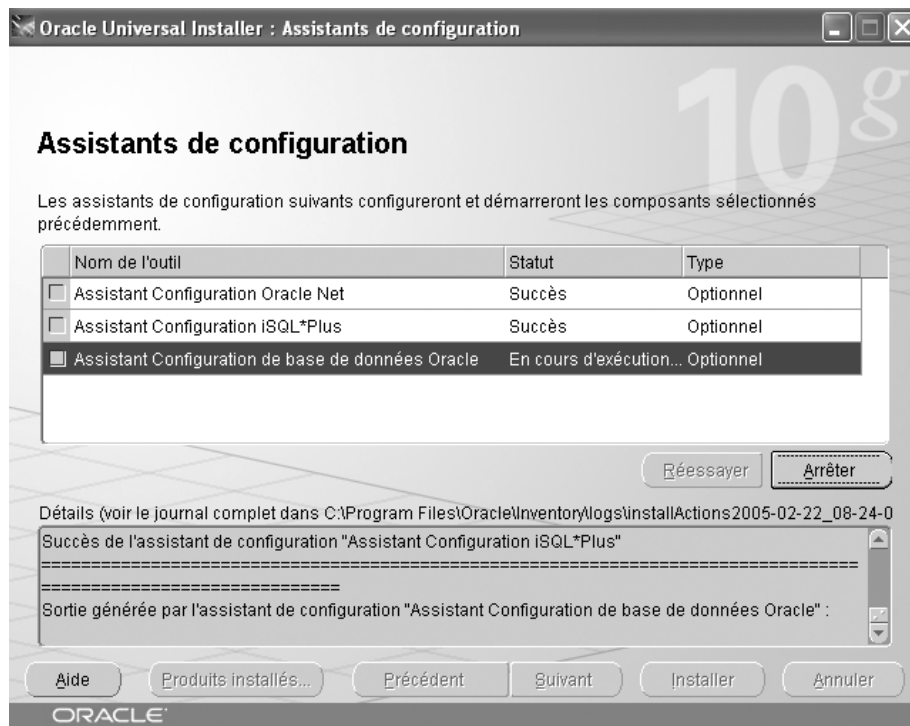


Figure 13-20

Copie de la base de données

L'identification de la base créée ainsi que l'URL d'accès à OEM (*Oracle Enterprise Manager*) sont rappelés.

Comme nous avons demandé la création d'exemples, les différents mots de passe doivent impérativement être créés : tous les comptes sont verrouillés par défaut. Encore un bon point pour la sécurité !

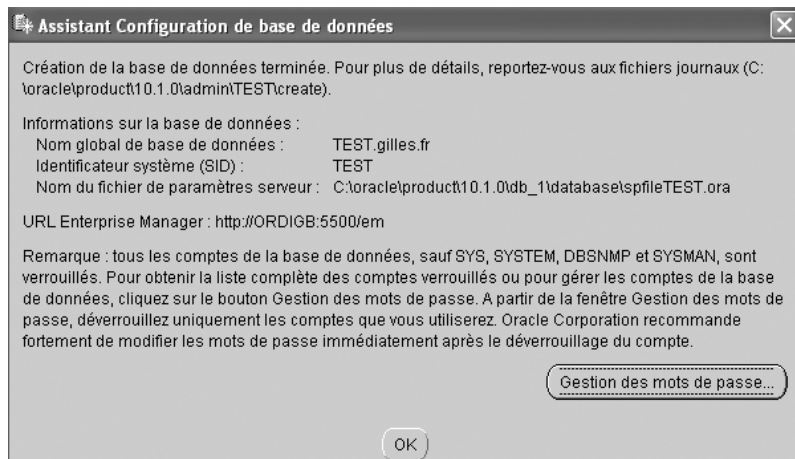


Figure 13-21

Changement des mots de passe

Dans l'exemple, les comptes HR et SCOTT sont déverrouillés et des mots de passe sont indiqués. Ces modifications peuvent être ultérieures, avec l'ordre SQL ALTER USER.

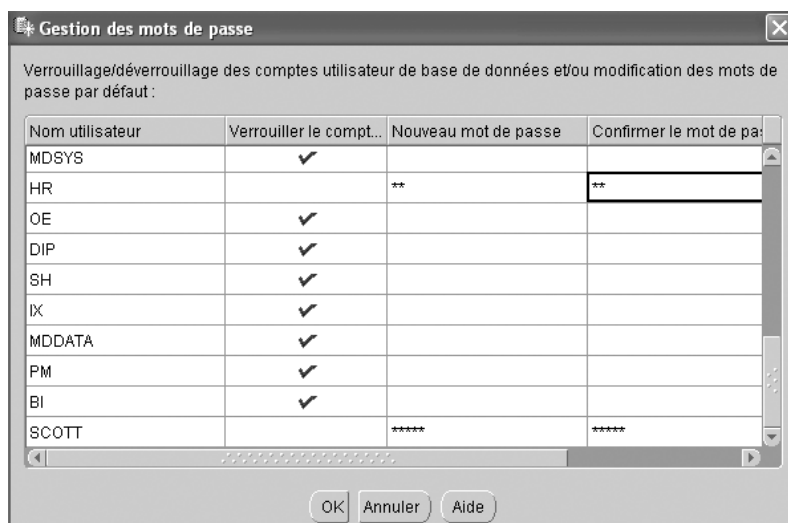


Figure 13-22

Fin de l'installation

Conservez les adresses (et numéros de port) des applications JEE installées :

Application installée	Exemple de valeur
Oracle Ultra Search	http://<nom du serveur>:5620/ultrasearch
Administration d'Ultra Search	http://<nom du serveur>:5620/ultrasearch/admin
iSQL*Plus	http://<nom du serveur>:5560/isqlplus
Oracle Enterprise Manager	http://<nom du serveur>:5500/em

Il est possible de modifier ultérieurement les numéros de port donnés par défaut.

Les logiciels et la base de données TEST sont maintenant installés et configurés. Félicitations, vous êtes le nouvel administrateur Oracle sur Windows ! Voyons maintenant comment vérifier le bon résultat de l'installation.

Connexion à la base initiale à l'aide de SQL*Plus

SQL*Plus est l'outil mode caractère « historique » d'Oracle. Rassurez-vous, l'outil iSQL*Plus propose des fonctions similaires (mais pas aussi complètes) via une interface Web.

Pour vérifier l'installation et le bon fonctionnement de la base TEST pré-installée, assurez-vous que le Service OracleServiceTEST fonctionne puis lancez SQL*Plus : Démarrer > Programmes > Oracle – Ora10gDb-Home<X> > Application Development > SQL*Plus :

```
# connectez-vous sous l'utilisateur Oracle SYSTEM
nom d'utilisateur: SYSTEM
mot de passe: <mot de passe de SYSTEM>

# la commande suivante récupère la date de la base TEST
select sysdate from dual ;
exit
```

Pour vérifier le fonctionnement du lien client/serveur Oracle Net, assurez-vous que le Service OracleOraDb10g_home1TNSListener fonctionne puis lancez SQL*Plus : Démarrer > Programmes > Oracle – Ora10gDb-Home<X> > Application Development > SQL*Plus :

```
# connexion de test en mode client-serveur
# connectez-vous sous l'utilisateur Oracle SYSTEM

nom d'utilisateur: SYSTEM
mot de passe: <mot de passe de SYSTEM>
chaîne centrale : TEST # la valeur correspond au SID de votre base

# la commande suivante récupère la date de la base TEST
select sysdate from dual ;
exit
```

Vérification d'Oracle iSQL*Plus

Pour utiliser iSQL*Plus, vérifiez que le service OracleOraDb10g_home1iSQL*Plus est démarré. Ouvrez ensuite un navigateur à l'adresse <http://<ordinateur ou adresse IP>:5560/isqlplus>. La fenêtre de connexion s'affiche.

En cas de problème d'accès au serveur, vérifiez le paramétrage réseau du serveur ou de la station évoqué au début de ce chapitre.



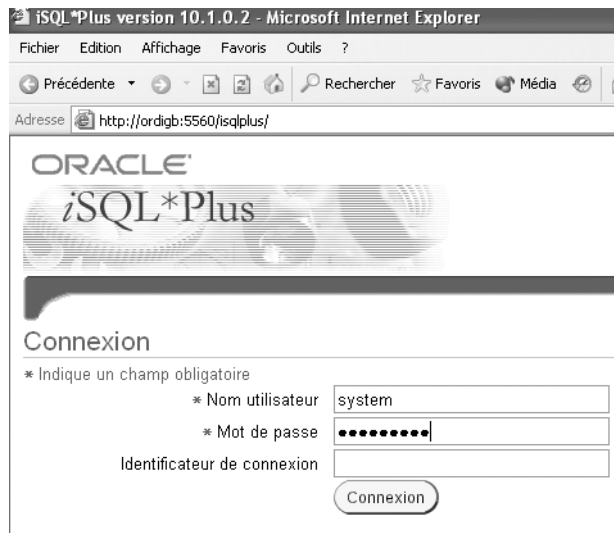
Figure 13-23

Connexion à iSQL*Plus

Une fois l'utilisateur et son mot de passe saisis, vous accédez à iSQL*Plus (voir figure 13-24).

Vérification d'Oracle Enterprise Manager

Pour vérifier le fonctionnement de la version locale d'Oracle Enterprise Manager, ouvrez un navigateur à l'adresse <http://<ordinateur ou adresse IP>:5500/em>. La fenêtre de connexion s'affiche (voir figure 13-25).

Figure 13-24*L'interface d'iSQL*Plus***Figure 13-25***Connexion à Enterprise Manager*

En cas de problème d'accès au serveur, vérifiez le paramétrage réseau du serveur ou de la station évoqués au début de ce chapitre.

Une fois l'utilisateur SYSTEM et son mot de passe saisis, vous accédez à Enterprise Manager.

Nous vous conseillons d'utiliser la console Web d'OEM pour découvrir le contenu et le paramétrage de cette base.

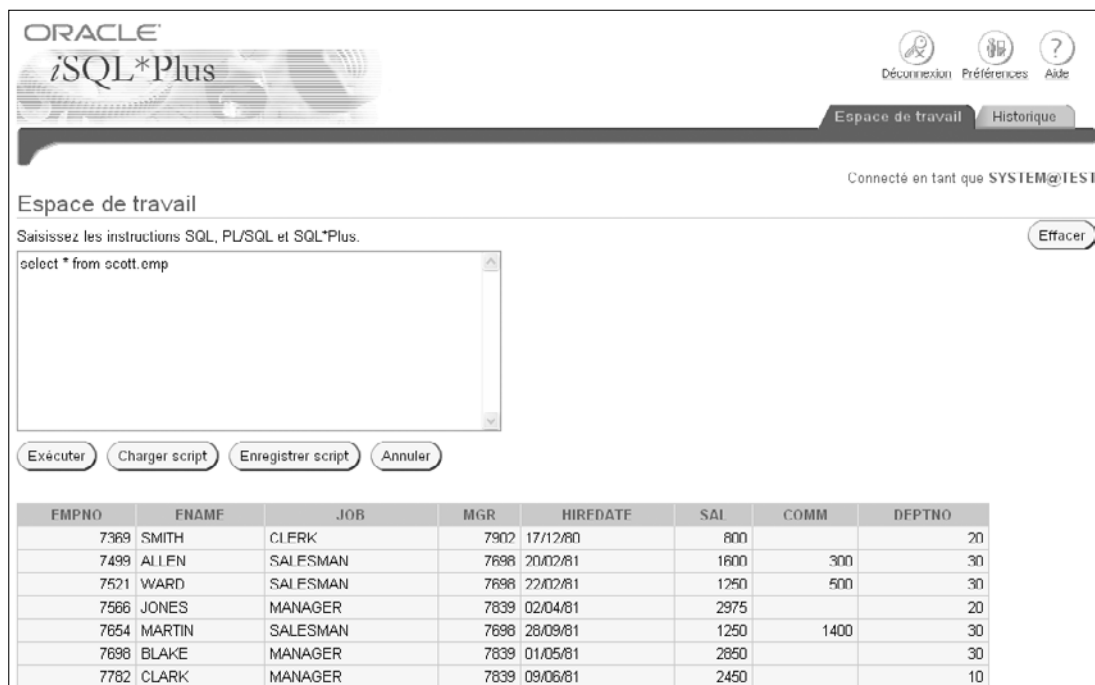


Figure 13-26

La console Web d'OEM

Les défauts de la base initiale

La base de données TEST qui est copiée depuis le CD-Rom présente certains défauts :

- certaines options importantes n'ont pas été proposées ;
- vous ne maîtrisez pas réellement ce qu'elle contient ;
- ses caractéristiques vous sont inconnues.

Bref, c'est flou ! Nous vous conseillons d'utiliser cette base uniquement pour de petits tests et de rapidement créer d'autres bases de données suivant la méthode décrite au chapitre 14, *Création d'une base Oracle 10g*.

Si l'installation échoue

Si vous souhaitez supprimer uniquement une base de données installée par Oracle, vous pouvez utiliser l'assistant DBCA (*Database Configuration Assistant*) présenté au chapitre suivant.

Si vous souhaitez supprimer physiquement tous les fichiers installés par Oracle, y compris les exécutables, vous pouvez effectuer cette opération manuellement. La plus grande prudence s'impose si une base de production existe !

1. Connectez-vous sous l'utilisateur Windows possédant les droits *Administrateur*.
2. Utilisez DBCA pour supprimer toutes les bases de données existant sur le serveur. Il ne doit plus avoir de *Service* Windows associé aux bases.
3. Lancez Oracle Universal Installer pour supprimer tous les logiciels installés.
4. Supprimez les fichiers restants en effaçant les répertoires <X>:\oracle ainsi que les fichiers de données des bases.
5. Supprimez toutes les clés de registre situées en HKLM\SOFTWARE\ORACLE.
6. Supprimez les services Oracle<XXX> situés dans la base de registre en HKLM\SYSTEM\ControlSet001\Services.
7. Supprimez le répertoire <X>:\Program Files\Oracle.
8. Supprimez les variables d'environnement faisant référence à Oracle.
9. Supprimez les entrées dans les menus.

La machine est maintenant « nettoyée », vierge de tout logiciel Oracle.

Automatiser l'installation

L'outil graphique OUI (*Oracle Universal Installer*) permet une installation graphique, mais cela impose d'être derrière son écran et de répondre une par une aux options d'installation proposées. Rapide et convivial pour des installations peu fréquentes, ce mode n'est absolument pas adapté aux installations répétitives.

À cet égard, OUI propose un mode « silencieux », par lequel un fichier préalablement paramétré indique toutes les options choisies. Cela permet d'intégrer l'installation d'Oracle 10g dans des scripts, pratique chère aux administrateurs. Différents fichiers très bien documentés sont proposés par Oracle. Présents sur le CD-Rom, ils possèdent l'extension *.rsp* (fichier réponse). Adaptez-les à vos besoins et lancez-les par la commande :

```
■ runInstaller [-silent] -responsFile nom_du_fichier.rsp
```

Le fichier *silentInstall.log* permet de suivre le déroulement de l'installation lorsque l'option *-silent* est utilisée.

Conclusion sur cette étape

L'installation des logiciels Oracle 10g est une étape importante. Une installation initiale correcte, respectant la procédure décrite et les raisons qui la motivent, facilitera le travail au quotidien de vos administrateurs Windows-Oracle.

Étape 4 : post-installation

Le rôle de l'administrateur Windows-Oracle ne s'arrête pas une fois Oracle 10g installé et une base de données initiale créée. Il faut maintenant mettre en place un environnement d'exploitation :

- créer les autorisations d'accès aux fichiers et aux répertoires ;
- lancer des travaux à partir du Job Scheduler ;
- adapter si besoin la configuration client-serveur initiale (chapitre 15, *Oracle Net, le middleware Oracle*) ;
- démarrer/arrêter vos bases en même temps que Windows ;
- mettre en place des sauvegardes.

Les autorisations d'accès aux fichiers et aux répertoires

Lors de l'installation, Oracle Universal Installer affecte des permissions et des droits aux répertoires situés sous ORACLE_BASE\ORACLE_HOME, soit C:\oracle\product\10.1.0\db_1 dans cet exemple. Les droits concédés sont :

- Administrateur : contrôle total ;
- System : contrôle total ;
- Utilisateur autorisé : lecture, exécution, liste des fichiers.

D'une façon identique, lorsqu'une base est créée lors de l'installation ou via le Database Configuration Assistant (DBCA, voir le chapitre suivant), le contrôle total est accordé aux utilisateurs Administrateur et System à :

- ORACLE_BASE\admin\nom_de_la_base (administration de la base) ;
- ORACLE_BASE\oradata\nom_de_la_base (fichiers de la base) ;
- ORACLE_HOME\ORACLE_BASE\database (fichier SPFILESID.ORA)

Lancer des travaux à partir du Job Scheduler

Le service Job Scheduler associé à chaque base permet de lancer des travaux ordonnés depuis Oracle Enterprise Manager. Ses travaux pouvant être lancés ordinateur en veille, sans aucun utilisateur connecté, vous devez définir le compte utilisateur sous lequel les travaux s'effectueront (figure 13-27).

Les variables « par défaut »

Pour qu'un utilisateur Windows puisse accéder et utiliser une base Oracle, il est indispensable de fixer au préalable un certain nombre de variables. Par exemple, si plusieurs bases de données sont installées, il faut que l'utilisateur « pointe » vers la base désirée.

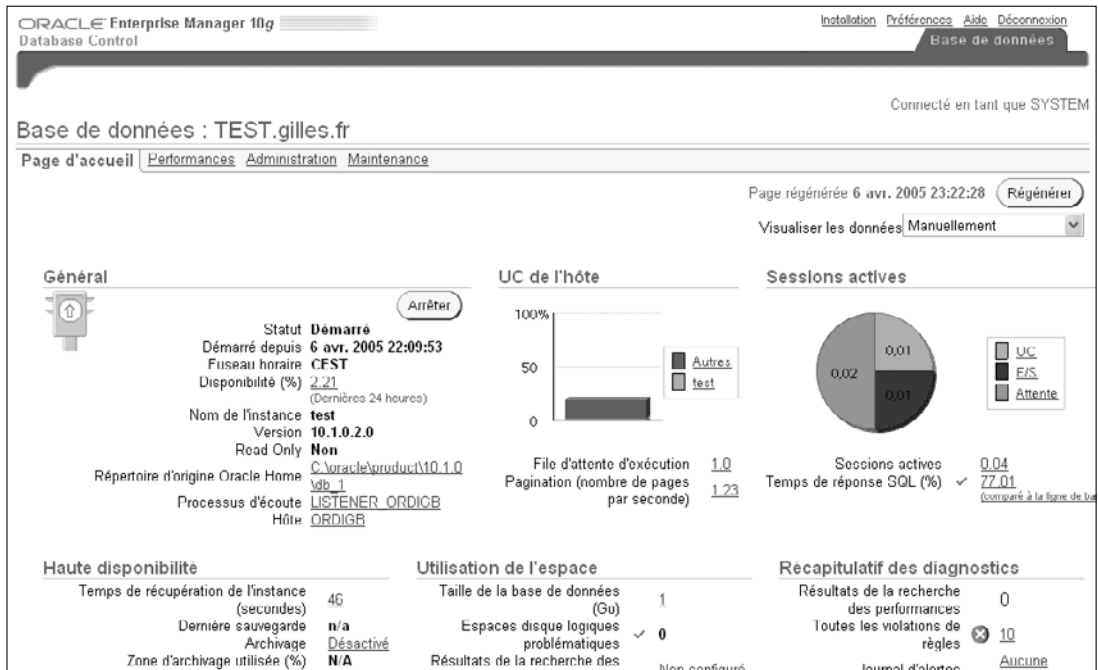


Figure 13-27

Configurer le service Job Scheduler

L'utilitaire *Oracle Administration Assistant for Windows* permet de modifier ces variables. Ce point est abordé au chapitre 5, *Fonctionnement d'Oracle 10g sous Windows*.

Automatiser le démarrage et l'arrêt des bases Oracle

Les utilisateurs peuvent accéder facilement aux bases Oracle. Encore faut-il qu'elles soient lancées ! C'est pourquoi nous vous proposons de lier le démarrage et l'arrêt des bases Oracle au démarrage et à l'arrêt du système Windows (ce point est abordé au chapitre 5, *Fonctionnement d'Oracle 10g sous Windows*).

Rien de plus simple sous Windows, il suffit de démarrer automatiquement le service qui existe pour chaque base de données. Vous pouvez paramétrer le démarrage automatique depuis le gestionnaire de service Windows ou plus simplement à partir de l'*Administration Assistant for Windows* d'Oracle, accessible depuis le menu Programmes > Démarrer > Oracle-Ora10gDb-Home<X> > Configuration and Migration Tools > Administration Assistant for Windows. Cet outil se présente sous la forme d'une console MMC (*Microsoft Management Console*).

Les options de démarrage sont accessibles par un « clic droit » sur le nom de la base de données.

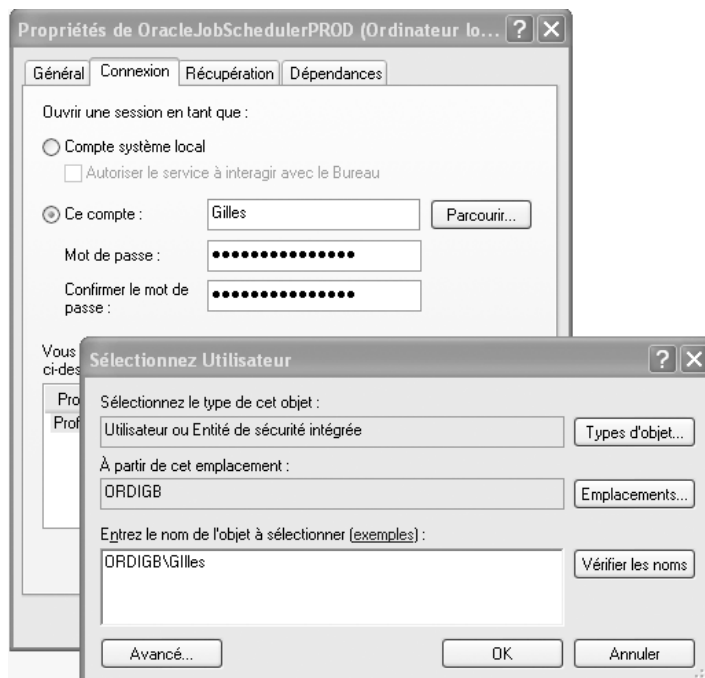


Figure 13-28
Options de démarrage

Les options de démarrage d'Oracle accessibles concernent aussi bien le démarrage du service Windows que celui de l'instance Oracle.

Ces étapes sont détaillées aux chapitres 5 et 10, *Fonctionnement d'Oracle 10g sous Windows* et *Démarrer et arrêter une base Oracle 10g*.

Automatiser les sauvegardes

Oracle est installé et une base de données fonctionne. Il est temps d'effectuer une sauvegarde globale et de préparer votre stratégie de sauvegarde. N'oubliez pas : c'est en cas de problème que l'on distingue les « amateurs » des « professionnels ».

Résumé de l'installation

Ce chapitre vous a guidé dans l'installation d'Oracle 10g sous Windows. Celle-ci est simple : peu de prérequis sont exigés, l'automatisation est excellente et vous obtenez une base Oracle installée et fonctionnelle en un temps très réduit. Nous détaillons toutes ces étapes.

Nous abordons la configuration d'autres logiciels livrés avec Oracle 10g dans des chapitres spécifiques. C'est le cas, par exemple, pour Oracle Net Services et ses interfaces Java JDBC. Chaque fois qu'un point est détaillé dans cet ouvrage, nous indiquons le chapitre correspondant.

Pendant l'installation, une base Oracle par défaut a été créée. Comme elle présente un certain nombre d'inconvénients, nous vous conseillons de passer au chapitre suivant, *Création d'une base Oracle 10g* pour créer une nouvelle base de données qui servira de support à l'ensemble des tests.

14

Création d'une base Oracle 10g

Cette étape peut faire partie du processus d'installation d'Oracle 10g ou être abordée ultérieurement. Nous en faisons une étape à part pour bien dissocier l'installation du logiciel Oracle de la création de nouvelles bases de données.

Le logiciel *Oracle Database Configuration Assistant* (écrit en Java) offre une interface graphique conviviale pour vous guider. Cet utilitaire, réclamé par tous les administrateurs Oracle depuis des années, est disponible depuis Oracle8. Il s'avère très bien conçu et sa dernière version prend en compte toutes les nouveautés apportées par Oracle 10g.

Si l'interface graphique est un plus, la création d'une base reste néanmoins complexe. C'est pourquoi toutes les étapes de la création d'une nouvelle base sont particulièrement détaillées.

Pourquoi créer une nouvelle base de données ?

C'est l'une des premières questions à se poser. Avez-vous besoin d'une nouvelle base de données sur votre machine ? Nous vous proposons quelques éléments d'appréciation.

Une nouvelle application

Si une nouvelle application doit être hébergée sur votre serveur, vous n'avez pas forcément besoin de lui créer une nouvelle base de données. En effet, Oracle 10g vous permet d'avoir, au sein de la même base de données, des environnements complètement imperméables. Par exemple, pour une même application, les environnements de développement et de

production peuvent cohabiter dans la même base. Le nom de l'utilisateur et son mot de passe orienteront vers l'environnement cible (comme si vous aviez deux bases distinctes). De plus, une seule base de production facilite votre administration et vos sauvegardes.

La puissance de votre serveur

Si plusieurs bases de données fonctionnent simultanément sur la même machine, les performances diminuent car les ressources utilisées augmentent. Il est alors intéressant de limiter les ressources redondantes. Si vous multipliez les bases, il existera autant de zones mémoire réservées qu'il y aura de bases de données lancées.

Par ailleurs, il est parfois difficile d'optimiser un serveur qui possède plusieurs bases en fonctionnement. Il faut alors s'assurer qu'une surcharge ponctuelle sur l'une des bases n'induit pas une diminution des performances pour les autres.

La nécessité de tester

Il est évident que, pour vous exercer aux tâches d'administration, nous vous recommandons fortement de créer une base personnelle afin de vous entraîner et de mettre au point vos procédures de sauvegarde et toute manipulation technique un peu « pointue ».

Ce point est particulièrement intéressant sous Windows car Oracle s'installe aussi bien sous une version Windows Workstation que sur un Windows Server.

Des applications différentes

Bien que cela soit à éviter autant que possible, certaines circonstances imposent la mise en place de plusieurs bases fonctionnant simultanément. C'est notamment le cas si vous disposez de progiciels différents, nécessitant des environnements hétérogènes et dont les mises à jour de versions ne sont pas synchronisées.

Dans l'exemple que nous prenons, le choix de créer ou non la base de données vous est offert au final. De plus, vous pouvez conserver la création de la base sous forme de scripts SQL que nous commenterons au fur et à mesure.

Étapes préliminaires

Le respect des règles OFA assure une grande souplesse d'administration des différentes versions Oracle et de leurs bases de données.

Les règles OFA ont été abordées au chapitre 9, *Les fichiers d'une base Oracle 10g*. Pour rappel, voici l'organisation recommandée si deux bases fonctionnent simultanément. La base existante est TEST, la nouvelle sera PROD.

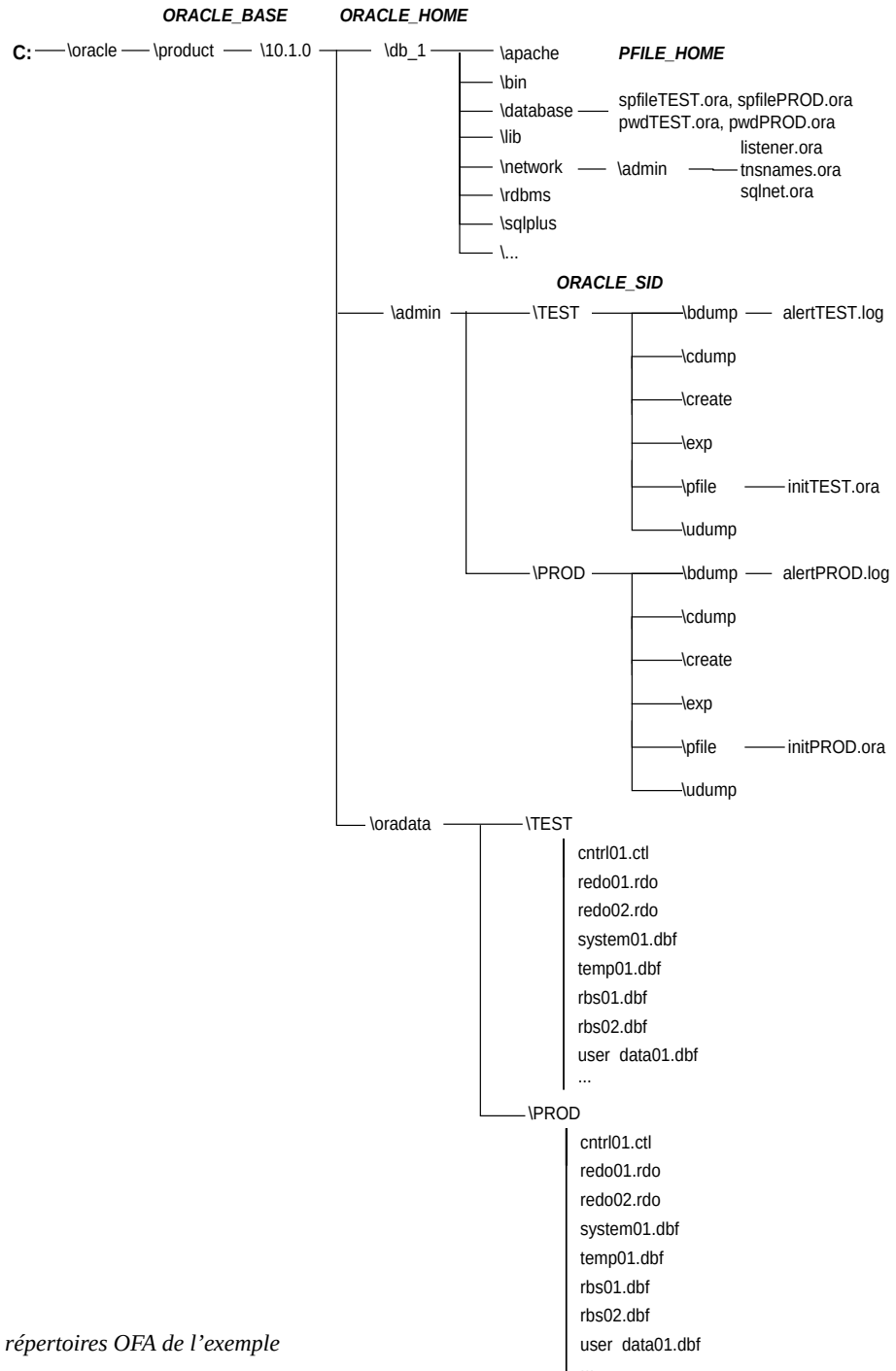


Figure 14-1

Principe des répertoires OFA de l'exemple

Étapes de la création d'une base Oracle 10g

Les nouveautés apportées par Oracle Database Configuration Assistant

La dernière version de DBCA (*DataBase Configuration Assistant*) arrive à maturité et apporte de nouvelles options très intéressantes. Avant de créer la base, DBCA permet d'enregistrer les scripts SQL de création. Consulter en détail ces scripts est un excellent exercice d'apprentissage. Les conserver est indispensable comme documentation.

La nouveauté la plus importante concerne l'apparition de *modèles*. Une fois les caractéristiques d'une base saisies, vous avez la possibilité de les enregistrer sous forme d'un modèle codé au format XML. Ce modèle est alors proposé dès le prochain lancement de DBCA. Comme le format XML est un format « texte », vous pouvez déplacer des modèles de serveur à serveur.

Cette notion de modèle, associée à la génération de l'ensemble des scripts SQL nécessaires à la création d'une base, est très souple pour concevoir et mettre au point une nouvelle base de données.

On peut aussi créer des modèles à partir d'une base de données existante, y compris la définition de toutes les structures de données. Pour cela, DBCA se connecte à la base existante et recueille toutes les informations nécessaires.

DBCA permet de créer non seulement les fichiers qui composent la base de données mais aussi tout l'environnement d'exploitation, c'est-à-dire un ensemble de répertoires et de fichiers destinés à faciliter l'administration des bases en proposant une organisation identique pour chaque base. Ces répertoires sont créés automatiquement si vous décidez de créer la base de données à la fin de l'opération, sinon il vous faudra les créer vous-même.

Juste un regret, DBCA ne respecte pas exactement les règles OFA (*Optimal Flexible Architecture*) proposées par Oracle. Il faut parfois modifier l'emplacement des fichiers composant la base de données. Ce n'est pas grave, l'ensemble reste parfaitement cohérent et DBCA constitue une aide précieuse pour tout administrateur Oracle.

Les étapes effectuées par DBCA

Les étapes réalisées par DBCA pour créer une nouvelle base de données correspondent à celles qui seraient effectuées manuellement. Elles se déroulent selon l'ordre suivant :

1. choix d'un nom significatif pour l'instance (SID) ;
2. création du service Windows OracleServiceSID ;
3. choix de la méthode qui sera utilisée par l'administrateur Oracle pour créer, démarrer et arrêter la base ;
4. création du fichier d'initialisation de la base ;
5. connexion à l'instance ;

6. démarrage de l'instance en mode NOMOUNT ;
7. exécution de l'ordre CREATE DATABASE ;
8. création de tablespaces supplémentaires ;
9. lancement des scripts de création des catalogues des options à installer ;
10. création d'un fichier d'initialisation persistant ;
11. modification des mots de passe par défaut de la base ;
12. sauvegarde de la base ;
13. actions de post-installation.

Tous ces points sont commentés dans ce chapitre. Seuls les deux derniers points ne sont pas effectués par DBCA et doivent être effectués manuellement.

L'utilisation de DBCA

Connectez-vous sous l'utilisateur Windows qui a été utilisé pour installer Oracle. Dans notre cas, c'est le compte Administrateur. L'utilitaire Oracle Database Configuration Assistant est accessible depuis le menu Démarrer > Programmes > Oracle – *Oradb10g home1* > Configuration and Migration Tools.

L'aide en ligne accessible dès le premier écran de l'assistant est très bien conçue. Utilisez-la sans réserve.



Figure 14-2

Écran d'accueil d'Oracle Database Configuration Assistant

Comme vous le constatez, l'interface de DBCA est proposée en français.

Figure 14-3

*Création d'une nouvelle
base de données*



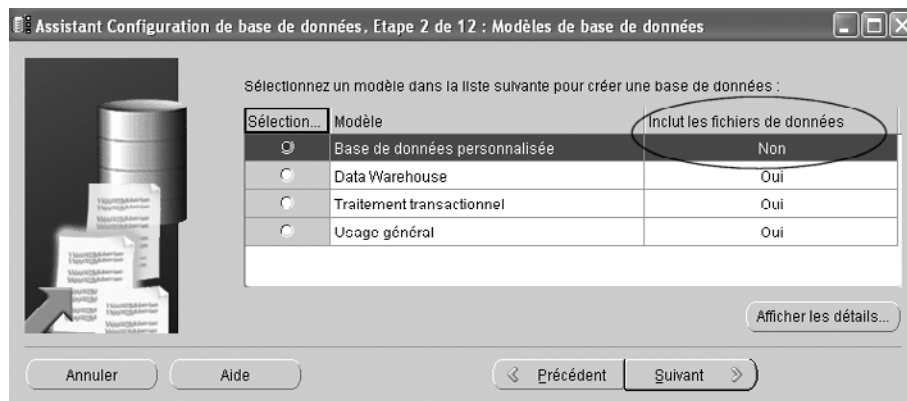
Plusieurs options sont proposées :

- Créer une nouvelle base de données ;
- Configurer une base de données existante, c'est-à-dire ajouter des options qui n'ont pas été initialement installées (Oracle JVM, Oracle Spatial, etc.) ;
- Supprimer une base de données ;
- Gérer des modèles permettant d'enregistrer sur disque la définition de la base codée dans un format XML. Cette option propose aussi de récupérer automatiquement la structure d'une base de données existante, qu'elle soit locale ou distante. Cela permet de réaliser très rapidement des copies de bases existantes.

Choisissez la création d'une nouvelle base Oracle 10g.

Figure 14-4

*Choix d'un
modèle initial*



Quatre modèles sont proposés dans cet exemple. Pour trois d'entre eux, l'option *Inclut les fichiers de données* est *oui*. Cela signifie que la base de données sera construite rapidement par copie de fichiers préexistants. Cette option est la plus rapide mais vous disposerez de moins d'options de configuration. Comme nous l'avons déjà utilisée lors de l'installation initiale d'Oracle au chapitre précédent, nous allons créer une nouvelle base de données, sans copie des fichiers de données livrés par Oracle.

Avant d'utiliser le modèle *Base de données personnalisée*, affichons le détail de ses caractéristiques. Les modèles codés au format XML sont placés en <X :>\oracle\product\10.1.0\db_1\assistants\dbca\templates*.dbt. Vous pouvez les copier d'un serveur vers un autre, mais Oracle déconseille d'en modifier le contenu.

Figure 14-5

*Caractéristiques du modèle
New Database*

Base de données personnalisée
Utilisez ce modèle pour créer une base de données personnalisée.

Options communes

Option	Sélectionné
Oracle JVM	true
Oracle Intermedia	true
Oracle Text	true
Oracle XML DB	true
Oracle OLAP	true
Oracle Spatial	true
Oracle Data Mining	true
Oracle Ultra Search	true
Oracle Label Security	false
Exemples de schémas	false
Référentiel Enterprise Manager	true

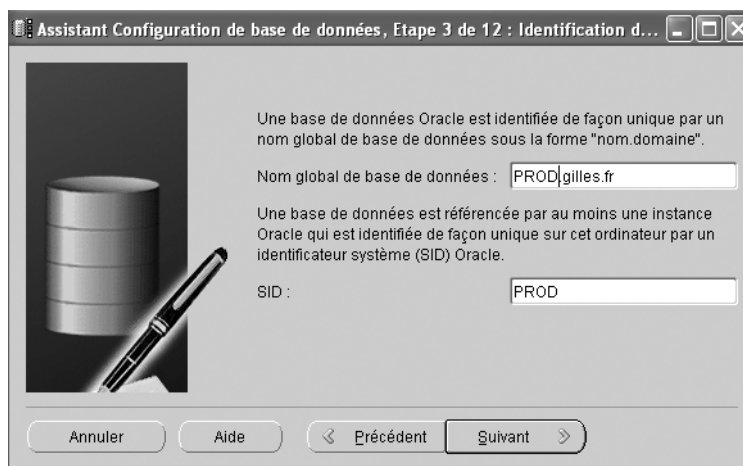
Paramètres d'initialisation

Nom	Valeur
background_dump_dest	{ORACLE_BASE}\admin\{DB_UNIQUE_NAME}\bdump
compnathle	10 1 0 0

L'affichage des caractéristiques des différents modèles permet de choisir rapidement le modèle le plus proche de la base de données à créer.

Figure 14-6

Identification de la base de données



Les paramètres suivants précisent le nom global de la base et celui de l'instance :

- Nom global de la base de données ou GLOBAL_DBNAME : identifiant unique de la base de données. Dans cet exemple, il se compose du nom de la base ou DB_NAME (PROD), concaténé avec le DB_DOMAIN (gilles.fr). On le retrouvera dans le fichier d'initialisation ainsi que dans le script de création de la base ;
- SID : c'est le nom donné à votre instance, c'est-à-dire votre base de données « en marche », avec en plus les exécutables lancés et une zone mémoire réservée pour l'instance. C'est le SID (*System Identifier*) qui différencie deux instances fonctionnant sur une même machine. On retrouvera aussi l'instance dans le nom des processus Oracle en marche ainsi que dans le nom des services Windows lancés. Il se nomme PROD dans cet exemple.

Pour plus de simplicité, **nom de base = nom d'instance**, codé sur 4 caractères majuscules.

Dans cet exemple, le nom global de la base de données est PROD.gilles.fr et son SID est PROD (figure 14-7).

L'écran précédent permet de choisir le mode d'administration de la nouvelle base : soit depuis une version locale d'Oracle Enterprise Manager, soit depuis une version centralisée nécessitant le relais d'un agent local pour communiquer. Ce point est détaillé au chapitre 25, *Oracle Enterprise Manager* (figure 14-8).

Fidèle à sa nouvelle pratique de sécurité, Oracle vous oblige à choisir de « vrais » mots de passe dès cette étape (figure 14-9).

Conservez les mots de passe dans un endroit sûr accessible aux seules personnes autorisées.

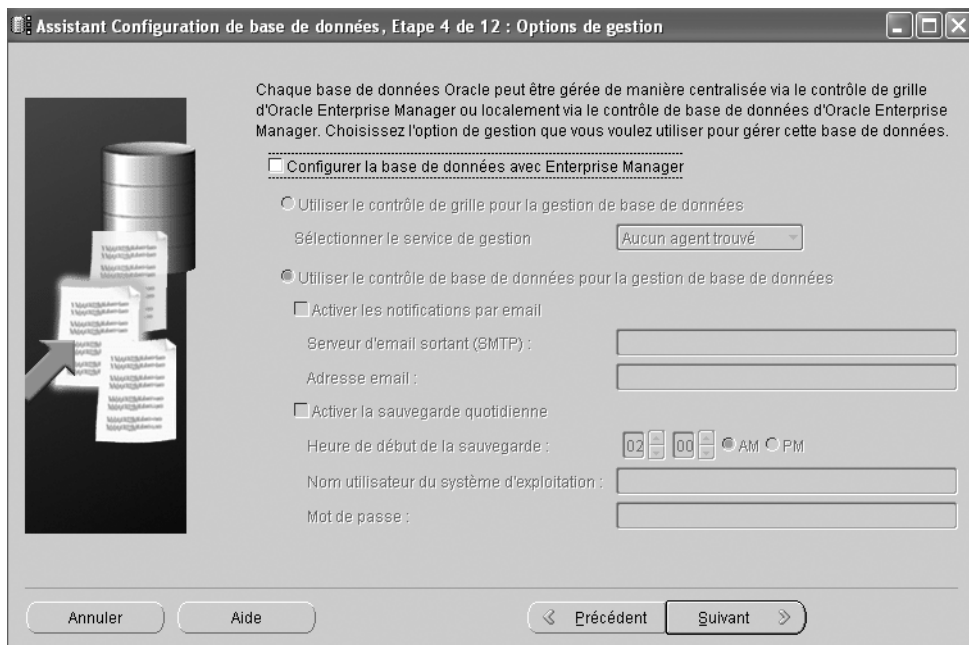


Figure 14-7
Choix d'administration

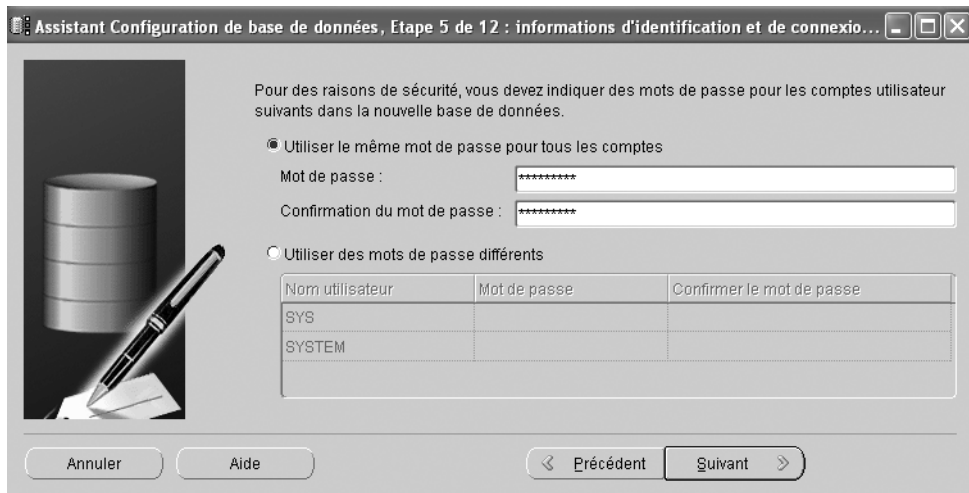


Figure 14-8
Initialisation des mots de passe

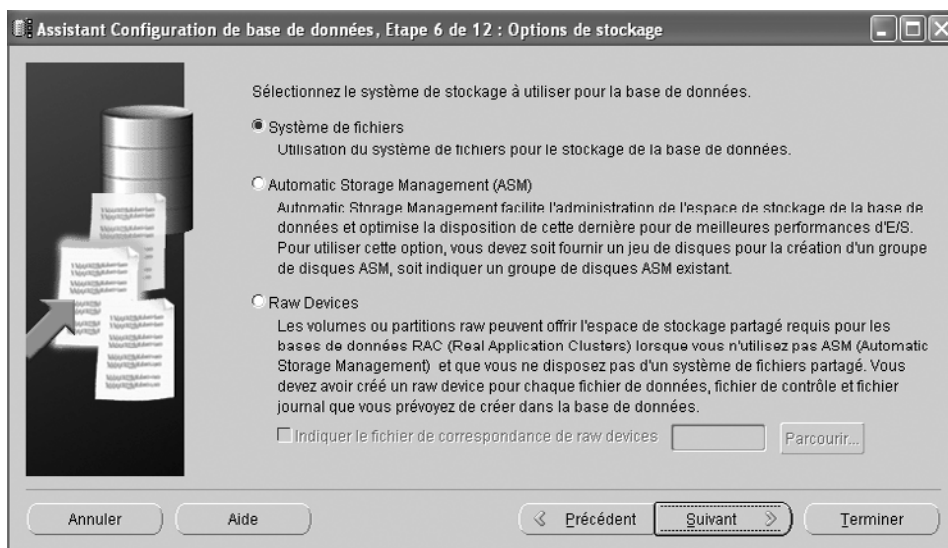


Figure 14-9

Choix du type de stockage disque

Choisissez le type de stockage adapté pour la base. Ce point est détaillé au chapitre 27, *Configurer les disques pour Oracle*.

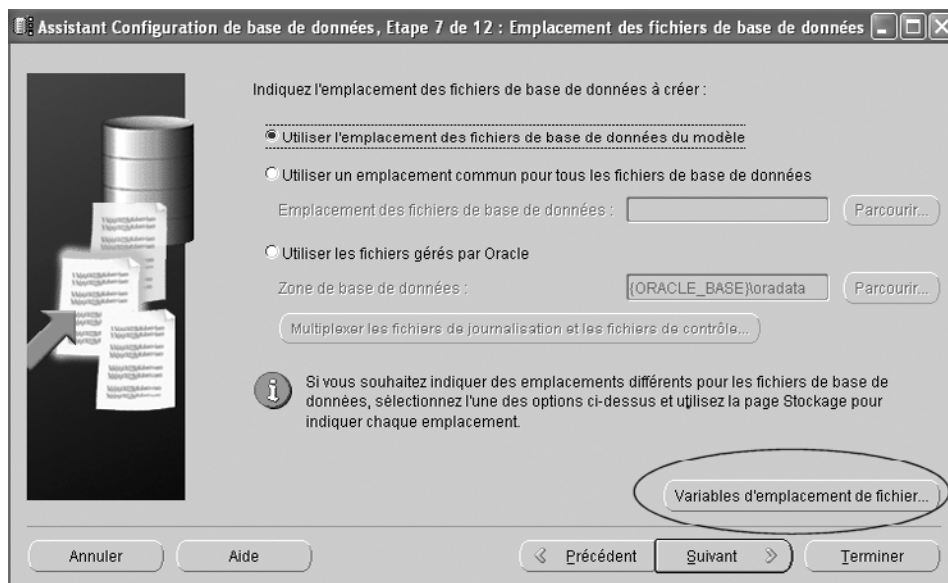


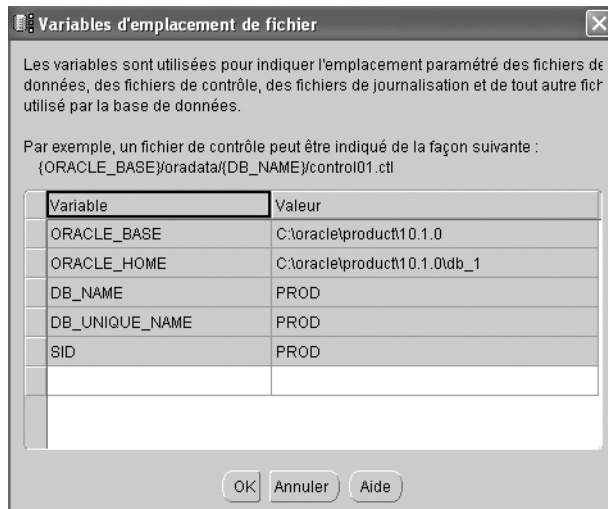
Figure 14-10

Choix des répertoires d'installation de la base

Étudions maintenant l'écran accessible depuis le bouton *Variables d'emplacement de fichiers*. Vous accéderez à cet écran dans plusieurs étapes de cet assistant.

Figure 14-11

Variables d'emplacement de fichiers



Les variables ORACLE_BASE et ORACLE_HOME proviennent des variables définies pendant l'installation du logiciel Oracle et sont conservées dans la base de registre en HKEY_LOCAL_MACHINE\SOFTWARE\ORACLE\KEY_OraDb10g_home1.

Ces valeurs sont décrites au chapitre précédent ainsi qu'au chapitre 5, *Fonctionnement d'Oracle 10g sous Windows*.

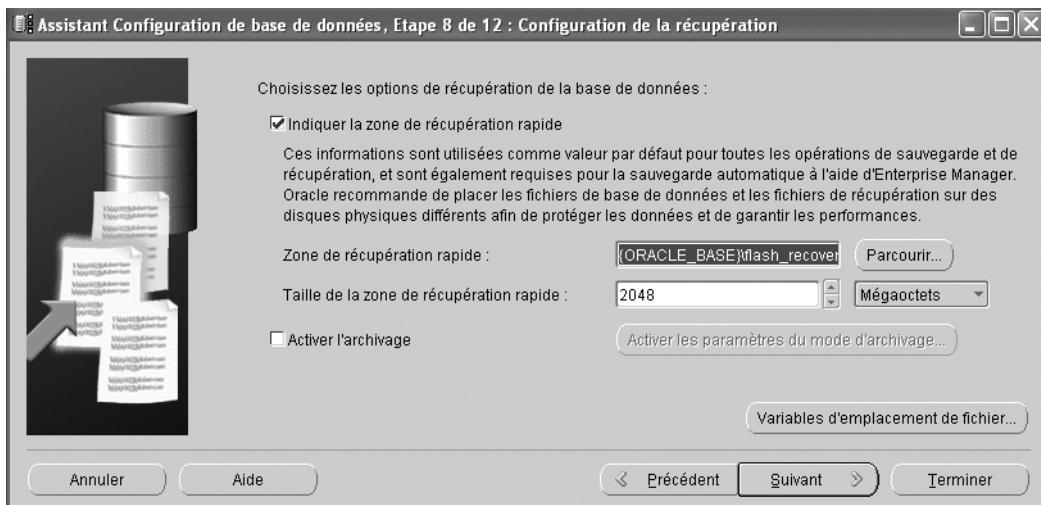


Figure 14-12

Zone de restauration

La zone de restauration concerne l'emplacement cible où seront effectuées les sauvegardes automatiques d'Oracle orchestrées par Oracle Enterprise Manager.

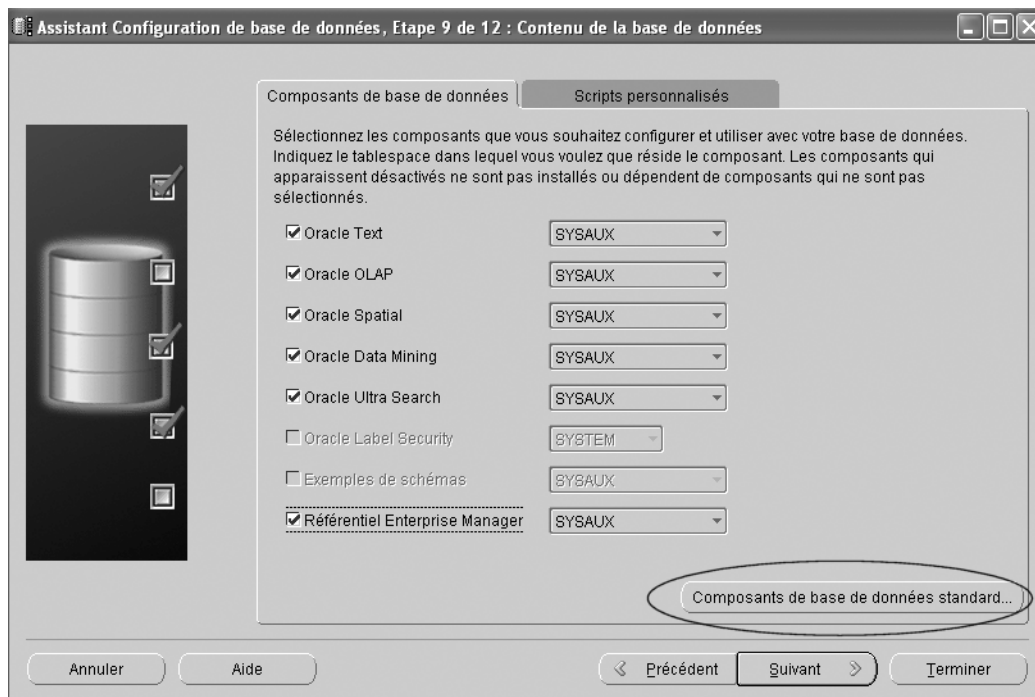


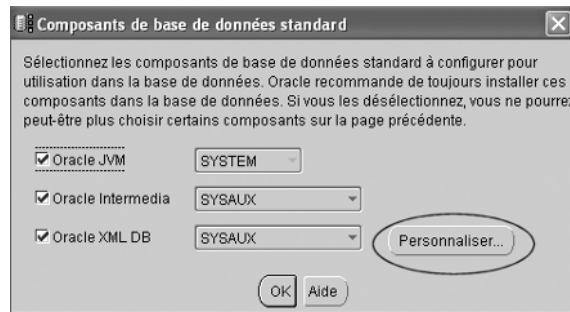
Figure 14-13

Choix des options

Choisissez toutes les options proposées, même si elles ne sont pas nécessaires à notre test. Vous observerez que cela ajoute des fichiers de commande SQL dans les fichiers d'ordres SQL générés. Comme l'installation de certaines options est très consommatrice de ressources, vous avez la possibilité de revenir sur ces choix. Parmi les options les plus consommatrices, Oracle JVM et Intermédia sont accessibles depuis l'option *Fonctions de base de données standard*.

Remarquez que, dans un second onglet, l'assistant propose d'ajouter vos propres scripts à l'issue de la création de la base.

Figure 14-14
Options supplémentaires



Comme nous sélectionnons systématiquement toutes les options, passons en revue la personnalisation de XML DB.

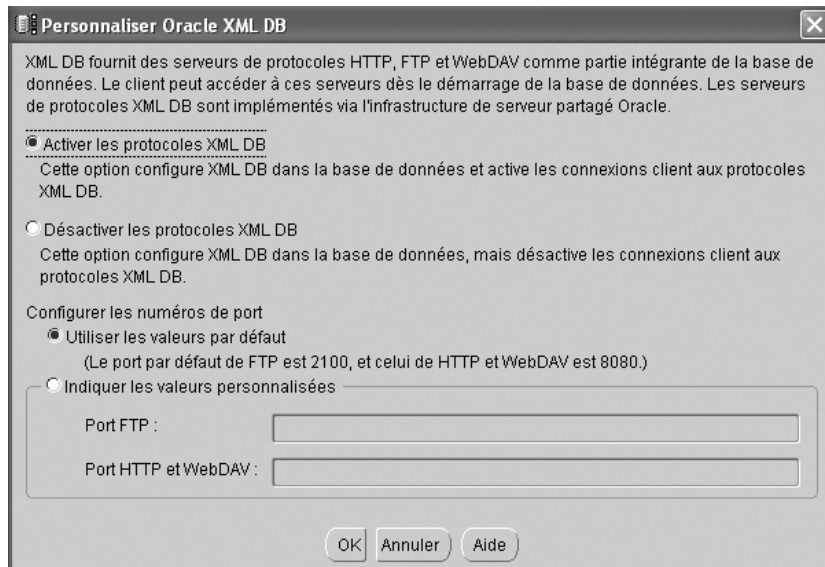


Figure 14-15
Options de XML DB

Profondément remanié dans Oracle 10g, XML DB ouvre de larges extensions XML. Destiné à communiquer via le réseau Internet, il convient de paramétrer les protocoles qui serviront à accéder à XML DB. Parmi ceux-ci : HTTP, FTP (protocole de transfert de fichiers) ou WebDAV.

Le Web-based Distributed Authoring and Versioning (WebDAV) est un protocole d'échange de fichiers et de gestion des révisions. Il étend les capacités du protocole Web HTTP afin de permettre aux utilisateurs de modifier des documents sur un serveur distant mais aussi de collaborer durant le cycle de vie de ces documents. Comme pour tous les protocoles réseaux, il convient de préciser les numéros de port IP. Nous conservons les valeurs proposées par défaut : 2100 pour FTP et 8080 pour HTTP et WebDAV.

Nous conservons toutes ces options pour étudier les scripts SQL nécessaires à leur installation.

Le fichier d'initialisation

Les différents onglets de l'étape suivante de l'assistant permettent de configurer des variables figurant dans les fichiers d'initialisation `init.ora` et `spfile.ora`. Elles sont modifiables ultérieurement et leur emplacement est indiqué en fin de paragraphe.

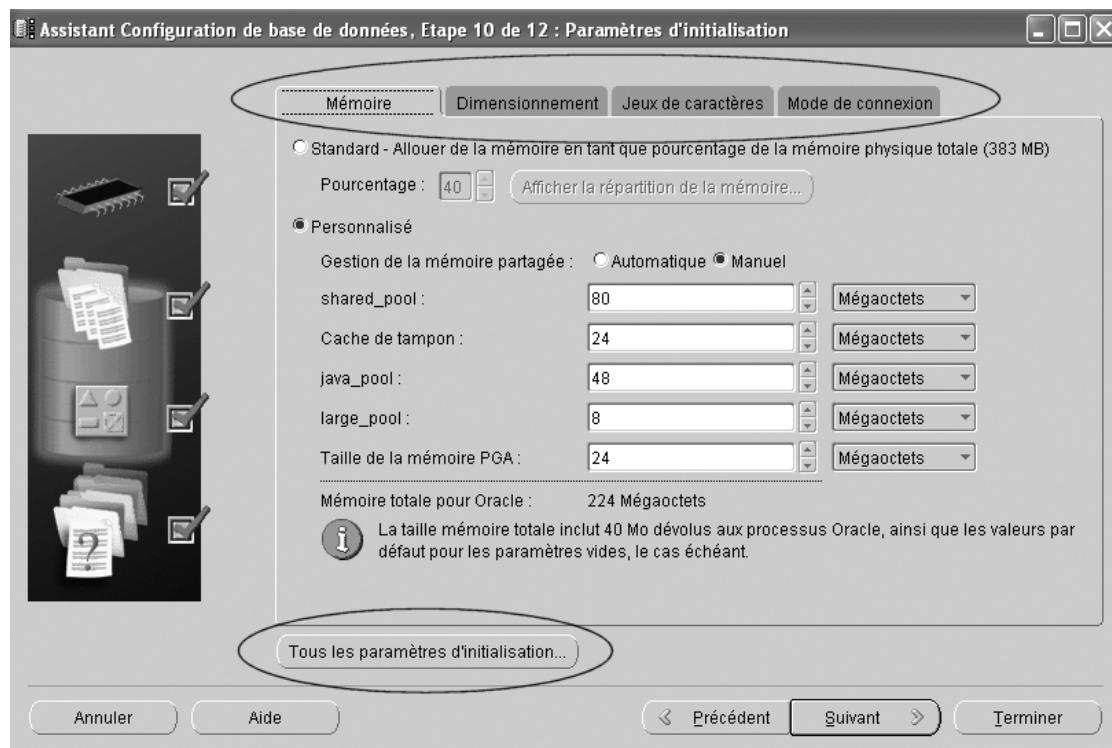


Figure 14-16

Paramètres d'initialisation

Les valeurs proposées tiennent compte des options précédemment choisies. Par exemple, la décision d'installer Oracle JVM impose la valeur JAVA_POOL. Cette valeur est nécessaire pour installer Oracle JVM mais peut être diminuée après cette étape.

Les paramètres saisis dans cet écran ont un impact direct sur la mémoire nécessaire : elle est de 224 Mo dans cet exemple. À vous de l'adapter suivant la configuration matérielle dont vous disposez.

Les caractéristiques mémoire d'Oracle sont précisées au chapitre 8, *L'utilisation de la mémoire par Oracle 10g*.

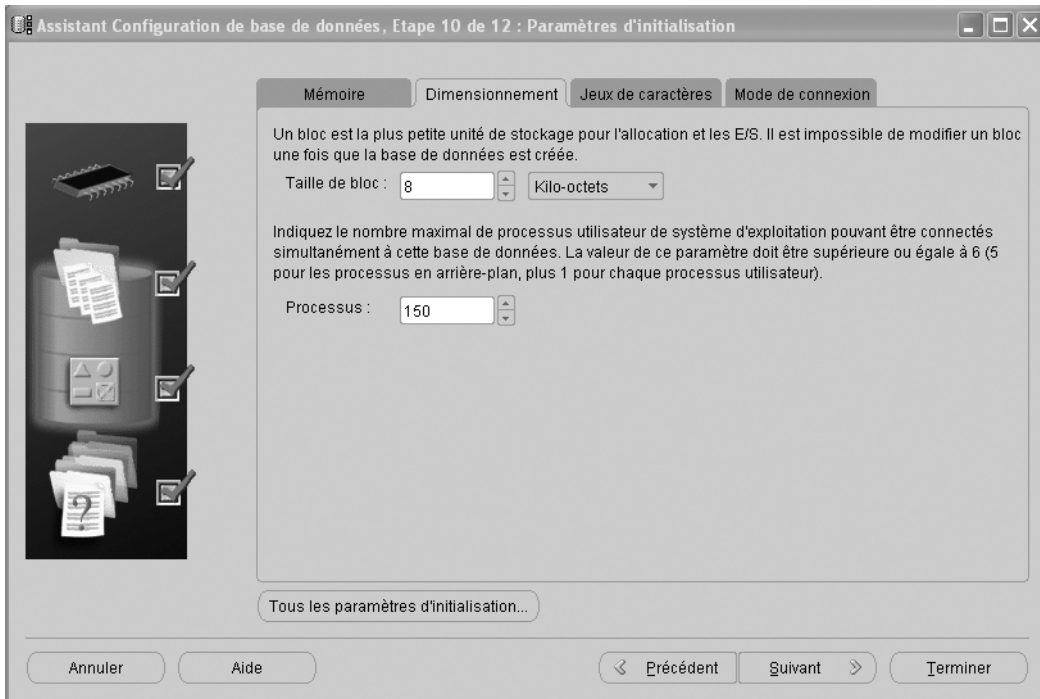


Figure 14-17

Dimensionnement de la base

La taille du bloc Oracle ou DB_BLOCK_SIZE est figée à la création de la base de données. Ce paramètre détermine la taille initiale pour le formatage interne des fichiers de données, ainsi que celle de la zone mémoire réservée à l'instance. Il précise aussi la dimension des échanges entre la mémoire et les disques. Ce paramètre est très important pour les performances. Ce point est développé au chapitre 9, *Les fichiers d'une base Oracle 10g* et nous en reparlerons au chapitre 28, *Optimisation et performances*.

Pour Windows, nous vous conseillons de positionner DB_BLOCK_SIZE à 8 Ko, ce qui est une valeur très polyvalente.

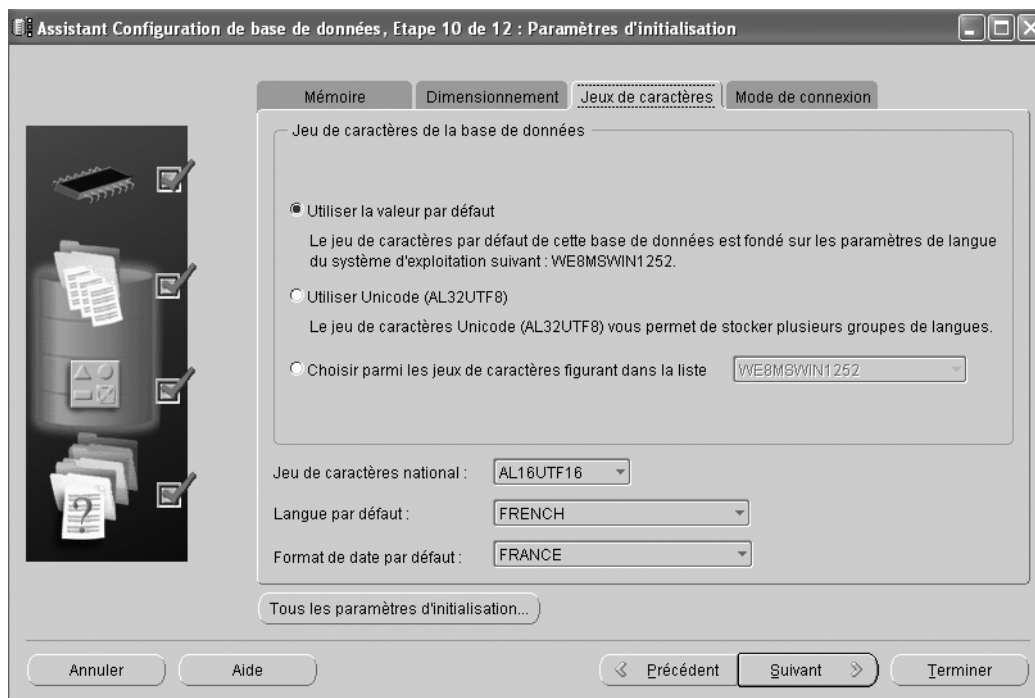


Figure 14-18

Jeux de caractères

Le jeu de caractères ou `CHARACTER_SET` est abordé dans l'annexe *NLS, le support des langues par Oracle et l'euro*. C'est un paramètre qu'il convient de choisir avec soin, car il est difficilement modifiable après la création de la base de données. Il détermine l'ensemble des caractères que l'on peut coder sous un même *jeu de caractères*. Cela va des Kanji japonais au russe !

Pour les caractères français et européens, nous vous conseillons l'option standard Windows `WE8MSWIN1252`, car son codage sur 8 bits permet de stocker nos accents et caractères spécifiques, y compris le sigle Euro. La langue anglaise (dénuée d'accents et du sigle Euro) se contente d'un jeu de caractères 7 bits, comme l'`US7ASCII` souvent proposé par défaut.

Pour supporter le français ET le caractère Euro (€), les seuls jeux de caractères compatibles sont : `WE8ISO8859P15`, `WE8MSWIN1252`, Unicode (`AL32UTF8` ou `UTF8`). Choisissez obligatoirement l'un d'entre eux pour que la base de données soit « euro-compatible ».

Le jeu de caractères national ou NLS_CHARACTER_SET est utilisé comme jeu de caractères pour les colonnes de tables de type NCHAR, NCLOB, NVARCHAR2, qui sont rarement rencontrés.

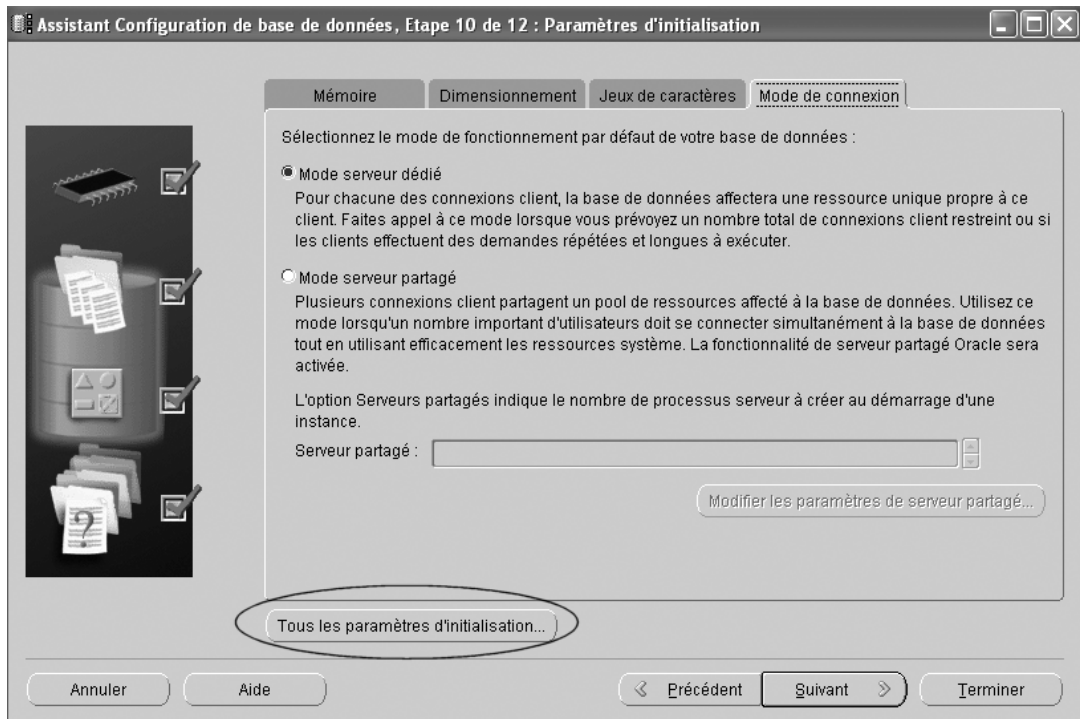


Figure 14-19

Mode de connexion Oracle Net

La base de données peut fonctionner en client-serveur suivant deux modes : le mode *serveur dédié* et le mode *serveur partagé*. Leur fonctionnement respectif est décrit au chapitre 7, *Les processus d'Oracle 10g*. Nous vous recommandons de toujours opter pour le mode *serveur dédié*.

L'option *Tous les paramètres d'initialisation* donne accès à l'écran suivant qui donne toutes les valeurs qui figureront dans le fichier d'initialisation `init.ora`. Les valeurs proviennent des saisies effectuées dans les écrans précédents. Une « croix » indique les valeurs différentes des valeurs par défaut. La description en français de chaque paramètre est disponible.



Figure 14-20

Paramètres du fichier d'initialisation

Utilisation du fichier d'initialisation

Le fichier d'initialisation est nécessaire pour créer la base et lors de chaque démarrage de l'instance. Il détermine par exemple la taille mémoire allouée à celle-ci, décide si les fichiers journaux doivent être archivés ou encore indique les fichiers de contrôle utilisés par l'instance.

Selon les règles OFA, le fichier d'initialisation est placé en <X:>\oracle\product\10.1.0\admin\SID\pfile sous le nom init.ora (par exemple C:\oracle\product\10.1.0\admin\PROD\pfile\init.ora pour l'instance PROD). C'est un fichier ASCII consultable et modifiable avec un éditeur de texte tel que Notepad. N'utilisez pas Word ou Wordpad pour le modifier car ils ajoutent des caractères parasites de présentation.

Pour démarrer la base, le fichier d'initialisation est indiqué dans la commande :

```
sqlplus /nolog
connect system/<mot de passe> as SYSDBA
startup pfile C:\oracle\product\10.1.0\admin\PROD\pfile\init.ora
```

Indiquer systématiquement où se situe le fichier d'initialisation est contraignant. Pour faciliter le démarrage à distance des bases, Oracle9i a introduit la notion de fichier d'initialisation persistant ou fichier SPFILE. Il est créé à partir d'un fichier init.ora donné en référence et il est conservé sous forme binaire en <X:>\oracle\product\10.1.0\db_1\database (suivant les systèmes, il est parfois conservé sous forme de fichier caché). Si aucun fichier d'initialisation n'est précisé, le fichier d'initialisation persistant est alors utilisé en priorité. Pour l'instance PROD, le fichier persistant SPFILEPROD.ora sera créé en C:\oracle\product\10.1.0\db_1\database. Ces points sont détaillés au chapitre 10, *Démarrer et arrêter une base Oracle 10g*.

Point sur la création de la base

Effectuons un point d'étape sur les éléments renseignés. Ont été précisés aux étapes précédentes :

- le nom de la base et celui de l'instance ;
- les options à installer dans la base (Oracle JVM, les exemples, etc.) ;
- des paramètres utilisés pendant l'ordre de création de la base ;
- des paramètres figurant dans le fichier d'initialisation de l'instance.

Par contre, aucune caractéristique concernant les tablespaces et les fichiers associés n'a été demandée. C'est l'objet de la suite de l'installation.

Les fichiers composant la base

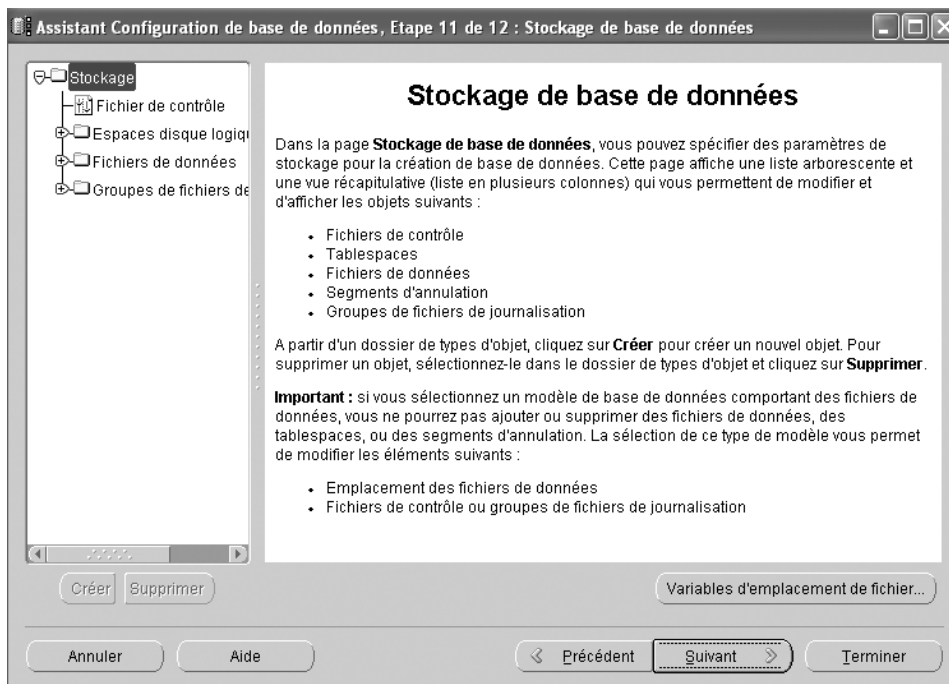
Les navigateurs et les onglets de l'écran suivant présentent les caractéristiques des trois types de fichiers composant une base Oracle 10g :

- les fichiers de contrôle (ou de commande) ;
- les tablespaces ;
- les fichiers associés à chaque tablespace ;
- les segments d'annulation ;
- les fichiers redo-log (ou journaux).

Il vous revient de déterminer leur taille, leur emplacement et leurs caractéristiques par défaut. Ces fichiers sont décrits au chapitre 9, *Les fichiers d'une base Oracle 10g*.

Dans tous les écrans qui suivent, veillez à respecter les normes OFA.

Figure 14-21
Stockage



Les caractéristiques des fichiers composant une base Oracle sont présentées au chapitre 9, *Les fichiers d'une base Oracle 10g*.

Les fichiers de contrôle

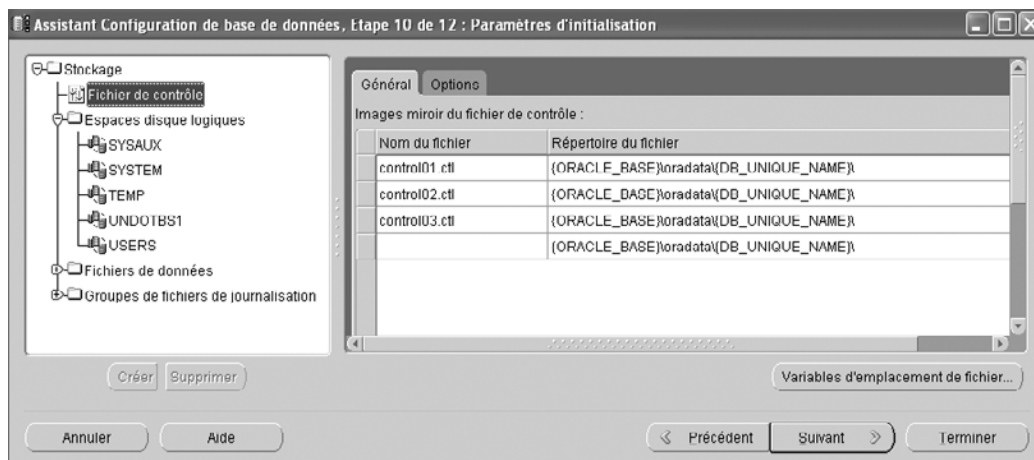


Figure 14-22
Caractéristiques du fichier de contrôle

Pour les fichiers de contrôle, seul leur emplacement est à indiquer. Ils sont créés automatiquement avec la base de données. Le fichier d'initialisation init.ora ou pfileSID.ora de la base doit indiquer leur localisation, car ils permettent de retrouver l'ensemble des autres fichiers, base et redo-log. Pour des raisons de sécurité, trois fichiers de contrôle sont créés automatiquement à l'emplacement indiqué. Nous vous conseillons de les répartir sur des disques (volumes) différents.

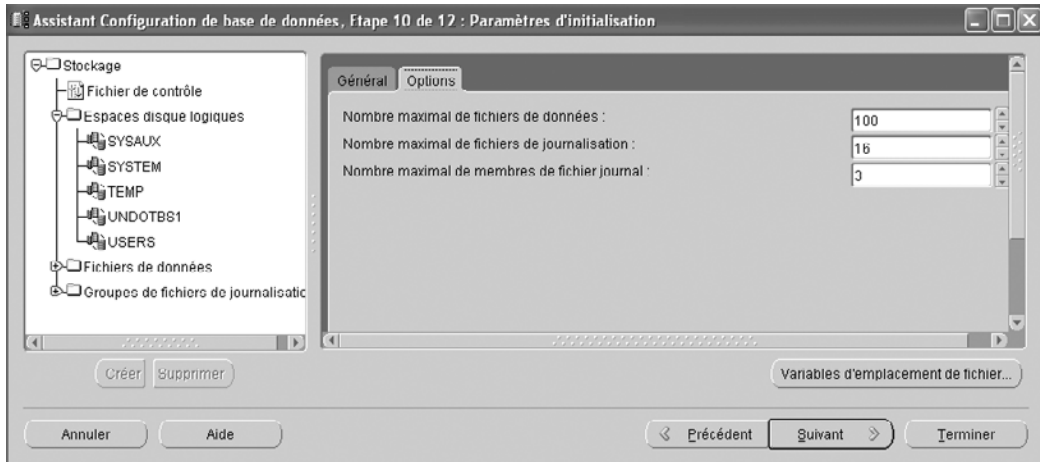


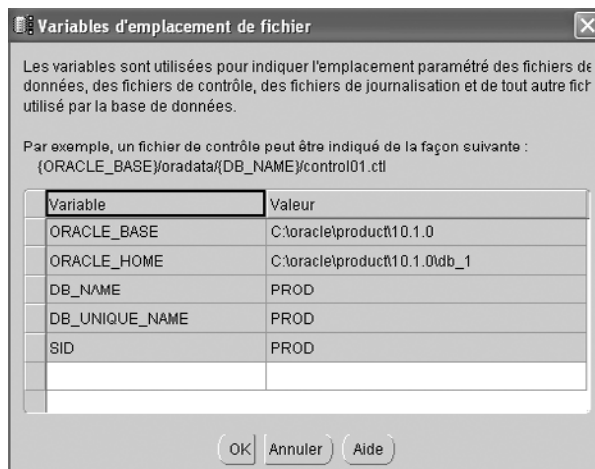
Figure 14-23

Options inscrites dans les fichiers de contrôle

Certaines options figurent dans les fichiers de contrôle, notamment le nombre maximum de fichiers qu'accepte la base de données. Comme un fichier peut maintenant « grossir » (ce n'était pas le cas pour d'anciennes versions d'Oracle), l'augmentation de la taille de la base de données ne nécessite plus forcément l'ajout de nouveaux fichiers.

Figure 14-24

Variables d'emplacement de fichiers



Le bouton *Variables d'emplacement de fichiers* (déjà commenté) est présent sur tous les écrans. Les variables ORACLE_BASE et ORACLE_HOME respectent les normes OFA.

Les tablespaces et leurs fichiers associés

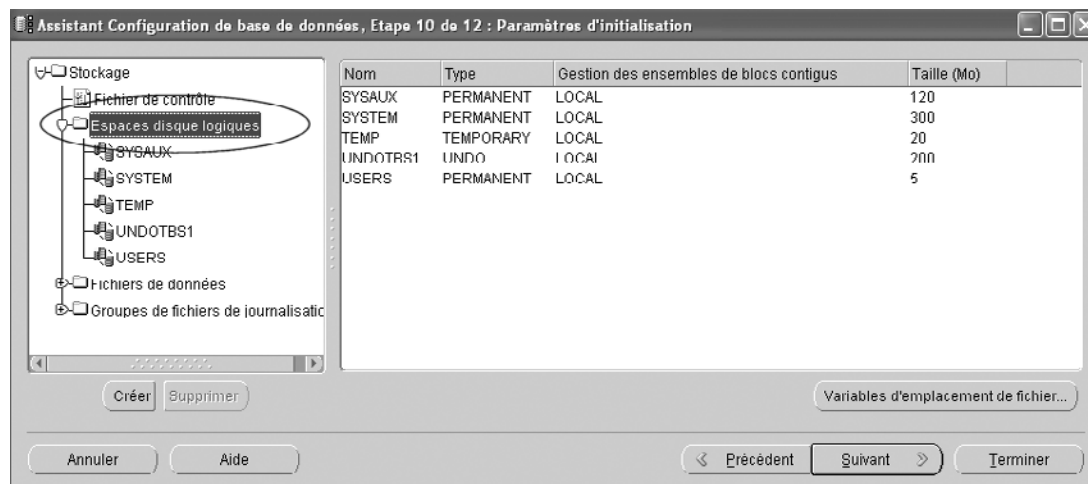


Figure 14-25

Liste des tablespaces

L'exemple proposé possède plusieurs tablespaces, chacun d'eux ayant un rôle spécifique. Un tablespace peut être constitué d'un ou de plusieurs fichiers, créés au fur et à mesure de son développement. L'utilisation des tablespaces est décrite au chapitre 23, *Gestion de l'espace disque et des fichiers*. Cet écran indique la liste des tablespaces. Leurs fichiers associés sont placés dans la rubrique *Fichiers de données* de la fenêtre de gauche. Étudions quelques-uns d'entre eux en détail.

Vous pouvez modifier les noms, sauf ceux de SYSTEM et SYSAUX. Attribuez-leur des noms évoquant leur fonction, c'est plus pratique :

- **SYSTEM** : destiné à contenir le dictionnaire interne nécessaire au fonctionnement de la base, au stockage des procédures, des fonctions, des descriptions des objets, des tables, à la gestion de la sécurité, etc. Il est créé pendant l'ordre de création de la base.
- **SYSAUX** : destiné à contenir la structure de toutes les options proposées par Oracle : Oracle Text, Spatial, Enterprise Manager, etc. Ces options étaient autrefois placées dans le tablespace SYSTEM, qui est maintenant recentré sur sa fonction première.
- **USERS, APPLI_A, APPLI_B, etc.** : tablespaces de type PERMANENT destinés à contenir des données et index de différentes applications. Chacun d'eux est constitué d'un ou de plusieurs fichiers.

- **UNDOTBS1** : c'est un tablespace nécessaire au fonctionnement d'Oracle de type UNDO. Il contient les segments d'annulation, c'est-à-dire les données en instance de validation (commit) ou d'annulation (rollback).
- **TEMP** : c'est un tablespace nécessaire au fonctionnement d'Oracle de type TEMPORARY. Il contient les opérations temporaires de tri qui n'ont pu être effectuées dans l'espace mémoire réservé à cette fin.

Que se passe-t-il lorsque les fichiers composant un tablespace sont saturés ? Deux solutions sont envisageables : soit vous avez accordé au tablespace le droit de s'allouer automatiquement l'incrément de taille nécessaire (paramètre **AUTOEXTEND**), soit vous ajoutez manuellement un fichier. Ce point, ainsi que d'autres, sont commentés dans les figures suivantes.

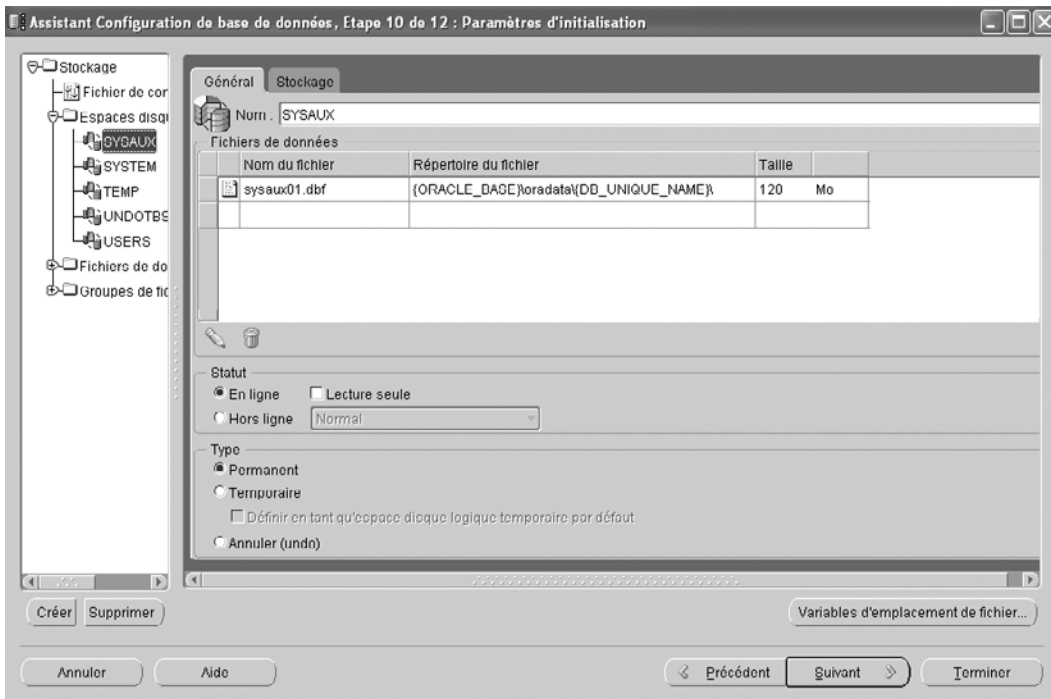


Figure 14-26

Caractéristiques du tablespace SYSAUX

Les caractéristiques générales du tablespace SYSAUX montrent qu'il est associé au fichier `sysaux01.dbf` placé en `{ORACLE_BASE}\oradata\{DB_UNIQUE_NAME}`, soit `C:\oracle\product\10.1.0\oradata\PROD` dans cet exemple. Il est de type **PERMANENT**.

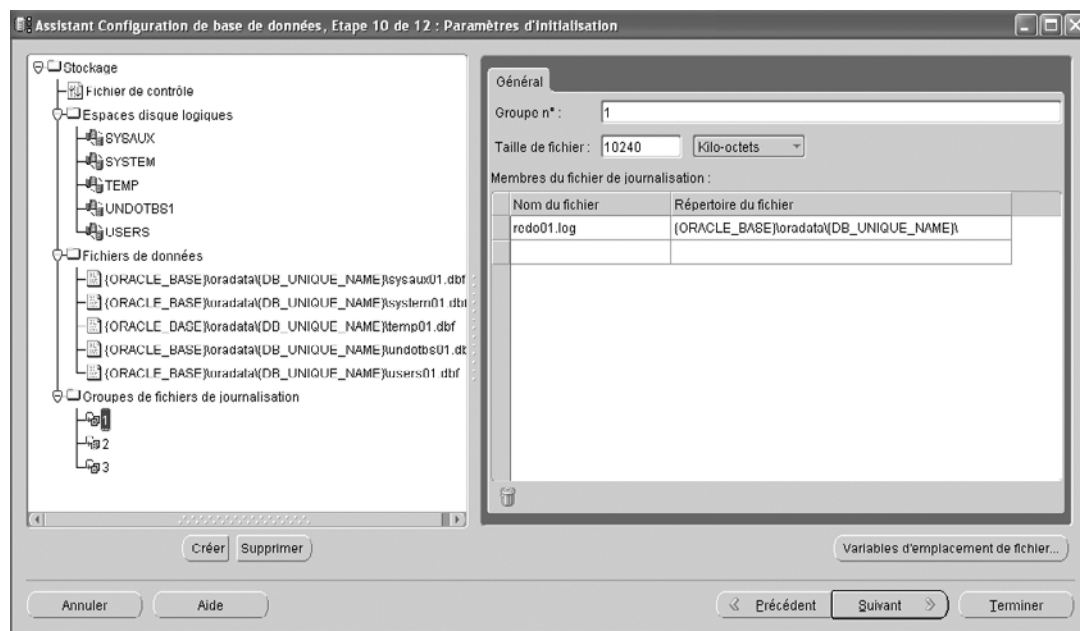


Figure 14-27

Options de stockage du tablespace SYSAUX

Des options de stockage sont précisées dans cet onglet. Tout d'abord, le tablespace est du type **LOCALLY MANAGED** (géré localement). Ce nouveau mode **LOCALLY MANAGED** stocke tous les aspects d'allocation de segments à l'intérieur de chaque tablespace et les informations sont codées au format *bitmap* dans chaque en-tête de tablespace.

L'ancien mode **DICTIONARY** centralise l'allocation d'espace dans le dictionnaire de données de chaque base.

Ces points sont détaillés au chapitre 23, *Gestion de l'espace disque et des fichiers*.

La gestion de l'espace disque est automatique et la journalisation est activée. Cette option peut être pénalisante en termes de place disque. On peut aussi préciser pour ce seul tablespace une taille de bloc de données ou **DB_BLOCK_SIZE**. Ce paramètre a déjà été défini au niveau supérieur, celui de la base de données.

Oracle permet de transporter des tablespaces d'une base vers une autre. Cette possibilité est très pratique par exemple pour alimenter une base d'infocentre depuis une base de données de production. Plutôt qu'extraire les données, les transférer puis les intégrer, on copie un tablespace entier d'une base source et on le raccroche à la base cible. Comme les bases d'origine et cible peuvent être créées avec des **DB_BLOCK_SIZE** différents, Oracle 10g permet maintenant d'avoir des tablespaces formatés avec une taille de bloc différente de celle de la base à laquelle on les accroche. C'est l'objectif de cette option. Nous n'utiliserons pas cette possibilité avancée, qui dépasse l'objectif de ce livre.

Le principe de paramétrage des autres tablespaces est identique.

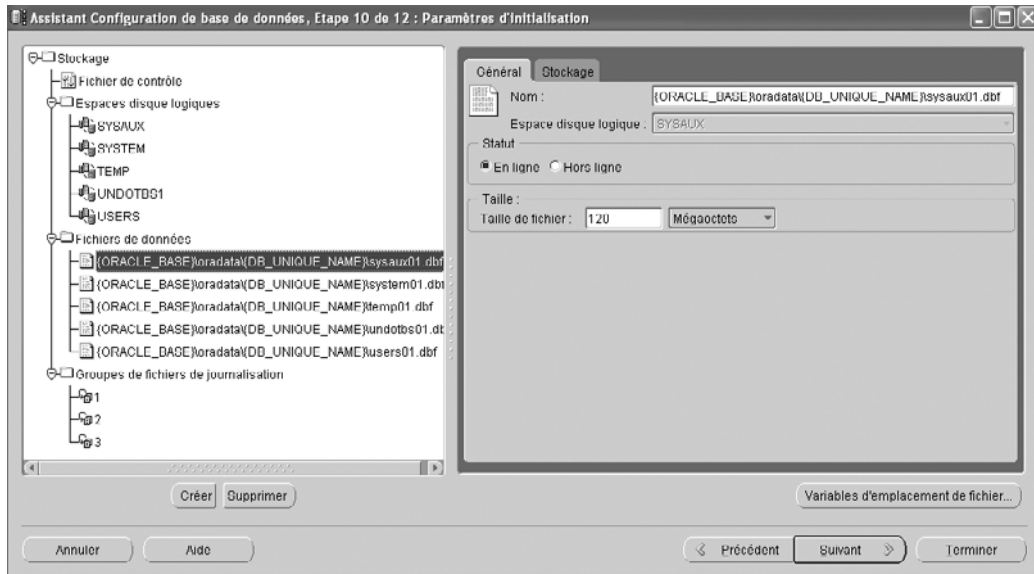


Figure 14-28

Caractéristiques d'un fichier

Cet autre fichier associé à un tablespace possède des caractéristiques générales « classiques ».

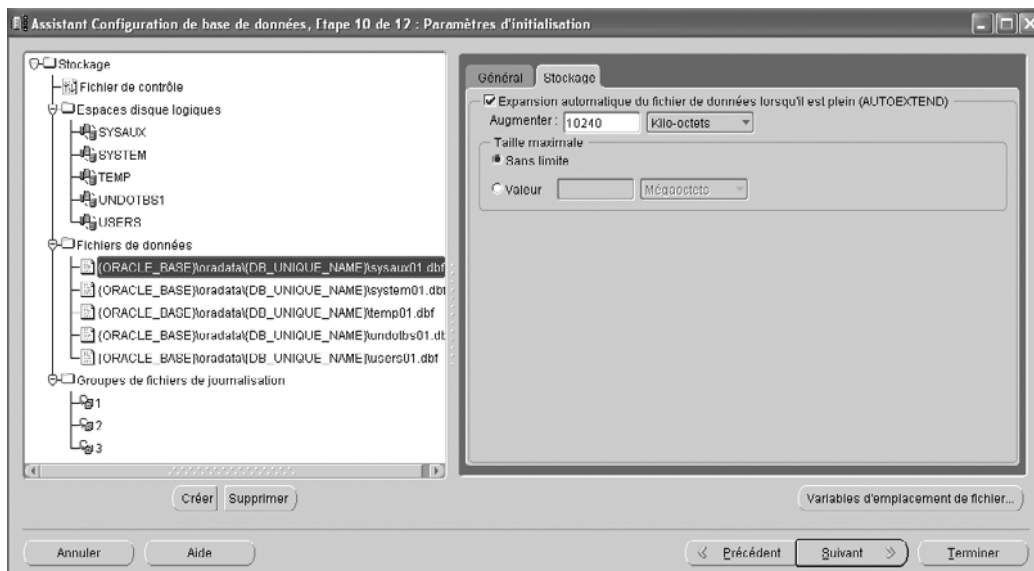


Figure 14-29

Caractéristiques d'un fichier

Les caractéristiques de stockage de ce fichier doivent être commentées :

- lorsqu'il est plein, le fichier augmente automatiquement de taille par incrément de 1024 Ko ;
- sa taille maximale a été précisée, de façon à ce qu'un fichier ne soit pas plus gros que la taille maximale d'un fichier manipulable par mon logiciel de sauvegarde.

Ces options sont très pratiques car elles permettent une gestion automatique de la taille des fichiers.

Remarquez en passant que ces options ne concernent pas le tablespace mais ses fichiers. Il est possible d'interdire à un fichier d'augmenter de taille, s'il est placé sur un disque « plein » et d'autoriser un autre fichier du même tablespace à augmenter de taille s'il est placé sur un disque disposant d'espace libre.

Permettre l'augmentation des fichiers de données avec l'option « sans limite » est dangereux. Par exemple, il est possible d'arriver à une taille de fichier supérieure à celle permise par vos médias de sauvegarde. Vous aurez alors des fichiers impossibles à sauvegarder ! Je vous recommande fortement de toujours fixer une taille limite aux fichiers.

Le segment d'annulation

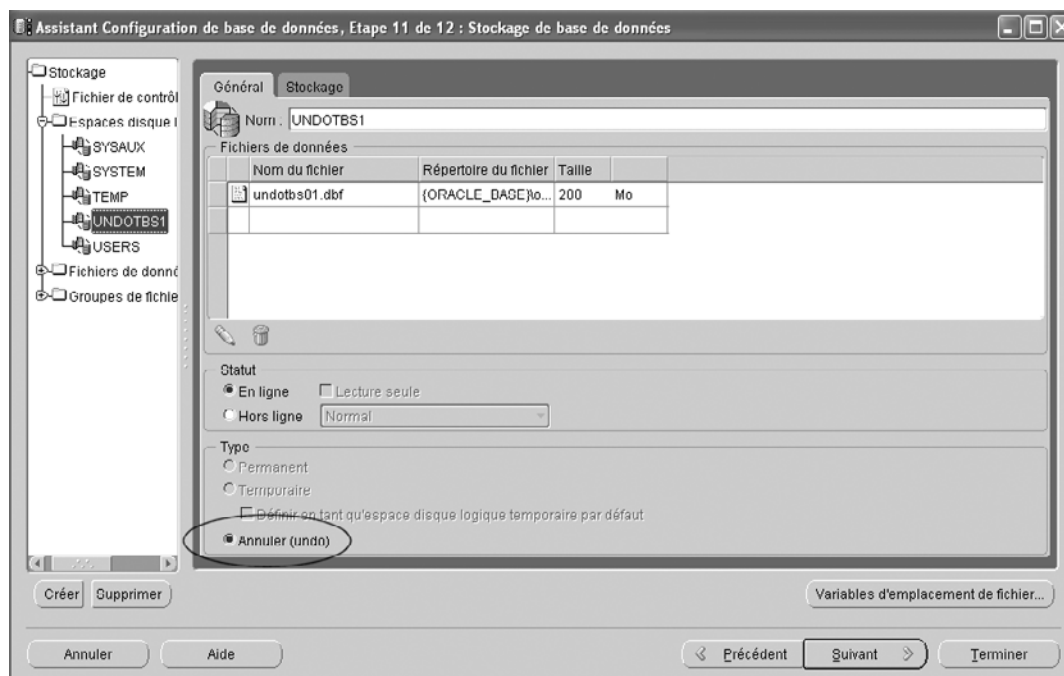


Figure 14-30

Tablespace d'annulation

Le tablespace d'annulation nommé UNDOTBS n'est pas de type permanent mais de type UNDO. Son principe est présenté au chapitre 11, *Transactions et accès concurrents* et son fonctionnement détaillé au chapitre 23, *Gestion de l'espace disque et des fichiers*.

Les fichiers redo-log

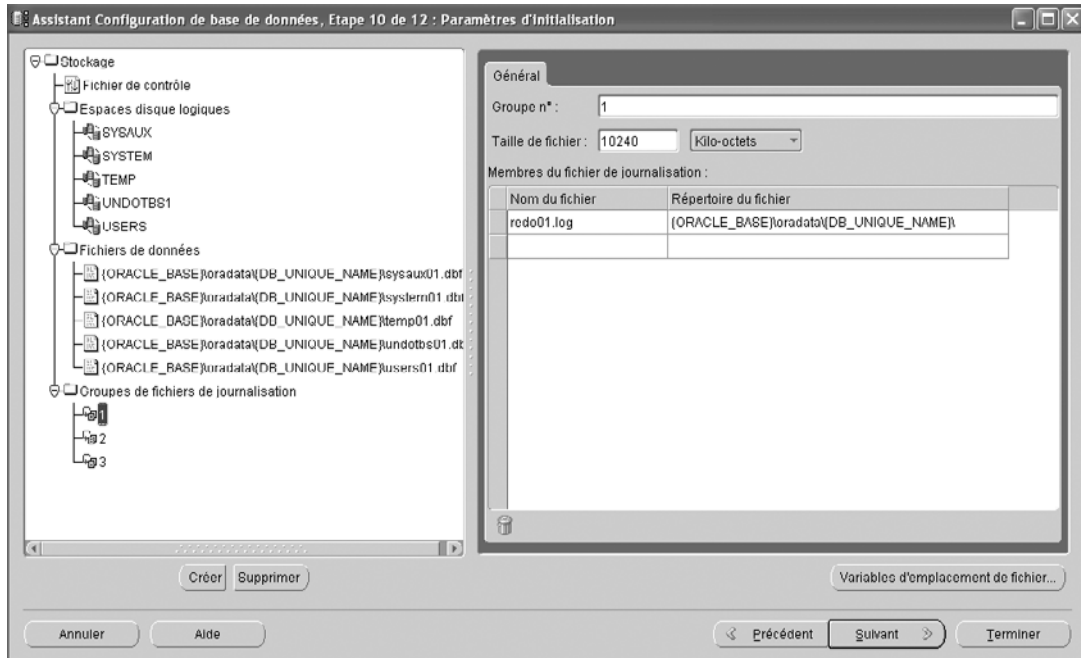


Figure 14-31

Caractéristiques d'un fichier redo-log

La gestion des fichiers journaux est assurée par Oracle. Seul leur emplacement et leur taille doivent être précisés. Pour modifier leurs caractéristiques par la suite, il suffit d'ajouter des fichiers journaux ayant de nouvelles caractéristiques, puis de supprimer les anciens.

Pour des raisons de sécurité, il est possible de multiplexer les fichiers redo-log. Dans ce cas, des fichiers identiques, répartis dans des *groupes*, seront placés sur des disques différents. Une panne disque n'entraînera pas une indisponibilité longue de la base. Leur principe est présenté au chapitre 11, *Transactions et accès concurrents* et leur fonctionnement détaillé au chapitre 23, *Gestion de l'espace disque et des fichiers*.

L'étape finale

Toutes les caractéristiques nécessaires à la création d'une nouvelle base sont maintenant renseignées.

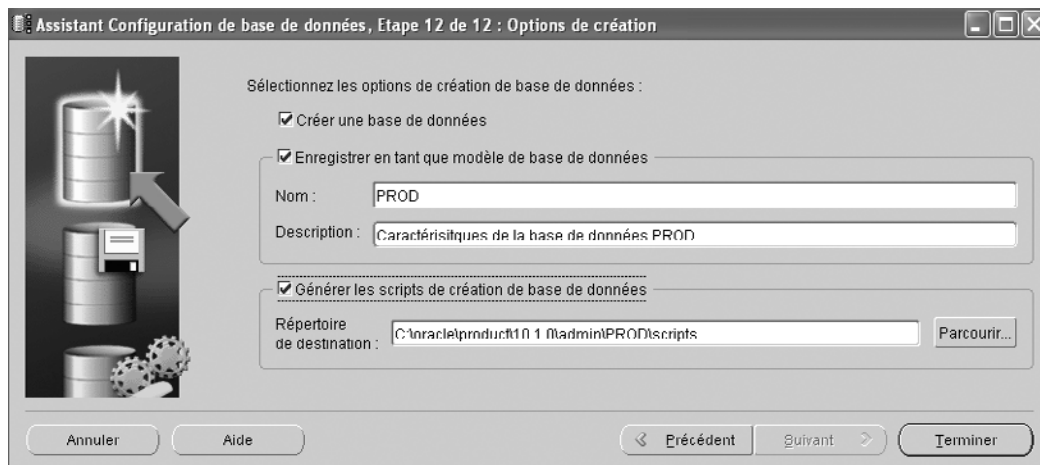


Figure 14-32

Options de création et de documentation

Si vous générez les scripts et créez la base de données :

- tous les répertoires (udump, bdump, cdump...) sont automatiquement créés ;
- le fichier C:\oracle\product\10.1.0\db_1\database\pwdSID est créé ;
- le fichier C:\oracle\product\10.1.0\admin\PROD\pfile\init.ora sert à la création du fichier d'initialisation persistant C:\oracle\product\10.1.0\db_1\database\pfileSID.ora

Si vous générez les scripts sans créer la base de données :

- les répertoires (udump, bdump, cdump...) ne sont pas créés ;
- le fichier orapw<SID> n'est pas créé (il faut le faire manuellement ou commenter la ligne remote_login_passwordfile=EXCLUSIVE dans le fichier init.ora) ;
- le fichier d'initialisation persistant C:\oracle\product\10.1.0\db_1\database\pfileSID.ora sera utilisé par défaut.

Démarche proposée

La création de la base entraîne celle de nombreux répertoires et fichiers, la création de la base et le déroulement des catalogues des options. Cela peut être long. Nous vous proposons de suivre les étapes suivantes :

1. Enregistrez le modèle et générez les scripts. Ne créez pas la base de données à cette étape.
2. Étudiez soigneusement les scripts SQL générés. Ils sont commentés dans ce chapitre.

3. Si tout vous convient, relancez DBCA, ouvrez le modèle sauvegardé et exécutez la première étape y compris la création de la base.

Figure 14-33

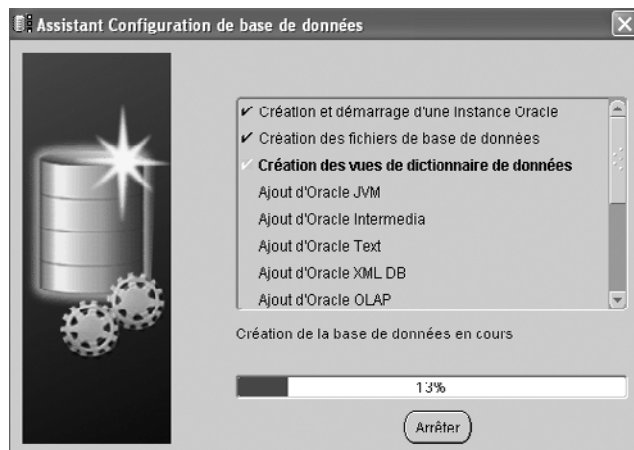
Résumé de l'installation



Ce résumé précède la création de la base. Il peut être enregistré sous forme d'une page HTML. Nous vous conseillons d'utiliser cette option de documentation et de le conserver en C:\oracle\product\10.1.0\admin\PROD\scripts.

Figure 14-34

Création de la base



Il est temps de vous reposer ! C'est maintenant à Oracle d'exécuter toutes les options demandées.

La création de la base de données est assez rapide, mais l'exécution des catalogues peut prendre plusieurs heures, suivant la puissance du serveur.

Les ordres de création d'une base Oracle 10g

- À l'issue de l'étape précédente, un ensemble de fichiers « .bat » et « .SQL » est créé :
- un fichier destiné à enchaîner l'exécution de tous les autres ;
- la création et le démarrage du service OracleServiceSID ;
- un fichier d'ordres SQL contenant la création de la base de données ;
- un fichier d'ordres SQL destiné à compléter cette création en ajoutant les nouveaux tablespaces, à modifier leurs caractéristiques, etc. ;
- des fichiers d'ordres SQL ayant pour rôle d'installer les catalogues nécessaires au fonctionnement d'options du noyau Oracle, comme Oracle JVM, Intermédia, Spatial, les démonstrations, etc. ;
- le fichier d'initialisation propre à votre base de données.

Il est important de bien saisir le lien de cause à effet qui existe entre les options choisies dans l'assistant graphique et les commandes qui en résultent dans les fichiers créés. Pour cela, nous vous présentons maintenant le contenu des ces fichiers.

Le fichier de lancement

Tous les fichiers nécessaires à la création de la base sont regroupés en C:\oracle\product\10.1.0\admin\PROD\scripts.

Pour créer la base PROD, le fichier de commande PROD.bat s'exécute. Il sert à enchaîner l'ensemble des étapes de création de votre base de données PROD. Nous avons modifié son contenu pour y ajouter des commentaires :

```
#####
# Création des répertoires d'administration de la base PROD
#####

mkdir C:\oracle\product\10.1.0\admin\PROD\bdump
mkdir C:\oracle\product\10.1.0\admin\PROD\cdump
mkdir C:\oracle\product\10.1.0\admin\PROD\create
mkdir C:\oracle\product\10.1.0\admin\PROD\pfile
mkdir C:\oracle\product\10.1.0\admin\PROD\udump
mkdir C:\oracle\product\10.1.0\db_1\database
mkdir C:\oracle\product\10.1.0\flash_recovery_area
mkdir C:\oracle\product\10.1.0\oradata
```

```

mkdir C:\oracle\product\10.1.0\oradata\PROD

#####
# La variable d'environnement ORACLE_SID est renseignée
#####

set ORACLE_SID=PROD

#####
# Pour plus de lisibilité, j'ajoute la variable ORA_EXE
# qui pointe vers le répertoire contenant les binaires Oracle
#####

set ORA_EXE=C:\oracle\product\10.1.0\db_1\bin

#####
# Création puis démarrage du service OracleServicePROD
#####

$ORA_EXE\oradim.exe -new -sid PROD -startmode manual -spfile
$ORA_EXE\oradim.exe -edit -sid PROD -startmode auto -srvcstart system

#####
# Lancement du fichier SQL contenant les ordres à exécuter
# Syntaxe : sqlplus /nolog @fichier_sql_a_executer
#####

sqlplus /nolog @C:\oracle\product\10.1.0\admin\PROD\scripts\PROD.sql

```

Avant toute installation, le service OracleServicePROD doit être créé. C'est le rôle dévolu à la commande ORADIM, présentée au chapitre 5, *Fonctionnement d'Oracle 10g sous Windows*.

Le fichier d'ordres SQL

Le contenu du fichier C:\oracle\product\10.1.0\admin\PROD\scripts\PROD.sql, lancé en fin de fichier précédent, regroupe l'appel de tous les fichiers SQL successifs nécessaires à la création de la base. Voici son contenu, des commentaires y ont été ajoutés :

```

#####
# Récupération des mots de passe pour les utilisateurs :
# SYS, SYSTEM, SYSMAN, DBSNMP
#####

set verify off
PROMPT specify a password for sys as parameter 1;
DEFINE sysPassword = &1
PROMPT specify a password for system as parameter 2;
DEFINE systemPassword = &2

```

```

PROMPT specify a password for sysman as parameter 3;
DEFINE sysmanPassword = &3
PROMPT specify a password for dbsnmp as parameter 4;
DEFINE dbsnmpPassword = &4

#####
# Création du fichier mot de passe et affectation d'un mot de passe
# qui sera utilisé pour le reste de l'installation.
#####

host C:\oracle\product\10.1.0\db_1\bin\orapwd.exe
      file=C:\oracle\product\10.1.0\db_1\database\PWDPROD.ora
      password=&&sysPassword force=y

#####
# Création de la base et des fichiers
#####

@C:\oracle\product\10.1.0\admin\PROD\scripts\CreateDB.sql
@C:\oracle\product\10.1.0\admin\PROD\scripts\CreateDBFiles.sql

#####
# Liste commentée des scripts lancés durant
# la création de la base PROD.
# !!! N'EXÉCUTEZ QUE LES SCRIPTS NÉCESSAIRES !!!
#####
# CreateDB.sql          création de la base PROD
# CreateDBFiles.sql     création des tablespaces
# CreateDBCatalog.sql   création du catalogue de la base
# JServer.sql           scripts SQL de l'option Oracle JVM
# ordinst.sql           scripts SQL de l'option ordinst
# interMedia.sql        scripts SQL de l'option interMedia
# odmmetadata.sql       scripts SQL de l'option Directory Manager
# context.sql           scripts SQL de l'option context
# xdb_protocol.sql      scripts SQL de l'option protocol XDB
# cwm1ite.sql           scripts SQL de l'option cwm1ite
# spatial.sql           scripts SQL de l'option Spatial
# odm.sql               scripts SQL de l'option odm
# ultraSearch.sql       scripts SQL de l'option iSearch
# emRepository.sql      scripts SQL de l'option Enterprise Manager
# cwm1ite.sql           scripts SQL de l'option cwm1ite
# ultraSearchCfg.sql    scripts SQL de configuration d'Ultra Search
# postDBCcreation.sql   scripts SQL de fin d'installation
# customScripts.sql     scripts SQL de customisation
#####

@C:\oracle\product\10.1.0\admin\PROD\scripts\CreateDBCatalog.sql
@C:\oracle\product\10.1.0\admin\PROD\scripts\JServer.sql
@C:\oracle\product\10.1.0\admin\PROD\scripts\ordinst.sql
@C:\oracle\product\10.1.0\admin\PROD\scripts\interMedia.sql
@C:\oracle\product\10.1.0\admin\PROD\scripts\odmmetadata.sql
@C:\oracle\product\10.1.0\admin\PROD\scripts\context.sql

```



```
@C:\oracle\product\10.1.0\admin\PROD\scripts\xdb_protocol.sql
@C:\oracle\product\10.1.0\admin\PROD\scripts\cwmlite.sql
@C:\oracle\product\10.1.0\admin\PROD\scripts\spatial.sql
@C:\oracle\product\10.1.0\admin\PROD\scripts\odm.sql
@C:\oracle\product\10.1.0\admin\PROD\scripts\ultraSearch.sql
@C:\oracle\product\10.1.0\admin\PROD\scripts\emRepository.sql
@C:\oracle\product\10.1.0\admin\PROD\scripts\ultraSearchCfg.sql
@C:\oracle\product\10.1.0\admin\PROD\scripts\postDBCreation.sql
@C:\oracle\product\10.1.0\admin\PROD\scripts\customScripts.sql
```

La première partie du script affecte les mots de passe définis dans DBCA à des variables qui seront réutilisées dans les scripts.

La seconde étape lance l'exécutable ORAPWD qui crée un fichier mot de passe. L'objectif et l'utilisation de ce fichier mot de passe sont présentés au chapitre 24, *Stratégie de sécurité sous Windows*.

La troisième étape crée la base. Cette étape est commentée au paragraphe suivant.

Enfin, le script met en place de nombreux produits (toutes les options possibles de la version Oracle Enterprise dans notre cas). N'installez que les options dont vous avez réellement besoin, sinon la durée de création du catalogue et son volume augmentent exagérément et vous consommerez des ressources pour des options que vous n'utiliserez pas.

Le fichier de création de la base

Voici le contenu du fichier CreateDB.sql. Il est lancé par le script précédent et exécute l'ordre CREATE DATABASE. Les étapes de création d'une base sont commentées au chapitre 10, *Démarrer et arrêter une base Oracle 10g*.

Certains paramètres sont présents dans ce fichier et d'autres, comme le DB_BLOCK_SIZE ou l'emplacement des fichiers de contrôle, se trouvent dans le fichier d'initialisation, C:\oracle\product\10.1.0\admin\PROD\scripts\init.ora.

Certains paramètres, comme le DB_NAME ou le DB_DOMAIN, figurent dans le fichier d'initialisation. Ils sont lus lors de la création de la base et sont difficilement modifiables ultérieurement.

Pour chaque étape de création, un fichier trace .log est créé. Placez-le au même endroit que les scripts de création pour, au fil du temps, conserver dans un seul répertoire les ordres de création et leur résultat.

```
connect SYS/ &&sysPassword as SYSDBA

set echo on
spool C:\oracle\product\10.1.0\db_1\assistants\dbca\logs\CreateDB.log

startup nomount
  pfile="C:\oracle\product\10.1.0\admin\PROD\scripts\init.ora";

CREATE DATABASE "PROD"
```

```
MAXINSTANCES 8
MAXLOGHISTORY 1
MAXLOGFILES 16
MAXLOGMEMBERS 3
MAXDATAFILES 100
DATAFILE 'C:\oracle\product\10.1.0\oradata\PROD\system01.dbf'
    SIZE 300M REUSE AUTOEXTEND ON NEXT 10240K MAXSIZE UNLIMITED
    EXTENT MANAGEMENT LOCAL
SYSAUX DATAFILE 'C:\oracle\product\10.1.0\oradata\PROD\sysaux01.dbf'
    SIZE 120M REUSE AUTOEXTEND ON NEXT 10240K MAXSIZE UNLIMITED
DEFAULT TEMPORARY TABLESPACE TEMP TEMPFILE
'C:\oracle\product\10.1.0\oradata\PROD\temp01.dbf' SIZE 20M REUSE
    AUTOEXTEND ON NEXT 640K MAXSIZE UNLIMITED
UNDO TABLESPACE "UNDOTBS1" DATAFILE
    'C:\oracle\product\10.1.0\oradata\PROD\undotbs01.dbf'
    SIZE 200M REUSE AUTOEXTEND ON NEXT 5120K MAXSIZE UNLIMITED
CHARACTER SET WE8MSWIN1252
NATIONAL CHARACTER SET AL16UTF16
LOGFILE GROUP 1
    ('C:\oracle\product\10.1.0\oradata\PROD\redo01.log') SIZE 10240K,
    GROUP 2
    ('C:\oracle\product\10.1.0\oradata\PROD\redo02.log') SIZE 10240K,
    GROUP 3
    ('C:\oracle\product\10.1.0\oradata\PROD\redo03.log') SIZE 10240K
# remarque : && indique un passage de paramètre
USER SYS IDENTIFIED BY "&&sysPassword"
USER SYSTEM IDENTIFIED BY "&&systemPassword"
;

spool off
```

Le fichier d'initialisation permanent n'étant pas encore créé, un fichier `init.ora` est appelé. Son contenu est détaillé plus loin.

L'exemple précédent utilise l'option `REUSE` pour écraser les fichiers s'ils existent déjà. Pratique en phase de mise au point, cette option est pourtant très dangereuse : imaginez les dégâts causés si l'ordre de création existant est relancé quelques mois plus tard ! Préférez toujours une suppression manuelle des fichiers, quitte à ce que l'ordre de création rencontre une erreur si le fichier existe déjà.

Remarquez que les tablespaces `SYSTEM` et `UNDOTBS1` ont des `DATAFILES` alors que le tablespace temporaire `TEMP` possède un `TEMPFILE`.

Les tablespaces `SYSTEM`, `SYSAUX` et `UNDOTBS1` sont créés par l'ordre `CREATE DATABASE`. Le tablespace `SYSTEM` contient le dictionnaire de données, le tablespace `SYSAUX` la structure des données nécessaires aux produits complémentaires Oracle. Le tablespace `UNDOTBS1` est de type `UNDO`. Il est destiné à remplacer l'ancienne gestion des Rollback Segments, qui est maintenant automatique avec Oracle 10g. Le choix entre

l'ancienne gestion de Rollback Segments et la nouvelle gestion automatique par le tablespace de type UNDO est indiqué à la base dans le fichier init.ora.

Depuis Oracle 10g, l'ordre de création d'une base CREATE DATABASE demande obligatoirement la modification des mots de passe des utilisateurs SYS et SYSTEM. L'assistant DBCA (*Database Configuration Assistant*) utilise cette possibilité. Oracle 10g sera la dernière version d'Oracle qui impose la création de l'utilisateur SYSTEM durant la création de la base.

Les fichiers complémentaires de création de la base

Le fichier CreateDBFiles.sql complète la création initiale de la base de données. Il crée un ou plusieurs tablespaces, dont les caractéristiques sont données pour information.

```
connect SYS/&&sysPassword as SYSDBA
set echo on
spool C:\oracle\product\10.1.0\db_1\assistants\dbca\logs\CreateDBFiles.log

CREATE TABLESPACE "USERS" LOGGING DATAFILE
    'C:\oracle\product\10.1.0\oradata\PROD\users01.dbf' SIZE 5M REUSE
    AUTOEXTEND ON NEXT 1280K MAXSIZE UNLIMITED EXTENT MANAGEMENT
    LOCAL SEGMENT SPACE MANAGEMENT AUTO ;

ALTER DATABASE DEFAULT TABLESPACE "USERS";

spool off
```

Premier d'une longue liste d'options, le fichier CreateDBCatalog.sql installe le catalogue principal du dictionnaire catalog.sql, puis un ensemble de catalogues complémentaires standard. Le catalogue Oracle est abordé au chapitre 12, *Le dictionnaire de données Oracle 10g*.

```
connect SYS/&&sysPassword as SYSDBA
set echo on
spool C:\oracle\product\10.1.0\admin\PROD\scripts\CreateDBCatalog.log

@C:\oracle\product\10.1.0\db_1\rdbms\admin\catalog.sql;
@C:\oracle\product\10.1.0\db_1\rdbms\admin\catblock.sql;
@C:\oracle\product\10.1.0\db_1\rdbms\admin\catproc.sql;
@C:\oracle\product\10.1.0\db_1\rdbms\admin\catocbk.sql;
@C:\oracle\product\10.1.0\db_1\rdbms\admin\owminst.plb;

connect SYSTEM/&&systemPassword
@C:\oracle\product\10.1.0\db_1\sqlplus\admin\pupbld.sql;

connect SYSTEM/&&systemPassword

set echo on
spool C:\oracle\product\10.1.0\admin\PROD\scripts\sqlPlusHelp.log

@C:\oracle\product\10.1.0\db_1\sqlplus\admin\help\hlpbld.sql helpus.sql;

spool off
spool off
```

Les scripts SQL complémentaires

Tous les fichiers servant à installer une option complémentaire ont la même structure : un fichier *log* enregistre l'opération et les scripts SQL, pilotant l'installation du catalogue propre à l'option sont lancés. Par exemple, voici le contenu du script Jserver.sql, servant à installer l'option *Oracle JVM*.

```
connect SYS/ &sysPassword as SYSDBA
set echo on
spool C:\oracle\product\10.1.0\db_1\assistants\dbca\logs\JServer.log
@C:\oracle\product\10.1.0\db_1\javavm\install\initjvm.sql;
@C:\oracle\product\10.1.0\db_1\jdk\admin\initxml.sql;
@C:\oracle\product\10.1.0\db_1\jdk\admin\xmlja.sql;
@C:\oracle\product\10.1.0\db_1\rdbms\admin\catjava.sql;
@C:\oracle\product\10.1.0\db_1\rdbms\admin\catexf.sql;
spool off
```

L'étape de postcréation

Les deux derniers fichiers SQL lancés : postDBCcreation.sql et customScripts.sql ne sont pas des options. Ils finalisent l'installation.

Voici le contenu de postDBCcreation.sql :

```
connect SYS/ &sysPassword as SYSDBA

set echo on
spool C:\oracle\product\10.1.0\admin\PROD\create\postDBCcreation.log

connect SYS/ &sysPassword as SYSDBA
set echo on

create spfile='C:\oracle\product\10.1.0\db_1\database\spfilePROD.ora' FROM pfile=
➔ 'C:\oracle\product\10.1.0\admin\PROD\scripts\init.ora';

shutdown immediate;

connect SYS/ &sysPassword as SYSDBA

startup ;

select 'utl_recomp_begin: ' || to_char(sysdate, 'HH:MI:SS') from dual;
execute utl_recomp.recomp_serial();
select 'utl_recomp_end: ' || to_char(sysdate, 'HH:MI:SS') from dual;

spool C:\oracle\product\10.1.0\admin\PROD\create \postDBCcreation.log
```

Le fichier d'initialisation persistant doit obligatoirement être installé en C:\oracle\product\10.1.0\db_1\database et son nom doit respecter la syntaxe spfileSID.ora ou SID correspondant au nom de l'instance (PROD dans notre cas).

Si vous utilisez un fichier d'initialisation persistant, ne vous fiez plus au contenu d'un fichier d'initialisation « classique » qui peut être différent des valeurs inscrites dans le fichier permanent. Même s'il est possible de lire le fichier persistant, Oracle recommande de ne jamais l'ouvrir et de ne surtout pas le modifier. Ces points sont détaillés au chapitre 23, *Gestion de l'espace disque et des fichiers*.

Dans cet exemple, le fichier customScripts.sql ne contient pas d'action car nous n'avons pas indiqué de script supplémentaire à l'étape n°9 de l'assistant DBCA.

Le fichier d'initialisation de la base

Observons le contenu du fichier d'initialisation de l'instance initPROD.ora qui a servi à la création du fichier persistant. Oracle recommande de plus en plus d'utiliser le fichier persistant présenté au chapitre 23, *Gestion de l'espace disque et des fichiers*.

Le contenu du fichier d'initialisation reprend les paramètres saisis dans DBCA. Nous aborderons ses principaux paramètres au chapitre 28, *Optimisation et performances* :

```
#####
# Copyright (c) 1991, 2001, 2002 by Oracle Corporation
#####

#####
# Cache and I/O
#####
db_block_size=8192
db_cache_size=25165824
db_file_multiblock_read_count=16

#####
# File Configuration
#####
control_files=( "C:\oracle\product\10.1.0\oradata\PROD\control01.ct1",
➤ "C:\oracle\product\10.1.0\oradata\PROD\control02.ct1",
➤ "C:\oracle\product\10.1.0\oradata\PROD\control03.ct1" )
db_recovery_file_dest=C:\oracle\product\10.1.0\flash_recovery_area
db_recovery_file_dest_size=2147483648

#####
# Cursors and Library Cache
#####
open_cursors=300

#####
# Diagnostics and Statistics
#####
```

```
background_dump_dest=C:\oracle\product\10.1.0\admin\PROD\bdump
core_dump_dest=C:\oracle\product\10.1.0\admin\PROD\cdump
user_dump_dest=C:\oracle\product\10.1.0\admin\PROD\udump
```

```
#####
# Miscellaneous
#####
compatible=10.1.0.2.0
```

```
#####
# Job Queues
#####
job_queue_processes=10
```

```
#####
# Database Identification
#####
db_domain=gilles.fr
db_name=PROD
```

```
#####
# NLS
#####
nls_language="FRENCH"
nls_territory="FRANCE"
```

```
#####
# Pools
#####
java_pool_size=50331648
large_pool_size=8388608
shared_pool_size=83886080
```

```
#####
# Processes and Sessions
#####
processes=150
```

```
#####
# System Managed Undo and Rollback Segments
#####
undo_management=AUTO
undo_tablespace=UNDOTBS1
```

```
#####
# Shared Server
#####
dispatchers="(PROTOCOL=TCP) (SERVICE=PRODXDB)"
```

```
#####
# Security and Auditing
```

```
#####  
remote_login_passwordfile=EXCLUSIVE  
  
#####  
# Sort, Hash Joins, Bitmap Indexes  
#####  
pga_aggregate_target=25165824  
sort_area_size=65536
```

Pour les habitués d'Oracle, vous remarquerez de nouveaux paramètres apparus avec Oracle 10g, dont la gestion automatique des segments d'annulation. Ce point est abordé au chapitre 11, *Transactions et accès concurrents*.

Création de la base

Une fois les fichiers examinés, relancez DBCA, ouvrez le modèle, effectuez les modifications identifiées et sauvegardez le modèle ainsi que les fichiers de création. Si tout vous convient, lancez la création de la base. La durée d'exécution de ces commandes (surtout la création des catalogues des différentes options) peut être assez longue en fonction de la puissance de votre machine.

L'installation de la base par défaut, qui se fait en même temps que les logiciels Oracle, dure moins longtemps, car tous les fichiers la composant sont copiés à partir du CD-Rom d'installation d'Oracle 10g. Ils intègrent déjà les catalogues standard et certaines options préinstallées.

Si vous consultez les fichiers journaux générés pendant l'exécution, ne tenez pas compte des messages d'erreur lors du déroulement des catalogues : avant la création d'un objet, il est systématiquement détruit (au cas où ces scripts auraient déjà été lancés). La première fois, il y a donc une erreur avant chaque création.

En cas de problème

Les éventuels problèmes d'autorisations que vous pouvez rencontrer lors de la création de votre base de données, proviennent souvent de droits d'accès incorrects sur les répertoires.

Pour relancer une installation, il ne reste plus qu'à détruire les fichiers partiellement créés sous `C:\oracle\product\10.1.0\oradata\PROD` et sous `C:\oracle\product\10.1.0\admin\PROD` : redo-log, base de données et fichiers de contrôle. Il faut ensuite reprendre la création de la base de données en ouvrant le modèle enregistré. Pour devenir un administrateur Oracle confirmé, entraînez-vous !

Les étapes postérieures à la création de la base

Une fois la nouvelle base de données créée, il convient de l'intégrer dans l'environnement de travail.

La gestion des droits d'accès aux répertoires NTFS

Si vous avez utilisé DBCA (*Oracle DataBase Configuration Assistant*), l'assistant gère automatiquement pour vous les restrictions d'accès sur les répertoires NTFS de Windows (pour plus d'informations sur NTFS, référez-vous à la documentation Windows).

Les droits attribués automatiquement par DBCA sont résumés dans le tableau suivant :

Répertoire	Droits NTFS attribués par DBCA
ORACLE_BASE\ORACLE_HOME	ADMINISTRATEUR : contrôle total SYSTEM : contrôle total UTILISATEUR AUTHENTIFIÉ : lecture, exécution, visualisation du contenu
ORACLE_BASE\admin\nom_de_la_base	ADMINISTRATEUR : contrôle total SYSTEM : contrôle total
ORACLE_BASE\oradata\nom_de_la_base	ADMINISTRATEUR : contrôle total SYSTEM : contrôle total
ORACLE_BASE\ORACLE_HOME\database\spfileSID.ora	ADMINISTRATEUR : contrôle total SYSTEM : contrôle total

La gestion des droits d'accès à la base de registre

Oracle préconise aussi de limiter l'accès aux informations concernant Oracle, situées dans la base de registre en HKEY_LOCAL_MACHINE\SOFTWARE\ORACLE.

Pour réaliser cette opération :

1. accédez au contenu de la base de registre à l'aide de la commande regedit ;
2. placez-vous en HKEY_LOCAL_MACHINE\SOFTWARE\ORACLE ;
3. sélectionnez le menu Options puis Autorisations. Une boîte de dialogue apparaît ;
4. supprimez les autorisations d'accès pour tous les utilisateurs qui ne sont pas administrateurs Oracle ou administrateurs Windows. Conservez tous les droits à l'utilisateur SYSTEM puisque ce compte est celui sous lequel les services et la base Oracle sont lancés ;
5. assurez-vous que les utilisateurs qui utilisent la base Oracle possèdent tous le privilège de lecture ;
6. validez vos modifications ;
7. quittez la base de registre.

Cette manipulation apporte une grande sécurité. Comme toute mesure sécuritaire, elle augmente les contraintes. Par exemple, le point numéro 5 (limiter l'accès en lecture à une liste restreinte d'utilisateurs) sera contraignante dès qu'un utilisateur sera créé ou supprimé. Deux solutions sont possibles dans ce cas : ouvrez largement les droits en lecture à tous ou restreignez-les. Dans ce cas, des procédures sont à écrire et à suivre scrupuleusement pour associer à coup sûr la création d'un utilisateur Windows et la gestion des droits d'accès à la base de registre !

Configurer les options d'Oracle

Certaines options « évoluées » d'Oracle nécessitent une configuration supplémentaire. Généralement, si vous utilisez DBCA (*Oracle DataBase Configuration Assistant*), ces options sont configurées automatiquement. Si vous créez la base manuellement à l'aide de scripts, vous devrez configurer ces options.

Les options concernées sont :

- Oracle interMedia et l'option Oracle interMedia Locator ;
- Oracle Text ;
- Oracle Spatial ;
- Oracle Advanced Replication.

La configuration de ces options dépasse les objectifs de ce livre, référez-vous à leurs documentations respectives.

Configurer Oracle Net

Si vous souhaitez accéder à la base de données en mode client-serveur, il est nécessaire de configurer Oracle Net côté client pour ajouter un point d'entrée au fichier tnsnames.ora. Cette étape est décrite au chapitre 15, *Oracle Net, le middleware Oracle*.

Accéder aux caractéristiques de la nouvelle base

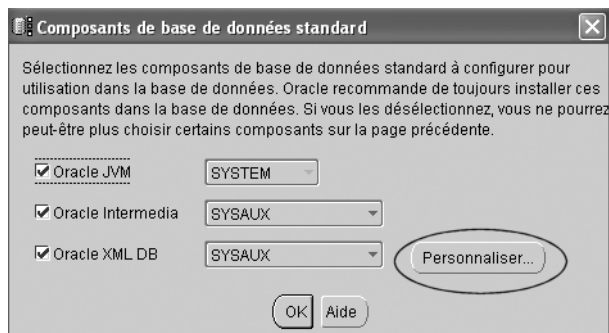
L'assistant Administration Assistant for Windows permet d'accéder aux caractéristiques de la base liées au système d'exploitation Windows. Cet outil est accessible depuis le menu Windows Programmes > Démarrer > Oracle – *Ora10gDb-Home*<X> > Configuration and Migration Tools > Administration Assistant for Windows.

Par exemple, deux bases de données sont maintenant installées, TEST et PROD. À quelle base de données nous connecterons-nous par défaut sur le serveur lorsque nous lancerons un outil Oracle, comme SQL*Plus ? Quelles sont les options de démarrage et d'arrêt de la base PROD ? Cet assistant permet d'accéder et de modifier ces paramètres très facilement.

Il suffit, à partir de l'écran précédent, d'effectuer un « clic droit » au niveau de l'Oracle Home dont dépendent les deux bases pour accéder aux caractéristiques et les modifier si besoin. Ce point est détaillé au chapitre 5, *Fonctionnement d'Oracle 10g sous Windows*.

Figure 14-14

*Liens entre Oracle
et Windows*



Démarrer automatiquement la base

Au chapitre précédent, *L'installation d'Oracle 10g sous Windows*, et au chapitre 5, *Fonctionnement d'Oracle sous Windows*, nous avons étudié comment automatiser le lancement des bases Oracle 10g avec le démarrage de votre ordinateur.

Intégrer la nouvelle base dans les sauvegardes

Et si votre disque dur tombait en panne demain, comment repartiriez-vous ? La sauvegarde de l'environnement de travail représente une tâche essentielle de l'administrateur de données, plus encore que de connaître toutes les subtilités d'optimisation de la base. Ce sujet important est abordé au chapitre 26, *La sauvegarde d'une base Oracle 10g*.

La gestion rigoureuse des sauvegardes est l'un des éléments qui différencie les amateurs des professionnels !

Tester l'accès à la base

Je vous recommande de tester l'accès à la base avec les logiciels usuels (SQL*Plus, iSQL*Plus, Enterprise Manager...) à la fois localement et en accès distant, client-serveur ou Web. Ces points sont détaillés à la fin du chapitre précédent.

Travailler sur la base

Votre nouvelle base de données est accessible, elle démarre automatiquement et peut être sauvegardée. Vous pouvez maintenant effectuer les tâches normales d'administration : création des utilisateurs, du schéma de la base, insertion des données, etc.

Résumé de l'installation

La création d'une nouvelle base de données est une tâche importante réservée aux administrateurs de données. Oracle Database Configuration Assistant offre une interface graphique conviviale pour vous guider. Elle a été simplifiée par rapport aux versions précédentes.

Mais, si l'interface graphique constitue un plus, la création d'une base reste néanmoins complexe. C'est pourquoi nous avons détaillé et commenté toutes les étapes de cette création et les actions à effectuer *a posteriori*.

L'avantage majeur de l'assistant est d'automatiser l'ensemble de ces opérations dans des scripts, puis de les exécuter.

15

Oracle Net, le middleware Oracle

Dans ce chapitre :

- la définition du middleware ;
- l'architecture client-serveur ;
- la configuration du Listener et des alias ;
- le test d'une installation complète.

Ce chapitre a pour but de démystifier Oracle Net, de vous en expliquer l'installation, le fonctionnement ainsi que le paramétrage. Oracle a mis ses clients à l'épreuve avec d'anciennes versions d'Oracle Net (SQL*Net puis Net8), dont l'installation était une véritable aventure. Heureusement, la situation est maintenant stabilisée grâce à des assistants graphiques conviviaux.

Instant Client, la nouveauté d'Oracle 10g

Dans les précédentes versions d'Oracle, le logiciel « client » était fourni avec la base de données. Vous utilisiez alors un seul jeu de CD-Rom pour installer la base de données (sur un serveur) et le logiciel client. Depuis Oracle 10g, pour installer uniquement le logiciel client sur un poste de travail ou un serveur Web devant se connecter à Oracle, vous devez utiliser un CD-Rom « allégé » : l'Oracle Database Client Software. Vous le trouverez pour le système d'exploitation de votre choix sur le site <http://otn.oracle.com>.

Le logiciel se nomme Instant Client. Il permet d'installer un client Oracle sur tous types de systèmes d'exploitation pour accéder à une base Oracle distante. Les applications s'appuyant sur les couches « basses » de communication Oracle, c'est-à-dire les OCI et les OCCI, ou sur des couches d'un niveau d'abstraction plus élevé, telles qu'ODBC, JDBC, fonctionnent nativement.

Le choix du périmètre de l'installation ne s'effectue plus pendant l'installation, mais par la sélection en amont d'un package logiciel correspondant à vos besoins. Cela offre à la fois une installation plus rapide, parfaitement automatisable pour de grands parcs matériels (sans devoir piloter une interface graphique d'installation) et utilisant moins d'espace disque que dans les versions précédentes où tout était systématiquement installé.

Cette prise en compte des contraintes d'installation pour de grands parcs est une très bonne chose.

Les différents packages existants à ce jour sont :

Nom du package	Contenu
Instant Client	
Basic	Tous les fichiers nécessaires à des applications s'appuyant sur les couches Oracle Call Interface (OCI), Oracle C++ Call Interface (OCCI) et Java Database Connectivity (JDBC-OCI).
JDBC Supplement	Ajout du support transactionnel XA, de l'Internationalisation et des opérations sur les groupes de lignes (RowSet) sous JDBC.
SQL*Plus	Ajout des bibliothèques et des exécutables pour utiliser SQL*Plus avec Instant Client.
ODBC Supplement	Uniquement sous Windows. Ajout des bibliothèques Oracle nécessaires à un accès ODBC.
SDK	Software Développement Kit permettant de développer des interfaces de communication (cas rare).

Les plates-formes disponibles sont pratiquement toutes celles où Oracle est diffusé : Windows 32 et 64 bits, Solaris, Linux, y compris la version pour processeur Itanium, HP/UX, IBM Aix.

Lors de la rédaction de cet ouvrage, quelques limitations d'Instant Client sont signalées. Par exemple, le support de .NET n'est pas encore assuré. Pour disposer d'informations récentes à ce sujet, consultez le site <http://otn.oracle.com> ainsi que le forum Oracle consacré à Instant Client.

Qu'est-ce que le middleware Oracle Net ?

La majorité des experts s'accorde à définir le middleware comme un regroupement de logiciels et de matériels mis en œuvre entre l'application cliente et le serveur de données.

La partie matérielle n'est pas de la compétence d'Oracle, c'est à vous qu'il incombe de choisir le type de réseau, son câblage, les cartes de communication, etc. Néanmoins, Oracle s'interface avec les logiciels que vous aurez choisis ou qui seront déjà installés sur le poste.

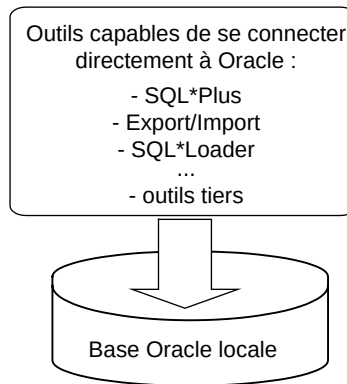
Les processus impliqués dans le fonctionnement d'une communication client-serveur sont abordés au chapitre 7, *Les processus d'Oracle 10g*.

Le middleware se compose d'Oracle Net ainsi que d'outils de communication, comme les composants Java JDBC ou Oracle Objects pour OLE, les pilotes ODBC ou .NET pour les environnements Windows.

Si la base de données et l'application se trouvent sur la même machine, les logiciels de middleware s'interfaçent directement à la base de données.

Figure 15-1

Lien direct entre une application et une base Oracle 10g



Si la base de données et l'application ne sont pas installées sur la même machine, un réseau s'avère indispensable entre le poste client et le serveur. Oracle Net s'appuie sur le protocole réseau que vous utilisez aussi bien sur le poste client que sur le serveur.

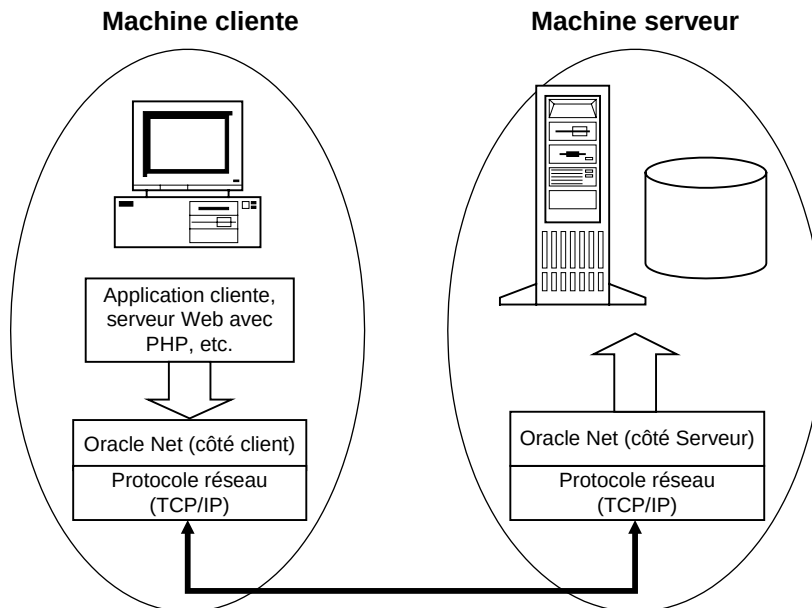


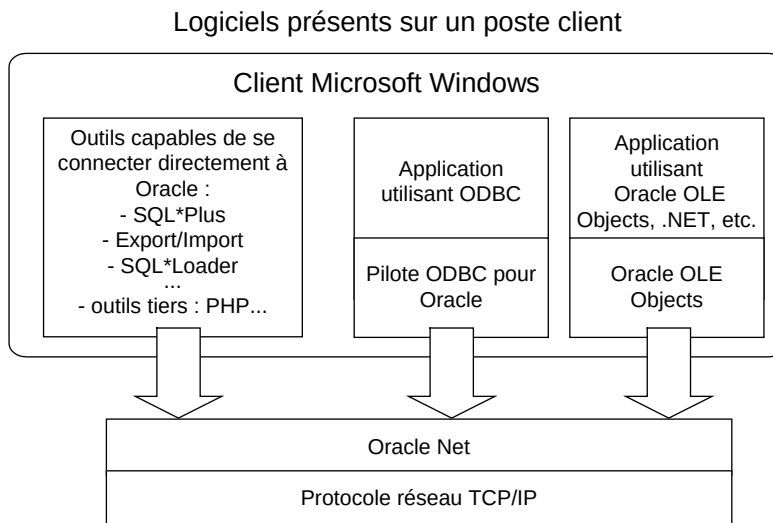
Figure 15-2

Principe d'un lien client-serveur

Dans le cas d'une architecture client-serveur, les logiciels de middleware comprennent Oracle Net et le protocole de communication, généralement TCP/IP. La partie client et la partie serveur d'Oracle Net doivent être installées et configurées séparément.

Figure 15-3

Logiciels présents sur un poste client

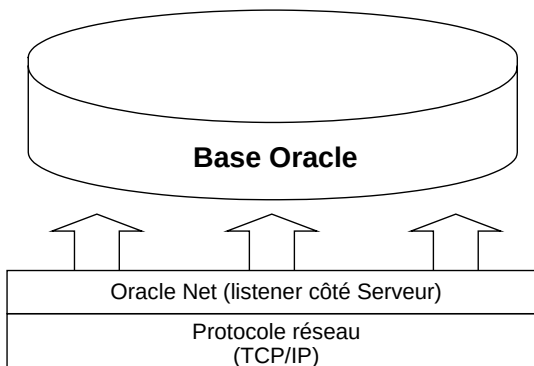


La majorité des outils destinés à fonctionner en client-serveur s'interfaçent à Oracle Net. Certains proviennent d'Oracle, d'autres sont développés par des sociétés tierces. C'est le cas de la majorité des applications et des outils présents sur le marché.

Sur un poste client Windows (plus de 90 % des postes utilisateur...), des logiciels peuvent accéder à Oracle en utilisant un pilote JDBC, le pilote ODBC, l'interface OLE2 ou les nouveaux accès .NET. Un tableur comme Excel a même le choix d'utiliser ODBC ou OLE2 ! C'est à vous de décider du chemin à emprunter, chaque interface ayant ses avantages et ses inconvénients.

Figure 15-4

Logiciels présents sur un serveur



Le schéma technique positionnant l'ensemble des couches logicielles mises en œuvre lors d'une relation client-serveur se présente comme suit :

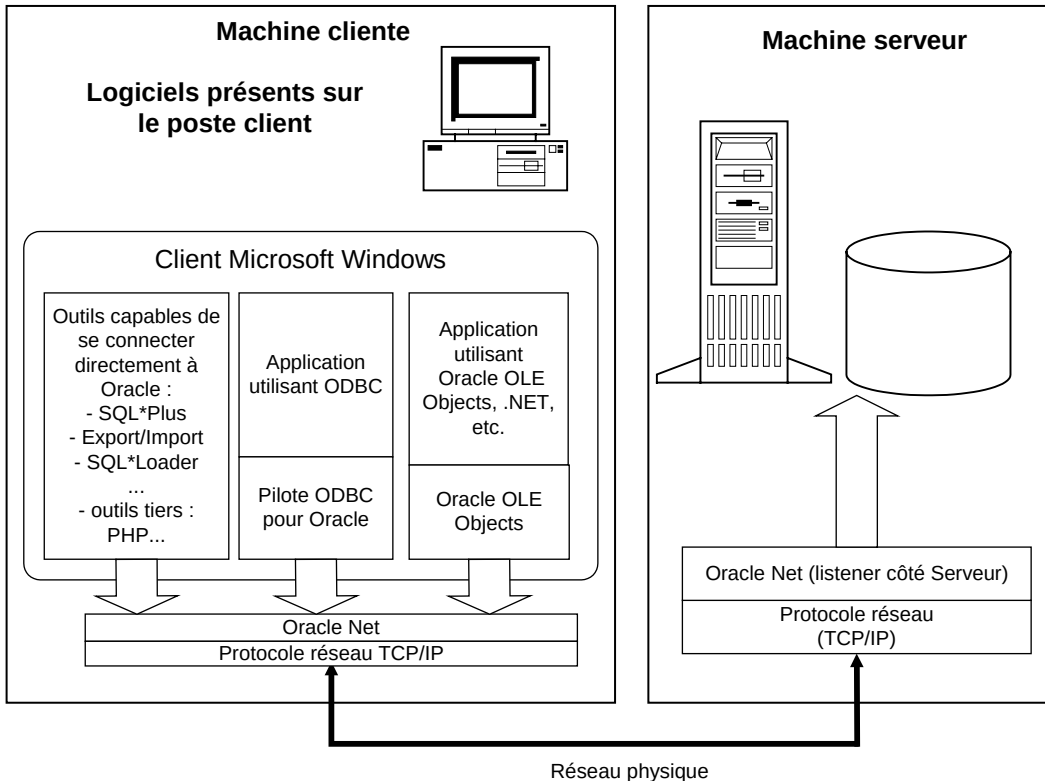


Figure 15-5

Détails d'un lien client-serveur

Ce schéma permet de visualiser le rôle « pivot » d'Oracle Net dans une relation client-serveur.

Cas des serveurs Web connectés à Oracle

Un serveur Web connecté à Oracle possède une architecture similaire à celle de la figure précédente. Le serveur Web (Apache, IIS Internet Information Server) sert de frontal Web. Une application est réalisée grâce à un langage de développement : .Net, PHP, Java, etc. La couche SQL*Net est indispensable pour assurer la communication entre ces langages et la base Oracle, souvent distante.

Dans le cas de .NET et de PHP, le logiciel client Oracle Net doit être installé et configuré sur la machine cliente, c'est-à-dire sur la machine hébergeant le serveur Web. Dans le cas

de Java, un pilote JDBC peut être téléchargé au moment du lancement de l'application. Imaginez JDBC comme une version « légère » d'Oracle Net, téléchargée et ne nécessitant pas de fichiers de configurations. Les possibilités sont moins étendues avec le JDBC téléchargé, mais il n'y a ni installation, ni paramétrage spécifique du client à effectuer.

Comment établir le dialogue entre plusieurs serveurs Oracle ?

Les couches logicielles d'une relation serveur-serveur sont identiques à celles d'une relation client-serveur, chaque serveur étant à un instant donné « client » ou « serveur » du serveur avec lequel il communique.

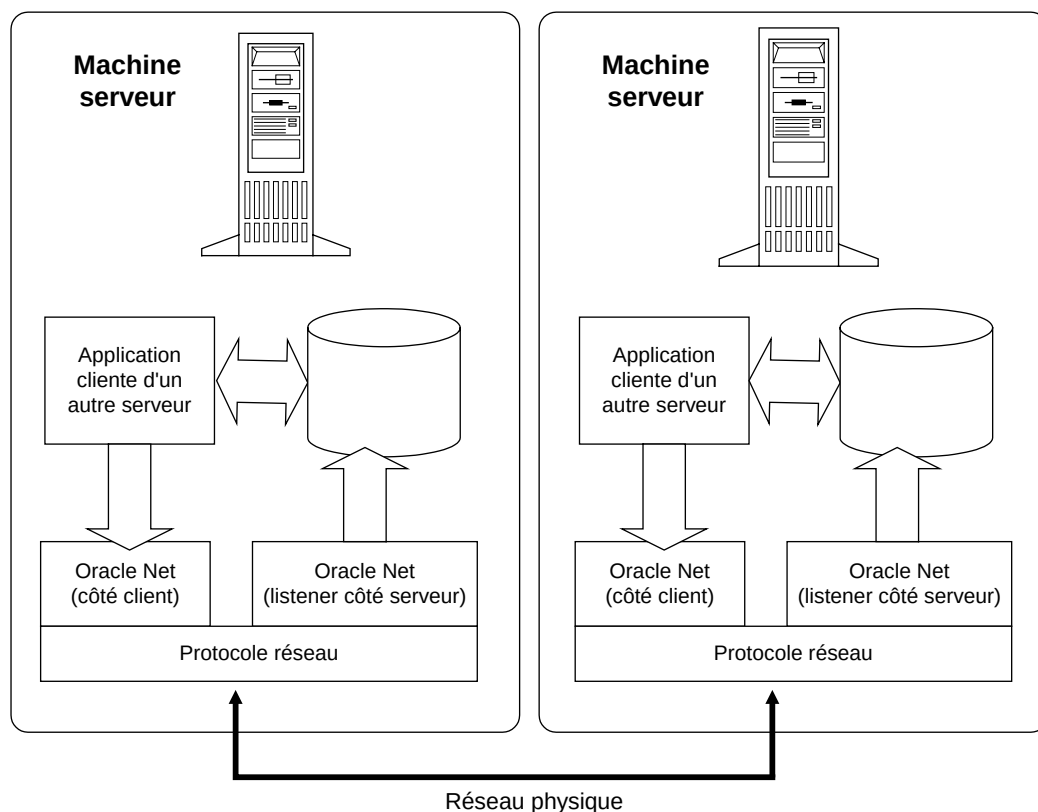


Figure 15-6

Principe d'une relation serveur-serveur

Dans une relation de ce type, Oracle Net client et Oracle Net serveur doivent être installés et configurés sur chaque serveur. Chaque machine devant être configurée à la fois comme serveur et comme client, aucune configuration spécifique nécessaire à un dialogue serveur-serveur n'est à prévoir. À un instant donné, le serveur se comporte comme client d'un autre serveur, tout cela étant parfaitement transparent pour l'utilisateur.

Oracle Net

Oracle Net assure un lien invisible entre les applications du poste client et la base de données sur le serveur. Si le poste client et la base se trouvent sur la *même machine*, il n'est pas forcément utile d'installer Oracle Net mais toute application client-serveur utilisant un réseau en aura besoin.

Pour se connecter à Oracle 10g, il faut fournir trois paramètres :

- le nom de l'utilisateur qui doit être déclaré dans la base Oracle 10g cible ;
- le mot de passe (vérifié dans la base Oracle 10g cible) ;
- un alias pour se connecter à la base cible.

L'alias d'Oracle Net a pour but de regrouper sous un label un ensemble de caractéristiques techniques : quelle est la base de données cible, sur quel serveur elle se situe, etc. Pour récupérer les paramètres techniques depuis l'alias, Oracle propose plusieurs méthodes :

- utiliser sur des fichiers de configuration installés à la fois sur le poste client et sur le serveur cible ;
- utiliser un annuaire centralisé de type LDAP. L'annuaire Active Directory de Microsoft ou une base Oracle Internet Directory spécifique doivent préalablement être créées ;
- utiliser un service de résolution de noms, Oracle Names.

Oracle Names est en passe de disparaître, il sera remplacé par l'annuaire LDAP dont la configuration dépasse les objectifs de ce livre. Nous ne commenterons donc que la première méthode, basée sur des fichiers de configuration. Cela ne constitue pas une limite car ce sont des paramètres identiques qu'il convient d'établir quelle que soit la méthode de résolution d'alias utilisée.

L'alias regroupe sous un nom un ensemble de renseignements :

- le protocole réseau utilisé pour accéder à la machine cible (TCP/IP dans la majorité des cas) ;
- le nom ou l'adresse de la machine cible sur laquelle se situe la base ;
- le SID cible car la machine distante peut héberger plusieurs bases ;
- d'autres paramètres dépendant du protocole utilisé.

Le fichier contenant l'ensemble des paramètres regroupés sous un alias figure plus loin dans ce chapitre.

Le Listener Oracle

Les paramètres regroupés sous un alias significatif sont ceux qui sont « envoyés » à la machine cible. Celle-ci doit être à l'écoute des demandes de connexion : c'est le rôle du Listener Oracle. Il doit être paramétré et lancé, opérations décrites dans ce chapitre.

La configuration la plus simple pour un Listener est d'en établir un seul par serveur, ce qui permet de se connecter à toutes les bases hébergées sur le serveur. Une fois démarré, le Listener reconnaît automatiquement toutes les instances démarrées sur ce serveur.

Auparavant, il fallait modifier le fichier de configuration du Listener présent sur le serveur pour ajouter des points d'entrée lorsqu'une nouvelle base de données était créée. Ce n'est plus utile, le Listener reconnaît automatiquement toutes les instances démarrées sur un serveur.

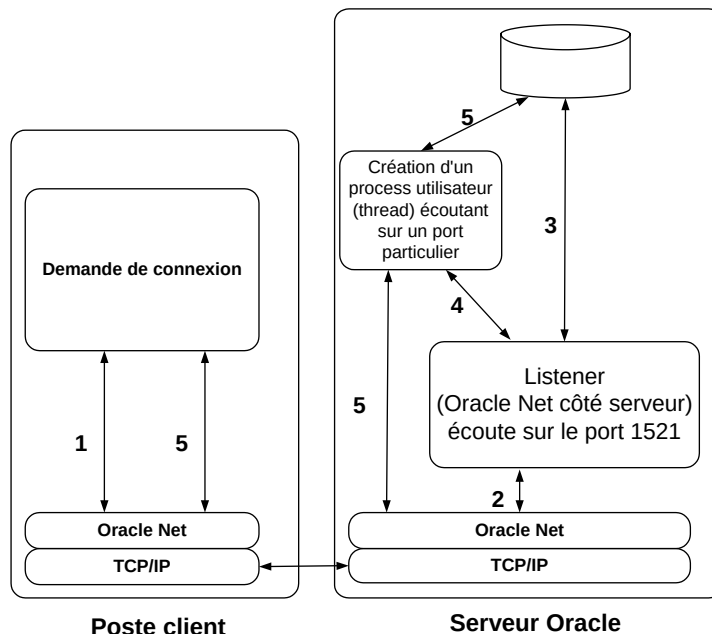
Par contre, pour communiquer avec une nouvelle base, les alias relatifs aux clients doivent être créés à l'aide d'un assistant car le nom de l'alias est généralement significatif pour les utilisateurs.

Principe d'une connexion Oracle Net

Les étapes d'une connexion client-serveur sont souvent ignorées. Pourtant, tout administrateur Oracle doit connaître le principe exposé ci-dessous afin de dialoguer efficacement avec son administrateur réseau.

Figure 15-7

Établissement
d'une connexion



Étape n° 1 : le logiciel client effectue une demande de connexion. Elle transite par Oracle Net puis TCP/IP jusqu'au serveur. Le fichier de paramétrage `tnsnames.ora` du poste client précise l'adresse IP du serveur, la base cible avec laquelle il souhaite entrer en relation et le port IP de communication sur lequel le Listener doit être en écoute.

Étape n° 2 : le Listener Oracle écoute TCP/IP sur le port par défaut n° 1521. Il reçoit la demande de connexion.

Étape n° 3 : le Listener attend le nom d'utilisateur, le mot de passe et vérifie les droits d'accès à la base.

Étape n° 4 : une fois l'autorisation validée, le Listener crée un processus utilisateur (un thread sous Windows) sur le serveur. Il retourne au client Oracle un nouveau port IP « privé » qui servira pour la communication entre le client et son processus utilisateur dédié. Ce numéro de port est pris de manière aléatoire dans une plage réservée par Oracle. Un port unique pour chaque processus est une garantie de performance, car plusieurs milliers d'utilisateurs peuvent simultanément établir une connexion Oracle. Le rôle du processus utilisateur est précisé au chapitre 7, *Les processus d'Oracle 10g*.

Étape n° 5 : le client est maintenant connecté au serveur. Il dialogue directement via le réseau avec le processus utilisateur sur le numéro de port indiqué.

Un serveur Web accepte plusieurs milliers d'utilisateurs sur un numéro de port unique, généralement le port 80. Pourquoi faut-il à Oracle un numéro de port unique pour chaque processus utilisateur ? La raison est simple : la connexion à un serveur Web est asynchrone, elle maintient (rarement) le contexte de l'utilisateur. Oracle établit une session permanente entre le client et le serveur. La charge de gestion est donc beaucoup plus importante pour une session Oracle que pour une liaison avec un serveur Web.

Fonctionnement d'Oracle Net avec un firewall ou un Proxy

Dans cette architecture, faites attention si vous intercalez un firewall ou un Proxy entre un poste client (par exemple un serveur Web situé en DMZ) et un serveur Oracle situé en interne. Si vous ouvrez uniquement le port 1521 du firewall, vous constaterez que le client et le serveur ne peuvent pas entrer en communication.

En effet, pour que la session puisse traverser le firewall, il faudrait ouvrir une énorme brèche dans votre firewall pour laisser passer l'ensemble de ces ports. C'est la raison pour laquelle vous devez avoir un firewall certifié Oracle Net, seul capable d'ouvrir et de fermer dynamiquement un port IP. Heureusement, il existe un contournement simple.

Le cas évoqué avec Oracle Net est identique avec JDBC.

Contournement conseillé

Pour éviter l'achat d'un firewall spécifique, utilisez Oracle Connection Manager placé sur un serveur à l'extérieur de la zone protégée par le firewall.

Un autre moyen consiste à forcer l'utilisation du port du Listener pour router les échanges entre le process serveur et le poste client. Le port IP « privé » devient alors public et correspond toujours à celui du Listener. Pour cela, il suffit de positionner la variable d'environnement `USER_SHARED_SOCKET` à `TRUE` (base de registre sous Windows, variable d'environnement sur Unix), et de relancer la base de données et le Listener (ainsi que le service de la base de données sous Windows).

Administration du Listener

Le démarrage et l'arrêt du Listener sont gérés manuellement ou automatiquement lors du démarrage de l'ordinateur. Des fichiers *log* permettent de repérer toutes les connexions à la base de données.

Les fichiers relatifs au fonctionnement d'Oracle Net seront placés sous Windows dans `C:\oracle\product\10.1.0\db_1\NETWORK\ADMIN`. Il existe trois fichiers principaux :

- `listener.ora` pour configurer le Listener ;
- `tnsnames.ora` pour configurer les alias utilisés lors de la connexion des clients ;
- `sqlnet.ora` qui contient des informations généralistes.

Ces fichiers ont une structure complexe. Heureusement, Oracle a conçu des assistants pour vous aider à les concevoir.

Les assistants Oracle Net : `netca` et `netmgr`

Les assistants Oracle Net vous aident dans toutes les tâches de conception des fichiers de configuration d'Oracle Net. Ces utilitaires sont écrits en Java, ce qui permet de les retrouver sur tous les serveurs où Oracle est disponible : Windows NT, Windows 2000, Linux, Unix, etc. Ils sont d'une utilisation fort pratique.

Le premier d'entre eux, *netca* (*Net Configuration Assistant*), permet de créer très simplement l'ensemble des fichiers de configuration.

Le second, *netmgr* (*Net Manager*), est beaucoup plus complet. Il permet d'accéder à l'ensemble des paramètres pouvant figurer dans les fichiers de configuration Oracle Net, tant sur le serveur que sur le poste client.

Comme tous les utilitaires graphiques, leur mise en œuvre nécessite de comprendre les actions réalisées, c'est pourquoi nous les commentons tous les deux.

Nous allons expliquer le fonctionnement de ces assistants à l'aide du cas concret ci-après.

Configuration du Listener

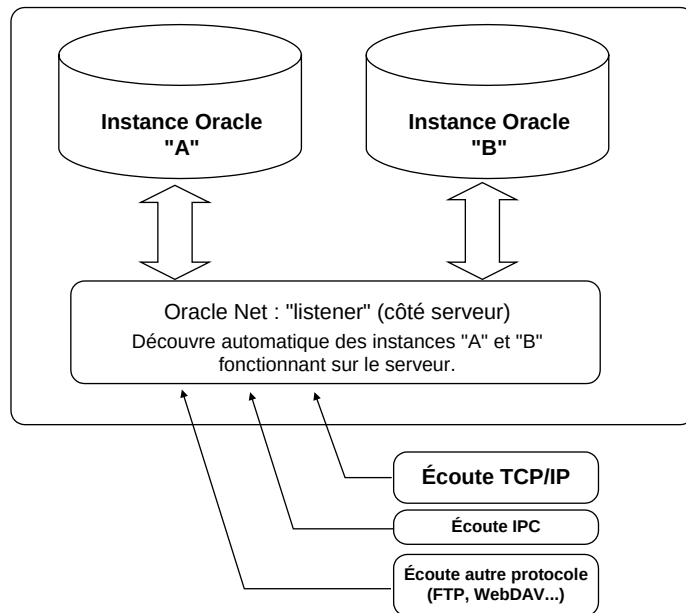
Chacun des deux assistants permet de configurer le Listener. C'est une tâche qui s'est beaucoup simplifiée depuis les dernières versions d'Oracle et les modifications à y apporter sont rares.

Le schéma suivant résume les deux grandes fonctions assurées par le Listener :

- écouter sur plusieurs réseaux (TCP/IP, IPC...) ;
- acheminer les communications vers l'instance cible.

Figure 15-8

Listener : liens avec les bases et les protocoles réseau



On retrouvera donc ces deux aspects dans le fichier listener.ora : une partie pour écouter les différents réseaux, l'autre pour router vers une instance ou l'autre.

Comme la découverte des instances est automatique, aucune action n'est requise lorsqu'on crée une nouvelle base de données.

Pour toute configuration Oracle Net, il est préférable de commencer par celle du Listener, c'est-à-dire du fichier C:\oracle\product\10.1.0\db_1\NETWORK\ADMIN\listener.ora.

Dans tous nos exemples, lors de la fermeture ou de l'ouverture de l'assistant, une sauvegarde des anciens fichiers de configuration réseau est réalisée en C:\oracle\product\10.1.0\db_1\NETWORK\ADMIN. Lors d'un test, vous avez ainsi la possibilité de revenir à la configuration initiale des fichiers.

Configuration d'Oracle Net avec netca

L'assistant Oracle Net Configuration Assistant (*netca*) vous guide pas à pas dans la réalisation d'une configuration Oracle Net. Celle-ci se réalise en trois étapes, correspondant aux trois fichiers de configuration :

- le fichier `listener.ora` pour la configuration du Listener (on le trouvera uniquement sur les serveurs Oracle) ;
- le fichier `sqlnet.ora` pour la configuration de la méthode de résolution d'alias ;
- le fichier `tnsnames.ora` pour la configuration du fichier contenant les alias.

Ces trois fichiers sont situés en `C:\oracle\product\10.1.0\db_1\NETWORK\ADMIN`.

Première étape : configuration du Listener

Lancez l'assistant *netca* grâce au menu Démarrer > Programmes > Oracle - *OraDb10g_home1* > Configuration and Migration Tools Oracle > Net Configuration Assistant.

Figure 15-9

Écran d'accueil de l'assistant *netca*



Le premier écran de l'assistant permet de sélectionner la configuration du Listener.

Figure 15-10

Ajouter un Listener



Considérons que nous partons d'une situation vierge et que nous allons créer un nouveau Listener.

Figure 15-11

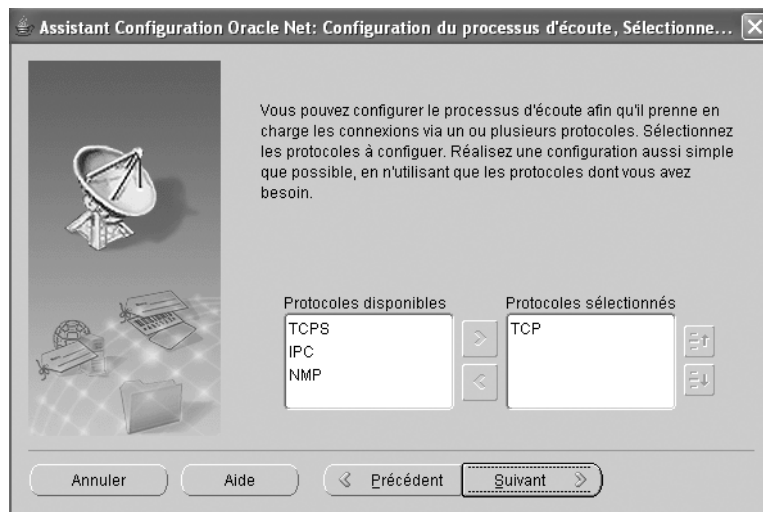
Nom du Listener



Pour ne pas compliquer la configuration, le Listener sera nommé LISTENER.

Figure 15-12

Choix des protocoles



Un Listener peut écouter plusieurs protocoles de communication différents. Dès que vous empruntez un réseau distant, le protocole TCP est le plus utilisé.

Le Listener peut aussi écouter un protocole IPC (*Inter Processus Communication*). Ce port simule une connexion client-serveur sans avoir recours au réseau. Si l'accès IPC est plus rapide que le TCP/IP, il est en revanche limité, car le programme et la base de données doivent être situés sur la même machine.

Figure 15-13

Choix du numéro de port



Par défaut, le Listener écoute sur le port 1521 du protocole TPC. Comme ce numéro lui est exclusivement réservé, pour aider les administrateurs réseau, il est d'usage de l'indiquer dans le fichier `services` en ajoutant la ligne suivante :

```
listener            1521/tcp    # réservé pour listener Oracle
```

Figure 15-14

Fin de la configuration du Listener



La configuration de Listener est terminée. Le fichier `listener.ora` est généré en `ORACLE_HOME\NETWORK\ADMIN`, ce qui correspond dans nos exemples à `C:\oracle\product\10.1.0\db_1\NETWORK\ADMIN\listener.ora`. Son contenu reprend les éléments saisis :

```
# listener.ora Network Configuration File:
# C:\oracle\product\10.1.0\db_1\network\admin\listener.ora
# Generated by Oracle configuration tools.

# cette étape de configuration permet d'appeler
# des procédures externes à la base depuis Oracle
SID_LIST_LISTENER =
  (SID_LIST =
    (SID_DESC =
      (SID_NAME = PLSExtProc)
      (ORACLE_HOME = C:\oracle\product\10.1.0\db_1)
      (PROGRAM = extproc)
    )
  )

# configuration du Listener
LISTENER =
  (DESCRIPTION_LIST =
    (DESCRIPTION =
```

```
(ADDRESS_LIST =  
  (ADDRESS = (PROTOCOL = TCP)(HOST = ORDIGB)(PORT = 1521))  
)  
(ADDRESS_LIST =  
  (ADDRESS = (PROTOCOL = IPC)(KEY = EXTPROC))  
)  
)  
)
```

Deuxième étape : méthode de la résolution de noms

Cette étape correspond au paramétrage du fichier sqlnet.ora.

Figure 15-15

Méthode de résolution de noms



Choisissez l'option *Configuration des méthodes de résolution de noms*.

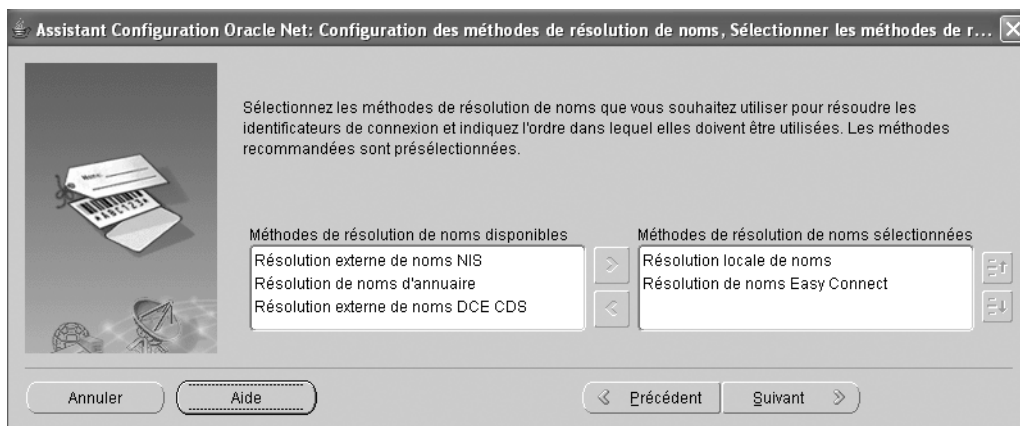


Figure 15-16

Méthode de résolution de noms

Comme nous l'avons présenté, nous établissons une configuration locale qui s'appuie sur des fichiers, sans recourir à un annuaire de type LDAP.

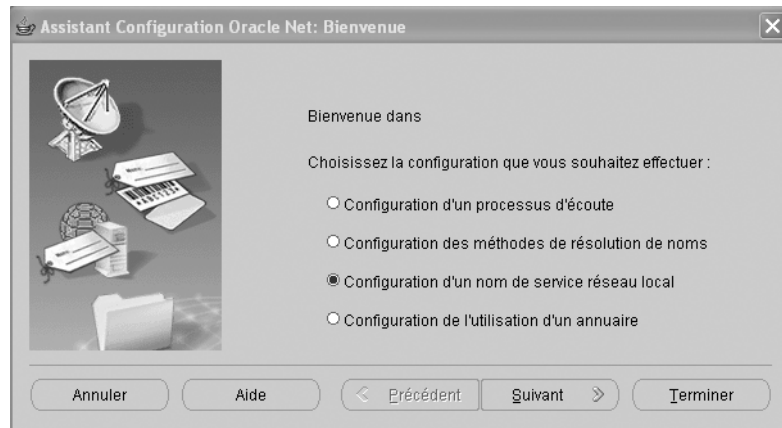
La configuration de la méthode de résolution de noms est terminée. Le fichier sqlnet.ora est généré en \$ORACLE_HOME\NETWORK\ADMIN\sqlnet.ora, ce qui correspond dans nos exemples à C:\oracle\product\10.1.0\db_1\NETWORK\ADMIN\sqlnet.ora. Son contenu reprend les éléments saisis :

```
# sqlnet.ora Network Configuration File:  
# C:\oracle\product\10.1.0\db_1\network\admin\sqlnet.ora  
# Generated by Oracle configuration tools.  
  
SQLNET.AUTHENTICATION_SERVICES= (NTS)  
  
NAMES.DIRECTORY_PATH= (TNSNAMES, EZCONNECT)
```

Troisième étape : configuration des alias

Cette étape correspond au paramétrage du fichier tnsnames.ora.

Figure 15-17
Configuration des alias



Choisissez l'option *Configuration d'un nom de service réseau local*.



Figure 15-18

Ajout d'un alias

Nous allons ajouter un nouvel alias.



Figure 15-19

Base de données cible

Les éléments à saisir diffèrent selon que la base de données cible est d'une version antérieure ou postérieure à Oracle version 8.1.7. Pour cet exemple, considérons que la base cible est une base Oracle 9i ou 10g.

Donnez le nom du service de base de données, c'est-à-dire du nom global de base de données ou GLOBAL_DBNAME.

Pour vérifier le nom global donné à une base :

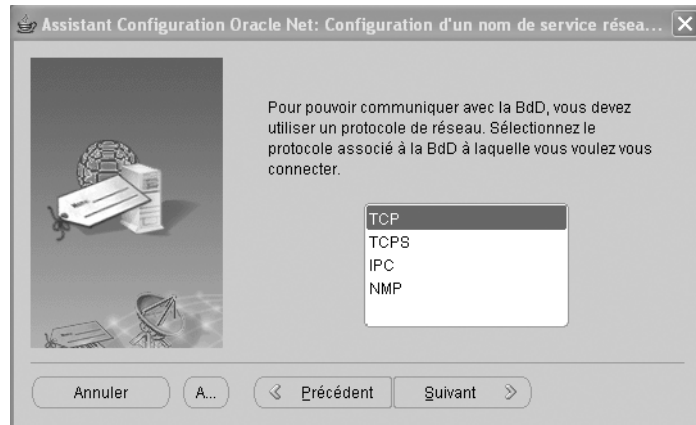
```
# connectez-vous
sqlplus system/mot_de_passe

# visualisez les paramètres d'initialisation de la base
show parameters GLOBAL_DBNAME

# GLOBAL_DBNAME figure dans cette liste.
```

Figure 15-20

Choix du protocole



Le protocole utilisé entre le client et le serveur sera TCP/IP.

Figure 15-21

Nom du serveur cible



Lors du choix d'un protocole, assurez-vous qu'il a été configuré pour le Listener cible. Si ce n'est pas le cas, la communication client-serveur sera impossible.

Indiquez le nom du serveur cible qui héberge le Listener et la base, ainsi que le port de communication utile au protocole TCP.

Testez votre configuration. Si vous utilisez *netca* sur le serveur qui héberge la base de données, assurez-vous que tout fonctionne :

```
# Test de connexion sans Oracle Net.  
sqlplus user/password # si la connexion fonctionne, quittez SQL*Plus  
  
# Vérifiez que le listener a été lancé  
lsnrctl status  
  
# Vous pouvez maintenant utiliser l'assistant pour tester Oracle Net
```

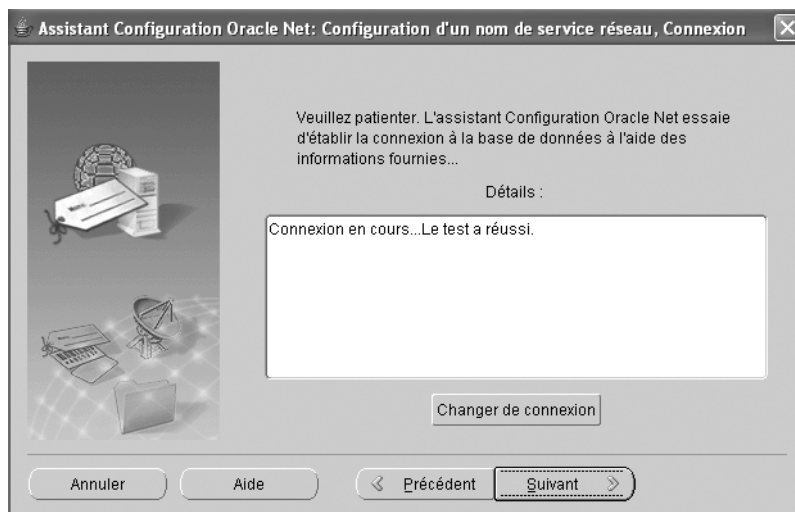


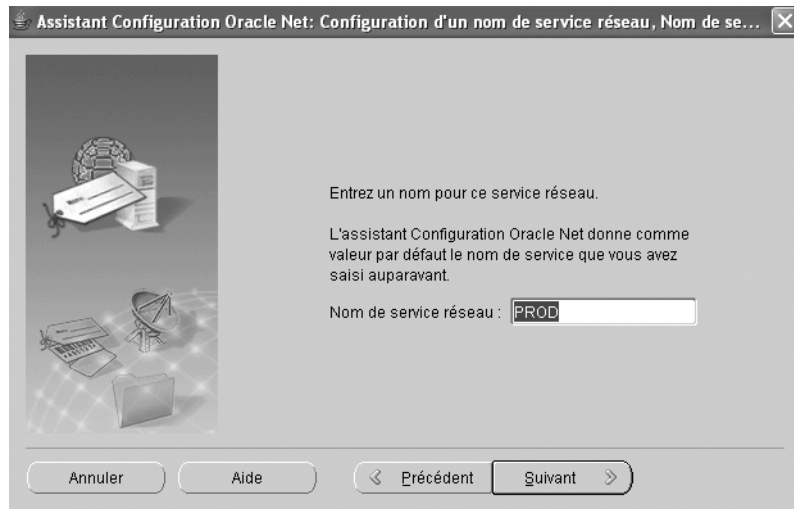
Figure 15-22

Test de la configuration de l'alias

Lors du test, vous pouvez modifier l'utilisateur sous lequel la connexion est effectuée (figure 15-23).

La configuration de l'alias a été testée : reste à donner un nom à cet alias.

La configuration est terminée. Le fichier `tnsnames.ora` est généré en `$ORACLE_HOME\network\admin\tnsnames.ora`, ce qui correspond dans nos exemples à `C:\oracle\product\10.1.0\db_1\NETWORK\ADMIN\tnsnames.ora`.

**Figure 15-23***Nom donné à l'alias*

Son contenu reprend les éléments saisis :

```
# tnsnames.ora Network Configuration File: C:\oracle\product\10.1.0\db_1\network\ad
min\tnsnames.ora
# Generated by Oracle configuration tools.

PROD =
  (DESCRIPTION =
    (ADDRESS_LIST =
      (ADDRESS = (PROTOCOL = TCP)(HOST = ORDIGB)(PORT = 1521))
    )
    (CONNECT_DATA =
      (SERVICE_NAME = PROD.gilles.fr)
    )
  )

TEST =
  (DESCRIPTION =
    (ADDRESS_LIST =
      (ADDRESS = (PROTOCOL = TCP)(HOST = ORDIGB)(PORT = 1521))
    )
    (CONNECT_DATA =
      (SERVICE_NAME = TEST.gilles.fr)
    )
  )
```

Dans l'exemple précédent, deux alias figurent dans le fichier : celui que nous venons de créer ainsi que l'alias TEST correspondant à une autre base lancée sur le même serveur.

N'oubliez pas que les alias décrits dans le fichier `tnsnames.ora` sont destinés à entrer en relation avec un Listener qui doit être configuré et lancé pour répondre à leurs demandes.

Enfin, *netca* permet de ne plus dépendre de fichiers de configuration mais d'aller rechercher les informations concernant les fichiers `listener.ora`, `tnsnames.ora`, `sqlnet.ora` directement dans un annuaire LDAP.



Figure 15-24

Stockage des informations dans un annuaire LDAP

Le moyen le plus simple est toutefois de conserver l'utilisation des fichiers. Nous ne sommes plus à l'époque du client-serveur où tous les PC distants devaient être configurés pour accéder à Oracle. Maintenant, avec les applications en mode Web, seuls quelques serveurs Web doivent être équipés d'Oracle Net. La charge d'installation et de maintenance en est réduite d'autant.

Configuration d'Oracle Net avec netmgr

L'assistant Oracle Net Manager ou *netmgr* est beaucoup plus complet que l'assistant *netca* étudié précédemment. Il permet d'accéder à l'ensemble des paramètres pouvant figurer dans les fichiers de configuration Oracle Net.

Dans les exemples qui suivent, nous reprenons la configuration établie avec *netca* pour vous commenter *netmgr*.

Lancez l'assistant grâce au menu Démarrer > Programmes > Oracle - OraDb10g_home1 > Configuration and Migration Tools Oracle > Net Manager.

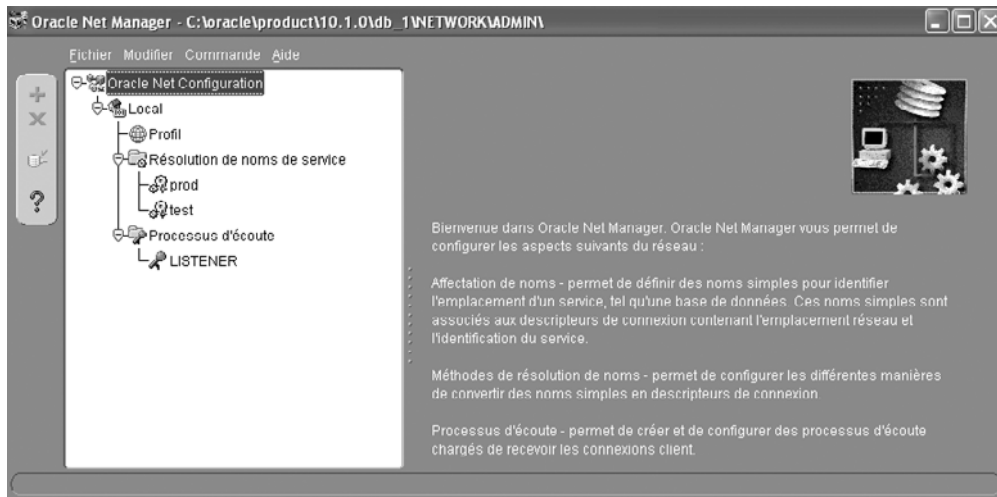


Figure 15-25

Écran d'accueil de netmgr

L'assistant *netmgr* reprend les trois étapes correspondant aux trois fichiers de configuration à créer :

- option *Profil* pour configurer la méthode de résolution d'alias (fichier *sqlnet.ora*) ;
- option *Résolution de noms de sService* pour configurer les alias (fichier *tnsnames.ora*) ;
- option *Listener* pour configurer le Listener (fichier *listener.ora*).

Les fichiers de configuration manipulés sont situés dans le répertoire indiqué sur le bandeau supérieur de la figure.

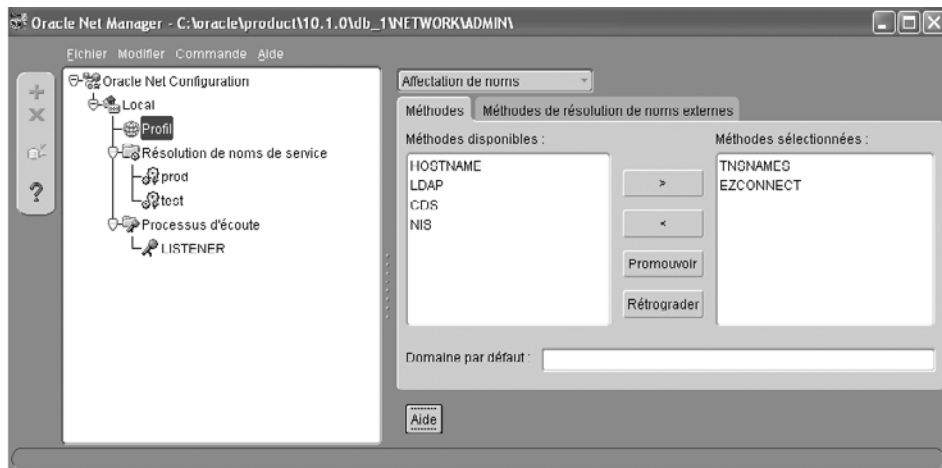


Figure 15-26

Configuration de la méthode de résolution de noms

La méthode de résolution de noms s'appuie sur les fichiers de configuration. Remarquez que, pour chaque option, plusieurs onglets sont proposés.

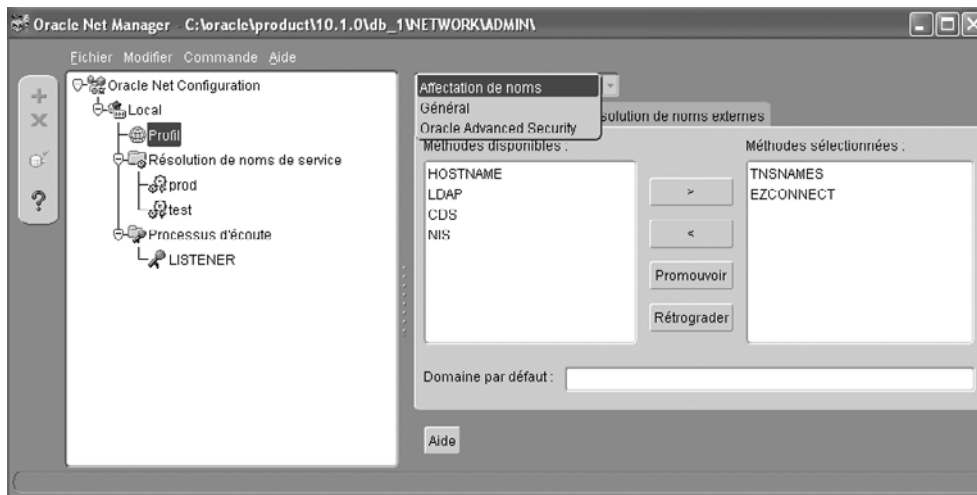


Figure 15-27

Options de configuration

Plusieurs rubriques détaillent les options de configuration de *Profil*.

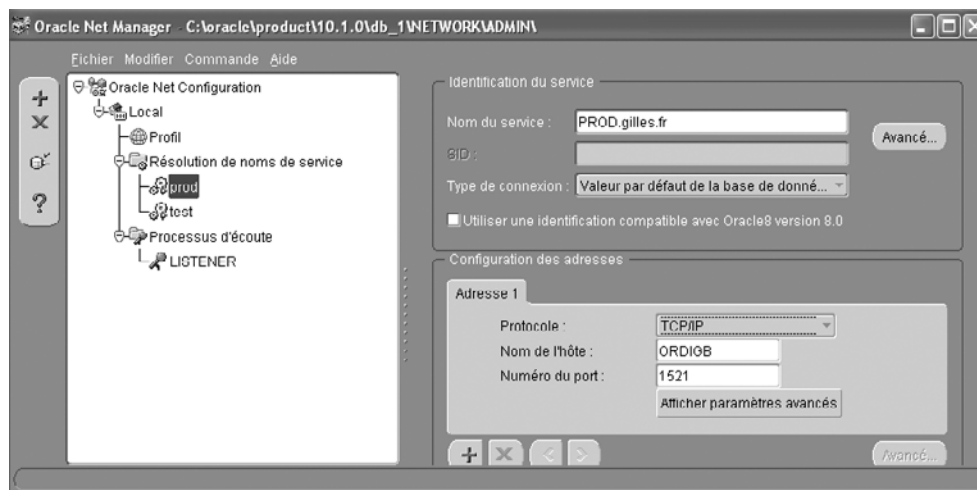


Figure 15-28

Configuration des alias

On retrouve les éléments saisis lors de la configuration des alias figurant dans le fichier tnsnames.ora.

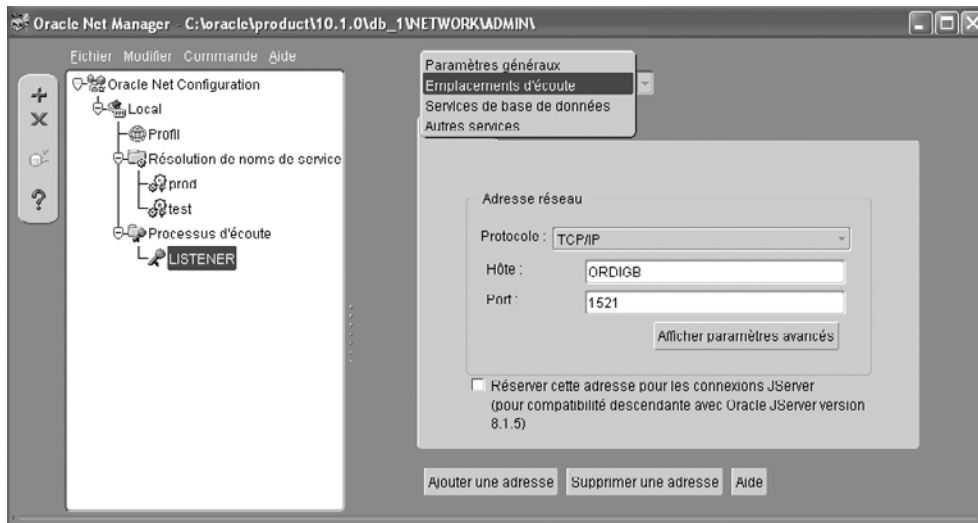


Figure 15-29

Détail du Listener

Détail de la configuration du Listener.

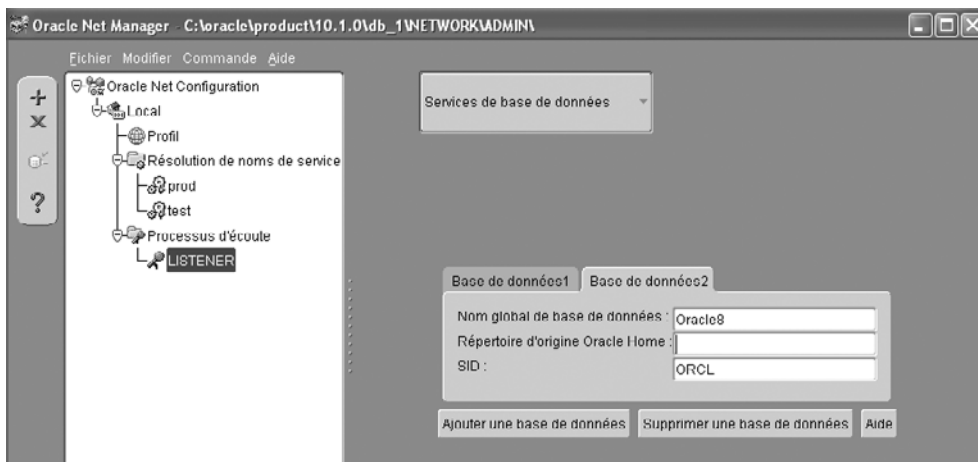


Figure 15-30

Recherche des bases

Cette dernière figure indique que le Listener reconnaît automatiquement toutes les instances démarrées sur le serveur.

Si le Listener doit accéder à des bases Oracle de version antérieure à la version 8.1.x, vous devez les déclarer manuellement en ajoutant des *Database Services*. C'est aussi le cas pour que l'Agent Intelligent d'Oracle Enterprise Manager puisse découvrir des bases.

Démarrer/arrêter le Listener

Le moyen le plus simple pour démarrer/arrêter un Listener consiste à lancer et arrêter le service Windows correspondant. Si l'Oracle Home a pour nom *OraDb10g_home1*, le service Windows aura pour nom **OracleOraDb10g_home1TNSListener**. Nous vous conseillons de toujours démarrer le Listener automatiquement, en même temps que l'ordinateur.

Comme tout service, vous pouvez le lancer et l'arrêter manuellement par la commande `net start service`, `net stop service` :

```
C:\> net start OracleOraDb10g_home1TNSListener
C:\> net stop OracleOraDb10g_home1TNSListener
```

Vous pouvez aussi accéder à l'outil de gestion du Listener en mode « ligne de commande » : `lsnrctl` (moyen mnémotechnique : *listener control*). Cette application possède plusieurs options.

Lancez l'outil de contrôle du Listener :

```
C:\> lsnrctl
LSNRCTL>
```

L'invite `LSNRCTL>` indique que nous sommes dans l'outil de contrôle. Les commandes les plus utiles sont les suivantes :

```
LSNRCTL> help
LSNRCTL> start
LSNRCTL> stop
LSNRCTL> status
```

Vous pouvez lancer ces commandes directement en mode ligne de commande sous l'invite Windows :

```
lsnrctl start
lsnrctl stop
lsnrctl status
...
```

La commande suivante est utilisée pour démarrer le Listener. Dans cet exemple, aucune base de données n'est démarrée sur le serveur lors du lancement du Listener :

```
lsnrctl start
```

```
LSNRCTL for 32-bit Windows: Version 10.1.0.2.0 - Production on 25-MAI -2005 21:58:33
```

Copyright (c) 1991, 2004, Oracle. All rights reserved.

Lancement de tnslnsr: Veuillez patienter...

TNSLSNR for 32-bit Windows: Version 10.1.0.2.0 - Production

Le fichier de paramètres système est C:\oracle\product\10.1.0\db_1\network\admin\listener.ora

Messages de journalisation écrits dans C:\oracle\product\10.1.0\db_1\network\log\listener.log

Ecoute sur : (DESCRIPTION=(ADDRESS=(PROTOCOL=tcp)(HOST=ordigb)(PORT=1521)))

Ecoute sur : (DESCRIPTION=(ADDRESS=(PROTOCOL=ipc)(PIPENAME=\\.\pipe\EXTPROCipc)))

Connexion à (DESCRIPTION=(ADDRESS=(PROTOCOL=TCP)(HOST=ORDIGB)(PORT=1521)))

STATUT du PROCESSUS D'ECOUTE

```
Alias                LISTENER
Version              TNSLSNR for 32-bit Windows: Version 10.1.0.2.0 - Production
Date de départ       25-MAI  -2005 21:58:34
Durée d'activité     0 jours 0 heures 0 min. 1 sec
Niveau de trace      off
Sécurité             ON: Local OS Authentication
SNMP                 OFF
Fichier de paramètres du processus d'écoute
➔C:\oracle\product\10.1.0\db_1\network\admin\listener.ora
Fichier journal du processus d'écoute
➔C:\oracle\product\10.1.0\db_1\network\log\listener.log
Récapitulatif d'écoute des points d'extrémité...
  (DESCRIPTION=(ADDRESS=(PROTOCOL=tcp)(HOST=ordigb)(PORT=1521)))
  (DESCRIPTION=(ADDRESS=(PROTOCOL=ipc)(PIPENAME=\\.\pipe\EXTPROCipc)))
Récapitulatif services...
Le service "PLSExtProc" comporte 1 instance(s).
  L'instance "PLSExtProc", statut UNKNOWN, comporte 1 gestionnaire(s) pour ce service...
La commande a réussi
LSNRCTL>
```

Comme aucune base n'a été lancée, la commande status contient peu d'informations :

lsnrctl status

LSNRCTL for 32-bit Windows: Version 10.1.0.2.0 - Production on 25-MAI -2005 21:58:47

Copyright (c) 1991, 2004, Oracle. All rights reserved.

Connexion à (DESCRIPTION=(ADDRESS=(PROTOCOL=TCP)(HOST=ORDIGB)(PORT=1521)))

STATUT du PROCESSUS D'ECOUTE

```
Alias                LISTENER
Version              TNSLSNR for 32-bit Windows: Version 10.1.0.2.0 - Production
Date de départ       25-MAI  -2005 21:58:34
Durée d'activité     0 jours 0 heures 0 min. 13 sec
Niveau de trace      off
Sécurité             ON: Local OS Authentication
```

```

SNMP                                OFF
Fichier de paramètres du processus d'écoute
➔C:\oracle\product\10.1.0\db_1\network\admin\listener.ora
Fichier journal du processus d'écoute
➔C:\oracle\product\10.1.0\db_1\network\log\listener.log
Récapitulatif d'écoute des points d'extrémité...
  (DESCRIPTION=(ADDRESS=(PROTOCOL=tcp)(HOST=ordigb)(PORT=1521)))
  (DESCRIPTION=(ADDRESS=(PROTOCOL=ipc)(PIPENAME=\\.\pipe\EXTPROCipc)))
Récapitulatif services...
Le service "PLSExtProc" comporte 1 instance(s).
  L'instance "PLSExtProc", statut UNKNOWN, comporte 1 gestionnaire(s) pour ce service...
  ➔La commande a réussi

```

Une fois la base TEST lancée, le Listener l'enregistre automatiquement :

```
lsnrctl status
```

```
LSNRCTL for 32-bit Windows: Version 10.1.0.2.0 - Production on 25-MAI -2005 22:04:58
```

```
Copyright (c) 1991, 2004, Oracle. All rights reserved.
```

```
Connexion à (DESCRIPTION=(ADDRESS=(PROTOCOL=TCP)(HOST=ORDIGB)(PORT=1521)))
```

```
STATUT du PROCESSUS D'ECOUTE
```

```

-----
Alias                                LISTENER
Version                             TNSLSNR for 32-bit Windows: Version 10.1.0.2.0 - Production
Date de départ                       25-MAI -2005 21:58:34
Durée d'activité                     0 jours 0 heures 6 min. 24 sec
Niveau de trace                      off
Sécurité                             ON: Local OS Authentication
SNMP                                  OFF
Fichier de paramètres du processus d'écoute
➔C:\oracle\product\10.1.0\db_1\network\admin\listener.ora
Fichier journal du processus d'écoute
➔C:\oracle\product\10.1.0\db_1\network\log\listener.log
Récapitulatif d'écoute des points d'extrémité...
  (DESCRIPTION=(ADDRESS=(PROTOCOL=tcp)(HOST=ordigb)(PORT=1521)))
  (DESCRIPTION=(ADDRESS=(PROTOCOL=ipc)(PIPENAME=\\.\pipe\EXTPROCipc)))
  (DESCRIPTION=(ADDRESS=(PROTOCOL=tcp)(HOST=ordigb)(PORT=8080))(Presentation=HTTP)
  ➔(Session=RAW))
  (DESCRIPTION=(ADDRESS=(PROTOCOL=tcp)(HOST=ordigb)(PORT=2100))(Presentation=FTP)
  ➔(Session=RAW))
Récapitulatif services...
Le service "PLSExtProc" comporte 1 instance(s).
  L'instance "PLSExtProc", statut UNKNOWN, comporte 1 gestionnaire(s) pour ce service...
Le service "TEST.gilles.fr" comporte 1 instance(s).
  L'instance "test", statut READY, comporte 1 gestionnaire(s) pour ce service...
Le service "TESTXDB.gilles.fr" comporte 1 instance(s).
  L'instance "test", statut READY, comporte 1 gestionnaire(s) pour ce service...
La commande a réussi

```


Le comportement est identique après le démarrage de l'instance PROD :

```
lsnrctl status
```

```
. . . le début de l'exemple est identique à l'exemple précédent . . .  
Le service "TEST.gilles.fr" comporte 1 instance(s).  
  L'instance "test", statut READY, comporte 1 gestionnaire(s) pour ce service...  
Le service "TESTXDB.gilles.fr" comporte 1 instance(s).  
  L'instance "test", statut READY, comporte 1 gestionnaire(s) pour ce service...  
Le service "PROD.gilles.fr" comporte 1 instance(s).  
  L'instance "test", statut READY, comporte 1 gestionnaire(s) pour ce service...  
Le service "PRODXDB.gilles.fr" comporte 1 instance(s).  
  L'instance "test", statut READY, comporte 1 gestionnaire(s) pour ce service...  
La commande a réussi
```

Vérifier la présence du Listener sous Windows

Lorsqu'il fonctionne, le Listener Oracle Net est identifié sous Windows par la présence de l'exécutable *tnslsnr*, visible au moyen du Gestionnaire des tâches de Windows.

Contrôler le fichier log du Listener

Le Listener possède un fichier log très utile dans la résolution de problèmes de toutes sortes. Dans notre exemple, il est situé dans \$ORACLE_HOME\network\log\listener.log, mais peut se trouver ailleurs, suivant la valeur définie dans le fichier *listener.ora*.

Veillez à contrôler la taille du fichier *listener.log* et à le purger régulièrement car chaque connexion en mode client-serveur génère une trace.

Lancer le Listener Oracle Net au démarrage du serveur

Au chapitre 13, *L'installation d'Oracle 10g sous Windows*, nous avons étudié le démarrage simultané de l'ensemble des bases et du serveur. Il suffit de lancer automatiquement le service **OracleOraDb10g_home1TNSListener**.

Veillez à toujours tester le démarrage du Listener manuellement au moins une fois, car des messages vous préviennent des erreurs de configuration du fichier *listener.ora*.

Création de nouveaux alias

La création de nouveaux alias Oracle Net ne pose pas de difficulté majeure. Pour cela, il suffit d'utiliser l'assistant Oracle Net Configuration Assistant ou de travailler directement sur le fichier *tnsnames.ora* avec un éditeur de texte.

Où placer les fichiers *tnsnames*, *sqlnet* et *listener* ?

Le fichier *listener.ora* doit être placé uniquement sur le serveur pour lequel il a été conçu.

Les fichiers *tnsnames.ora* et *sqlnet.ora* sont utilisés par tous les postes client. Vous devez en placer une copie sur chaque client. Si le serveur peut être client de lui-même ou d'un autre serveur, il doit disposer également de ce fichier.

Tester une connexion Oracle Net

Il ne vous reste plus qu'à vérifier la connexion avec un compte utilisateur existant. La toute première action consiste à tester sa configuration 100 % en local.

Tester Oracle Net en local

1. Connectez-vous sous *Oracle* et lancez le Listener :

```
lsnrctl start
```

2. Testez la connexion :

```
sqlplus system/mot_de_passe      # test sans Oracle Net
sqlplus system/mot_de_passe@alias # test empruntant Oracle Net
```

3. Testez chaque alias un par un.

Tester Oracle Net en client-serveur

1. Installez Oracle Net Client avec Oracle Universal Installer sur le poste client. Assurez-vous que la machine cliente entre bien en communication avec le serveur (*ftp*, *ping*, *telnet*...).
2. Copiez les fichiers *tnsnames.ora* et *sqlnet.ora* sur le poste client dans le répertoire correspondant à la configuration Oracle Net : C:\oracle\product\10.1.0\db_1\NETWORK\ADMIN.
3. Testez la connexion :
 - `sqlplus system/mot_de_passe` # test sans Oracle Net, ne doit pas fonctionner ;
 - `sqlplus system/mot_de_passe@alias` # test empruntant Oracle Net, doit fonctionner.
4. Testez chaque alias un par un.
5. Vérifiez vos connexions dans le fichier `$ORACLE_HOME\network\log\listener.log` du serveur.

L'installation Oracle Net est maintenant opérationnelle.

Oracle 10g et le Multithreading

Par défaut, pour chaque connexion client, Oracle Net ouvre un processus sur le serveur. Dans les cas de grandes configurations, cela entraîne une surconsommation de mémoire et de processeur.

Oracle 10g autorise l'utilisation du Multithreading. C'est une configuration spéciale d'Oracle Net qui permet de recourir au multiplexage au niveau du serveur, c'est-à-dire de gérer un seul thread au niveau du serveur pour « n » clients connectés. Cela limite la consommation mémoire et processeur. Oracle recommande d'utiliser ce type de configuration à partir de 100 ou 150 utilisateurs connectés. Vous pouvez consulter à cet effet le chapitre 7, *Les processus d'Oracle 10g*.

Coexistence d'Oracle Net et de Net8

Oracle Net est compatible avec les différentes versions de son prédécesseur SQL*Net ou Net8. Cela signifie qu'un poste client utilisant ce dernier est capable d'atteindre un serveur doté d'Oracle Net et de dialoguer avec lui.

Inversement, un client Oracle Net est en mesure de communiquer avec un serveur équipé de SQL*Net ou Net8.

Oracle Net, un middleware intelligent

Oracle Net vous propose beaucoup d'autres options de fonctionnement. Il s'agit bien plus que d'un simple support destiné à véhiculer des paquets sur un réseau.

Il vous permet, par exemple, de spécifier une liste exhaustive d'adresses TCP/IP, c'est-à-dire de clients qui ont le droit d'atteindre vos bases. Le paramétrage s'effectue dans la configuration du Listener.

L'autre caractéristique très importante d'Oracle Net concerne le nombre d'informations transmises au serveur. En consultant la vue V\$SESSION du serveur, vous êtes capable d'identifier très précisément les connexions à la base de données. Vous disposez entre autres de renseignements tels que :

- l'utilisateur de connexion à Oracle 10g ;
- le nom du poste connecté (le nom de la station Windows cliente) ;
- l'utilisateur sous lequel le client est connecté ;
- le nom du programme client utilisé.

Tous ces renseignements sont très utiles pour vos tâches d'administration. Si elles n'étaient pas remontées automatiquement au serveur par Oracle Net, vos applications en auraient la charge.

Comment identifier les sessions connectées ?

Vous pouvez utiliser Oracle Enterprise Manager ou la requête SQL suivante :

```
select sid, serial#, username, status  
from v$session ;
```

SID	SERIAL#	USERNAME	STATUS
---	-----	-----	-----
5	74	SYSTEM	ACTIVE
18	44	SYSTEM	INACTIVE
8	36	SCOTT	KILLED
...			

Les sessions affichées sont identifiées par leur SID et leur SERIAL# qui sont utilisés par l'ordre ALTER SYSTEM KILL SESSION. Dans l'exemple précédent, les sessions sont ACTIVE (un ordre SQL est traité par Oracle), INACTIVE (aucun ordre traité actuellement) ou KILLED (la session a été tuée).

Cette liste est réduite. Normalement, les sessions affectées aux processus qui assurent le fonctionnement de la base y figurent.

Résumé

Oracle Net est un logiciel très complet offrant un nombre important de possibilités. Notre objectif était de vous fournir les éléments essentiels pour comprendre et réussir une configuration sans que celle-ci tourne à l'épreuve.

Nous avons étudié les points suivants :

- la nature de la structure du middleware Oracle Net dans une architecture réseau ;
- l'utilisation de l'assistant Oracle Net pour réaliser l'ensemble des fichiers de configuration ;
- le contenu détaillé des différents fichiers de configuration ;
- la méthode pour modifier manuellement les fichiers de configuration.

16

Accéder à Oracle depuis Windows

Dans ce chapitre :

- la connexion depuis des environnements Microsoft aux bases Oracle 10g ;
- les pilotes ODBC et OLE DB ;
- le pilote Oracle Objects for OLE (OO4O) ;
- les nouveaux accès .NET ;
- quel accès choisir ?
- accéder à Oracle depuis un serveur Web IIS ou Apache ;
- Oracle COM automation : appeler les outils Microsoft depuis Oracle.

Ce chapitre constitue la suite logique des précédents : une fois le logiciel Oracle installé et Oracle Net configuré, quelles sont les possibilités d'accès aux bases de données depuis les applications développées dans un environnement Microsoft ?

Vous n'êtes pas forcément libre de choisir l'accès à Oracle : soit vous utilisez un *progriciel* et dans ce cas, le moyen d'accès est imposé, soit vous *développez* des applications et vous pouvez alors choisir le moyen d'accès accepté par votre outil de développement. Nous présentons les différents moyens d'accès depuis les applications Microsoft Office (Word, Excel, Powerpoint, Access), les outils de développement de Microsoft Visual Studio incluant Visual Basic, Active Server Pages (ASP) et Internet Application Server (IIS). Les environnements .NET sont eux aussi abordés.

Nous décrivons les accès ODBC, Oracle OLE DB, Oracle Objects for OLE (OO4O), ActiveX Data Objects (ADO), Oracle Call Interface (OCI) ainsi que les nouvelles interfaces .NET : ODBC .NET, Oracle OLE DB .NET, Oracle Data Provider for .NET.

Dans ce chapitre, tous les accès présentés sont à l'initiative d'un programme « appelant » Oracle. Or, il est possible d'appeler des logiciels **depuis la base de données**. Ainsi, une procédure PL/SQL ou Java peut appeler et manipuler des objets COM. C'est l'objectif de l'interface Oracle COM Automation, présentée en fin de chapitre.

Les différents accès

Au fil des ans, Microsoft a introduit une multitude de moyens d'accès aux bases de données. Certains sont *génériques*, c'est-à-dire que le même moyen d'accès permet d'accéder à plusieurs bases de données. D'autres sont *natifs* et nécessitent d'adapter l'écriture des programmes pour profiter des toutes les possibilités d'Oracle.

Les pilotes génériques

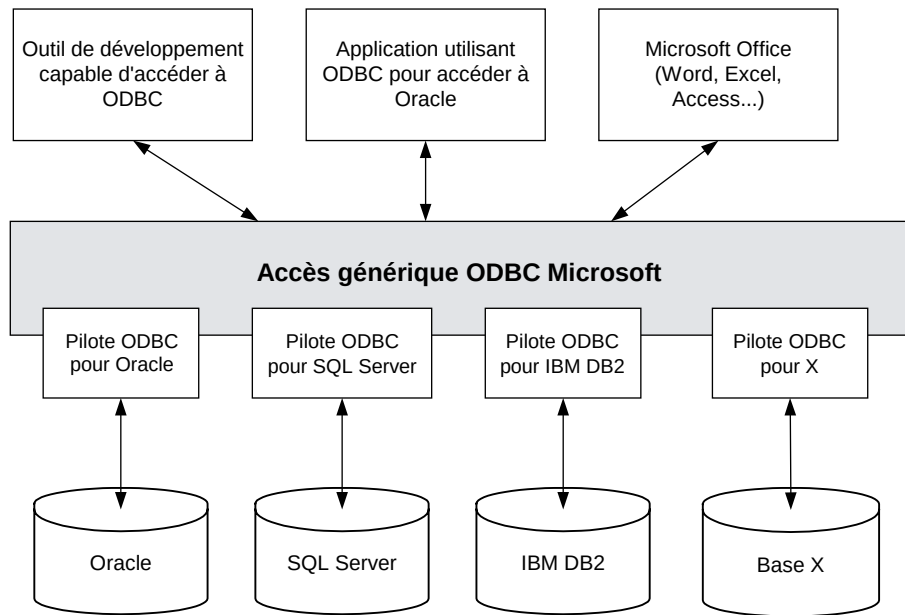
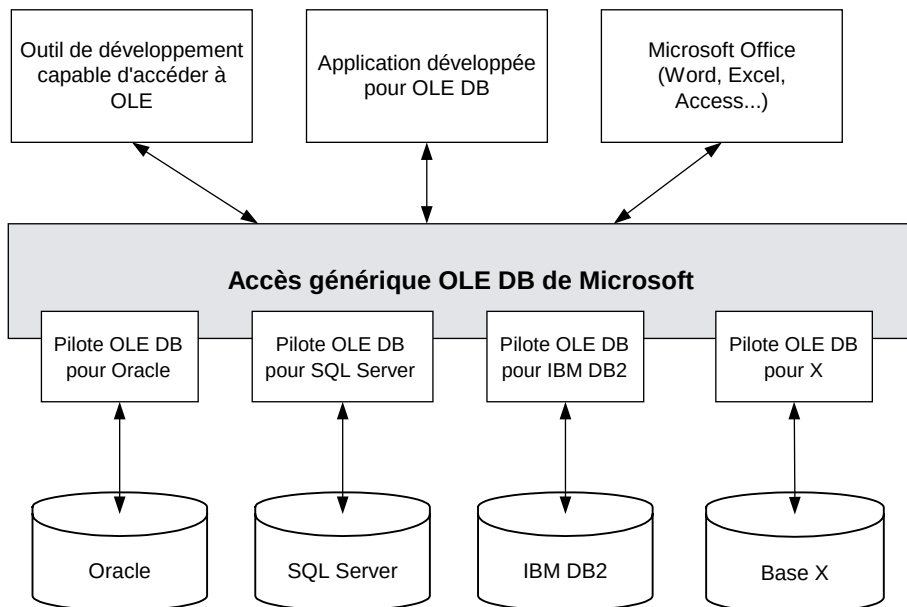
Le pilote ODBC (*Open DataBase Connectivity*), par exemple, est un pilote *générique*. Il a été conçu par Microsoft dans le but de populariser l'accès aux bases de données depuis ses systèmes d'exploitation. L'avantage majeur consiste à proposer une interface de programmation unique utilisée par de multiples outils de développements. L'inconvénient est que l'accès générique joue le rôle de plus petit dénominateur commun et empêche de profiter des fonctionnalités avancées de chaque base cible (figure 16-1).

Pour complexifier les interfaces génériques, chaque fournisseur propose un pilote ODBC pour sa base. Ainsi, il existe plusieurs pilotes ODBC proposés par Oracle, suivant la version Oracle cible. Mais d'autres fournisseurs proposent également leurs pilotes : par exemple, Microsoft propose des pilotes pour Oracle. On imagine facilement les difficultés engendrées : Microsoft (fournisseur du système d'exploitation et de SQL Server, concurrent d'Oracle) a-t-il intérêt à fournir un pilote ODBC pour Oracle stable, très performant et tirant parti des dernières nouveautés d'Oracle ? Nous vous laissons juge de la réponse !

Après ODBC, Microsoft a introduit la technologie Object Linking and Embedding (OLE) avec l'interface générique OLE DB (*DataBase*) pour l'accès aux bases de données. D'autres versions des outils de développement ont utilisé cette nouvelle interface (figure 16-2).

Le pilote générique OLE DB ressemble à ODBC dans son principe. Il utilise des liens OLE et des interfaces Component Object Model (COM) pour communiquer (figure 16-3).

Pour répondre à certains besoins, des « briques logicielles » directement intégrables dans les applications ont été développées, souvent par des sociétés tierces : les ActiveX Data Objects (ADO). Leur champ d'application est vaste. Elles permettent, par exemple, d'intégrer dans une application un composant capable de visualiser graphiquement des données. Suivant l'accès utilisé, le composant peut utiliser l'interface ODBC ou OLE DB.

**Figure 16-1***Exemple d'un pilote générique ODBC***Figure 16-2***Accès générique OLE DB*

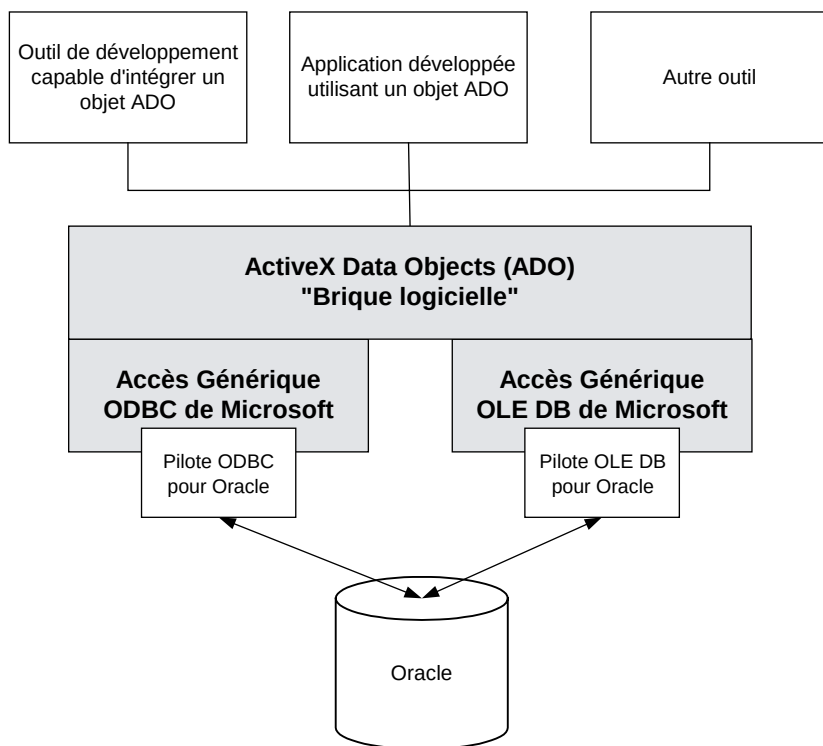


Figure 16-3

Exemple d'un ActiveX Data Objects (ADO)

Il est rare qu'un composant destiné à être commercialisé le plus largement possible utilise une interface native Oracle comme Oracle Objects for OLE (OO4O).

L'utilisation d'un ActiveX Data Objects (ADO) impose la présence d'un pilote générique pour accéder à Oracle.

Quels sont les outils de développement qui utilisent les accès ODBC et OLE DB ? Principalement des outils de développement généralistes comme ceux de la famille Microsoft Visual Studio, Visual Basic, Visual C++, Borland Delphi, C, C++, etc. Les outils de développement plus spécialisés, comme Sybase Powerbuilder ou ceux d'Oracle, utilisent une interface OCI dédiée à Oracle. C'est aussi le cas des outils d'infocentre comme Business Objects ou Impromptu de Cognos. Nous abordons l'accès OCI plus loin dans ce chapitre.

Certaines limites rencontrées avec les accès ODBC ont été levées avec les pilotes OLE DB, mais d'autres ne peuvent être contournées, et il faut alors utiliser des pilotes spécialement développés pour Oracle, des pilotes *natifs*.

Les pilotes natifs Oracle

Les pilotes génériques ont bien rempli leur mission, à savoir proposer un accès simple aux données. Il restait à améliorer plusieurs axes : la performance, l'accès à toutes les possibilités techniques offertes par les bases cibles, la synchronisation entre la sortie d'une nouvelle version d'Oracle et celle de ses interfaces d'accès. C'est pourquoi Oracle propose son propre pilote, Oracle Objects for OLE (OO4O).

Oracle Objects for OLE (OO4O)

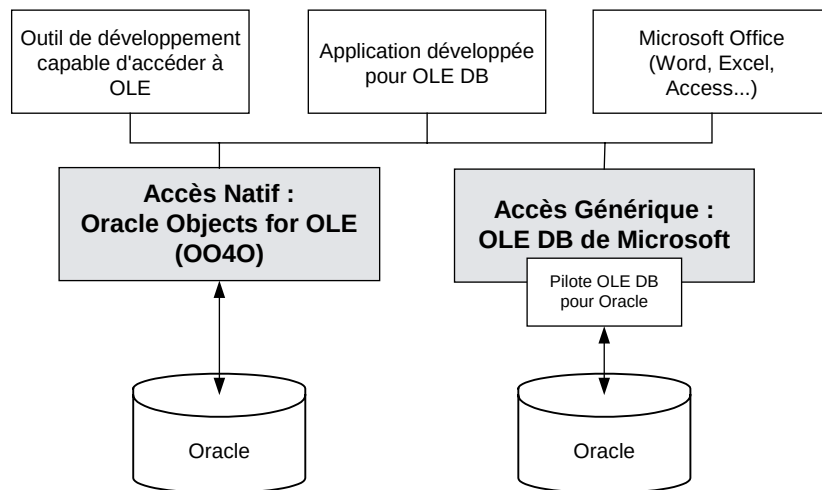


Figure 16-4

Les pilotes OLE Oracle

Oracle Objects for OLE (OO4O) peut être utilisé par de nombreux environnements, depuis des applications Web ou client-serveur. Il est accessible depuis la majorité des outils de développement ou les langages de script qui permettent l'accès COM : Visual Basic, Visual C++, Microsoft Office, Active Server Pages, Internet Information Server (IIS) et bien d'autres outils de développement.

Comme c'est un pilote Oracle natif, les performances ne sont pas limitées par l'usage des interfaces génériques ODBC ou OLE DB. Il offre l'accès à toutes les possibilités d'Oracle qui restent inaccessibles à ODBC ou OLE DB.

Les avantages d'Oracle Objects OLE

Il n'est pas nécessaire, comme pour ODBC, de réaliser un paramétrage quelconque du produit avant son utilisation : son installation suffit. Cela est particulièrement intéressant dans le cadre de grandes configurations, avec des postes utilisateur distants.

Une large partie de la syntaxe Oracle est accessible. Vous pouvez programmer en VBA (*Visual Basic Application*) à partir d'Excel ou Word ou encore à partir de Visual Basic ou en C++. L'éventail est large !

Comme l'accès via OLE se fait au travers de macros ou de développements spécifiques, la sécurité est renforcée et se trouve reportée au niveau de l'application : les utilisateurs ne peuvent effectuer que ce que leur application leur permet.

Les accès .NET

À grand renfort de marketing, Microsoft a lancé son offensive .NET pour accompagner la nouvelle version de son système d'exploitation. Tous les outils reçoivent l'estampille .NET, comme la nouvelle version de l'environnement de développement *Visual Studio .NET* qui permet d'exécuter des Web Services. Qu'en est-il pour les accès à Oracle ?

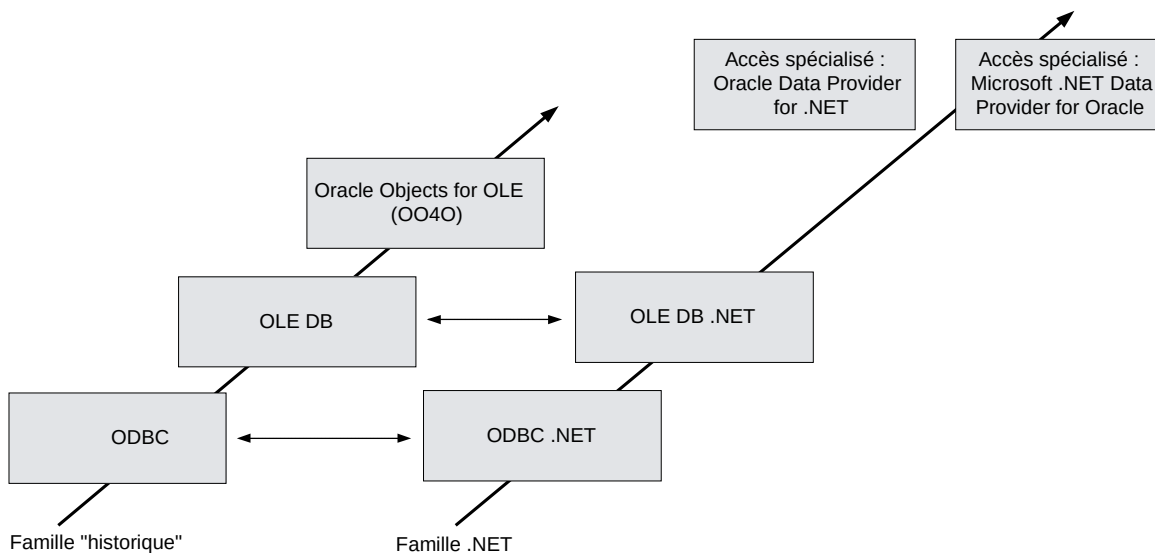


Figure 16-5

Les accès Oracle depuis .NET

Les accès ODBC .NET et OLE DB .NET sont proposés par Oracle. Ils permettent de migrer dans un environnement .NET les applications développées avec ODBC et OLE DB.

Point souvent omis par Oracle, Oracle Objects for OLE (OO4O) n'a pas de correspondant direct en environnement .NET.

Pour tirer le meilleur parti à la fois des langages de Visual Studio .NET et d'Oracle 10g, deux nouveaux pilotes natifs pour Oracle apparaissent. Le premier est proposé par Oracle : *Oracle Data Provider for .NET* et le second par Microsoft : *Microsoft .NET Data Provider for Oracle*.

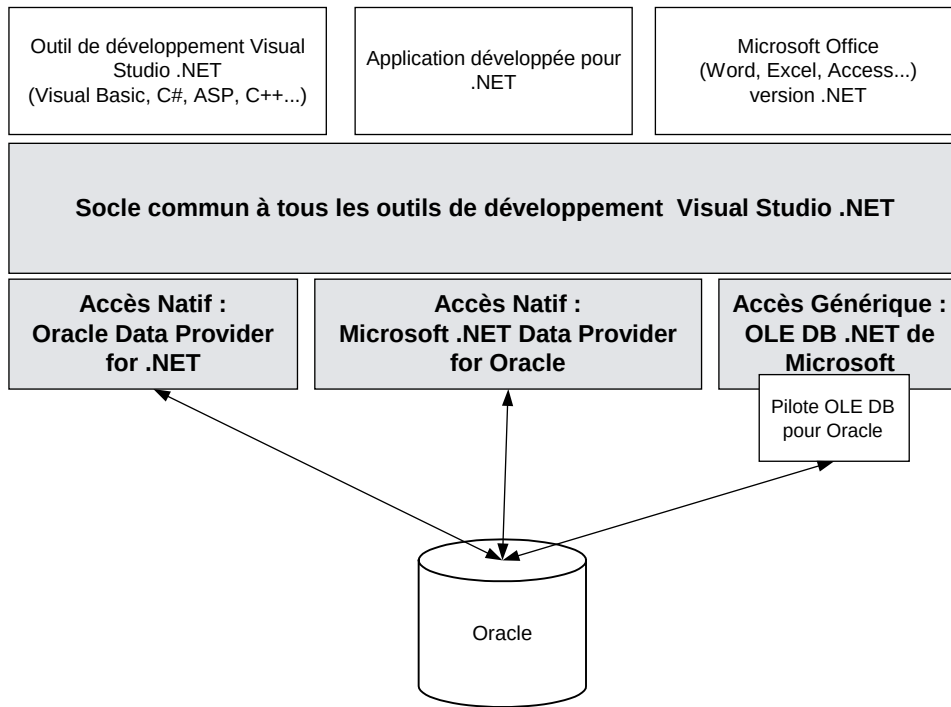


Figure 16-6

Les accès .NET

Ces deux accès n'étant disponibles que depuis quelques semaines, nous manquons de recul pour les comparer. Information intéressante, Microsoft a réalisé un benchmark publié sur son site <http://technet.microsoft.com>. Il compare les performances entre le pilote OLE DB .NET et Microsoft .NET Data Provider for Oracle. Sur les tests réalisés, les gains vont jusqu'à 200 %. Il faut se méfier de ce type d'argument, mais nous vous encourageons à consulter les programmes réalisés dans les deux environnements.

Oracle Call Interface (OCI)

Oracle Call Interface (OCI) est la couche de programmation la plus basse qui existe pour accéder à Oracle. Elle n'est accessible que depuis des langages compilés comme le C ou le C++. La bibliothèque des appels OCI est documentée, mais aucune stabilité n'est assurée lors du passage d'une version Oracle à une autre.

En d'autres termes, seules de grandes sociétés utilisent les accès OCI. Par exemple, des fournisseurs d'outils de développement voulant avoir une maîtrise complète de l'accès à Oracle développent leur interface d'accès en OCI. Cela impose pratiquement de re-développer cet accès pour chaque version Oracle.

Sont développés en utilisant les OCI :

- tous les pilotes d'accès à Oracle : ODBC, OLE DB, .NET... ;
- des interfaces d'accès à Oracle comme l'accès PHP ou Java... ;
- l'accès depuis des produits de développement ou d'interrogation comme ceux de Borland, Business Objects, Cognos, Crystal Reports, Powerbuilder... ;
- l'interface Oracle avec de grands progiciels : SAP, PeopleSoft, Baan...

Le recours aux OCI est donc à réserver à des utilisations bien ciblées. Généralement, l'utilisation des OCI a lieu lorsque la société ne veut dépendre d'aucune solution tierce pour accéder à Oracle. Elle maîtrise ainsi la totalité de la liaison technique, malgré le surcoût engendré par les développements en OCI.

L'accès à une base Oracle distante

Quel que soit le moyen d'accès choisi, vous devez utiliser Oracle Net pour communiquer entre une interface de programmation *locale* et une base Oracle *distante*. Ce point est largement évoqué dans le chapitre précédent

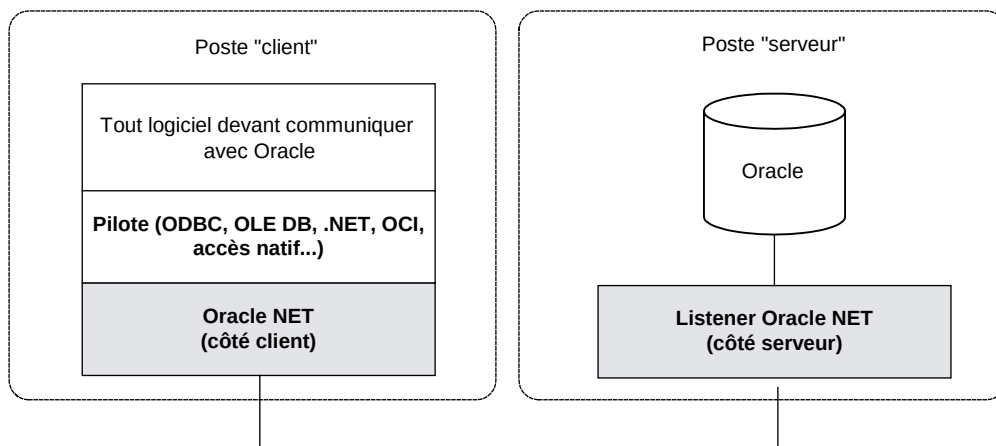


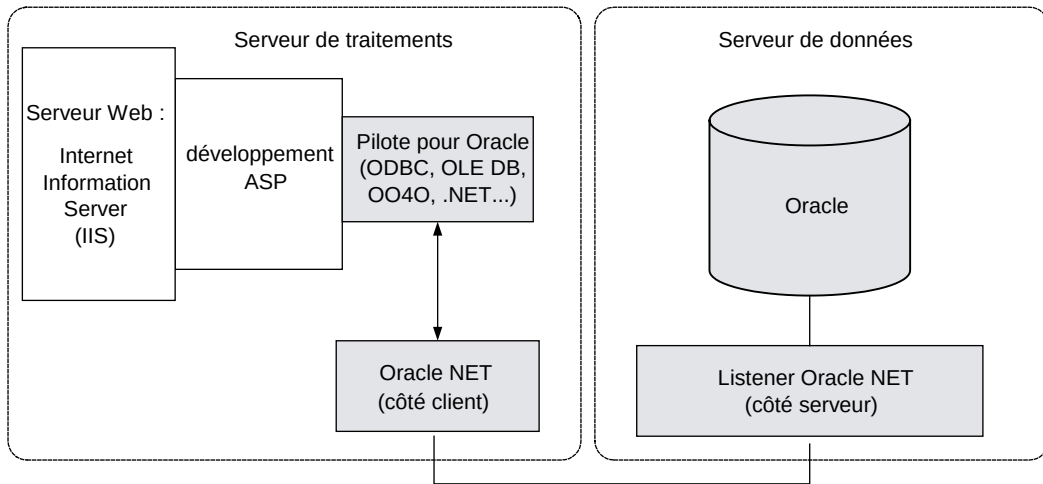
Figure 16-7

Rappel du lien client-serveur

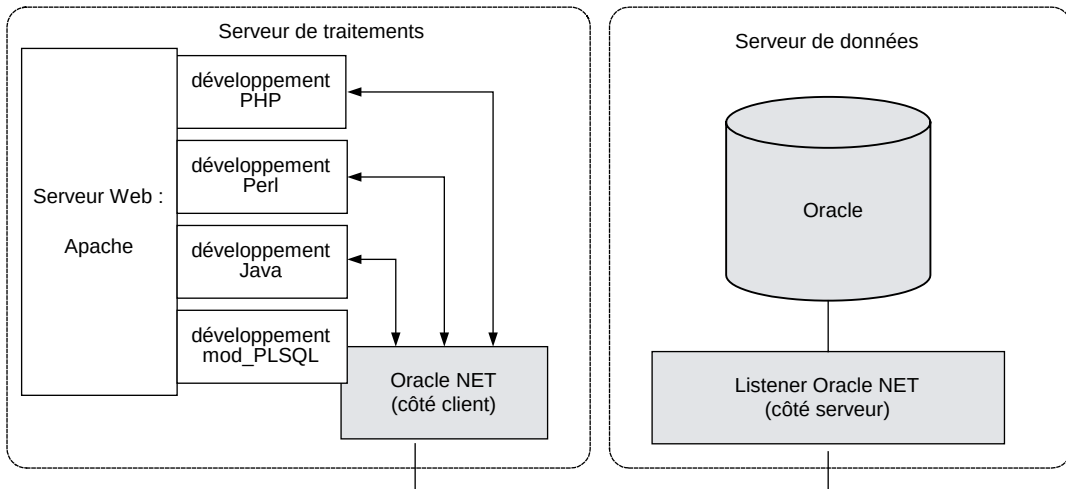
Accéder à Oracle depuis un serveur Web

Les accès à Oracle peuvent être utilisés depuis Internet Information Server (IIS) et les développements en ASP. Le serveur de traitement et le serveur de données peuvent être séparés, comme illustré à la figure 16-8.

Le serveur Web Apache est le plus utilisé dans le monde. Issu de développements en Open Source, il existe même sous Windows. Les interfaces permettant d'accéder à Oracle sont nombreuses : PHP, Perl, Java, etc.

**Figure 16-8**

Accès depuis le serveur Web de Microsoft

**Figure 16-9**

Accès à Oracle depuis Apache

Remarquez que toutes les interfaces créées par Oracle pour son serveur d'application Internet Application Server (iAS) utilisent Apache ! Les modules Apache n'utilisent pas les accès Microsoft (ODBC, OLE, etc.) mais des accès natifs à Oracle programmés en OCI.

Les développements Java peuvent utiliser un client Oracle Net local, un pilote JDBC local ou télécharger dynamiquement un pilote Thin JDBC à chaque démarrage de l'application.

Oracle et Microsoft Transaction Server (MTS)

Microsoft Transaction Server est un composant Component Object Model (COM) destiné à coordonner des transactions entre un serveur d'application et une ou plusieurs bases de données.

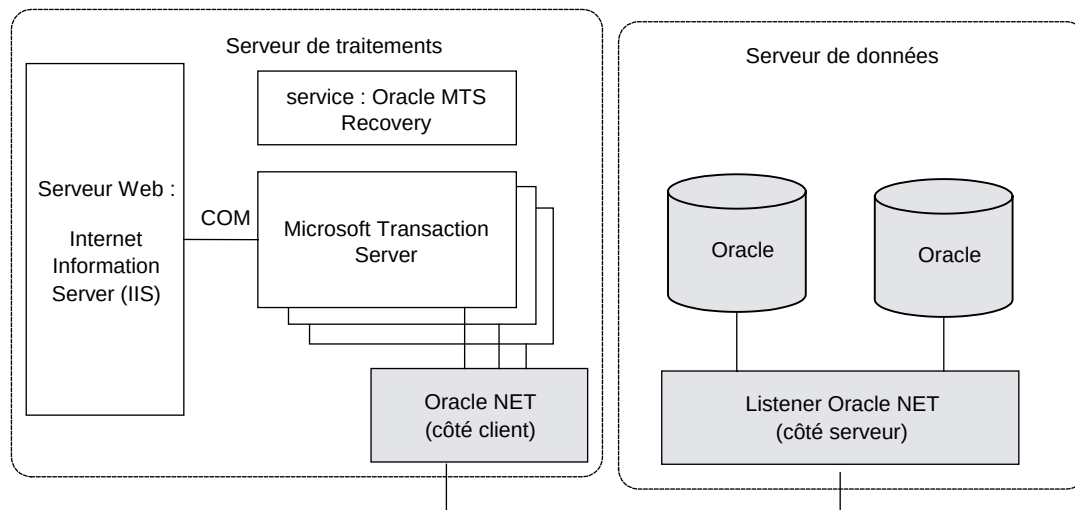


Figure 16-10

Principe de Microsoft Transaction Server (MTS)

Microsoft Transaction Server (MTS) permet de coordonner des transactions qui affectent plusieurs serveurs et plusieurs bases de données, qu'elles soient situées sur un ou plusieurs serveurs distants. Le service Oracle MTS Recovery est un service Windows développé par Oracle.

Quel accès choisir ?

Avec cette vaste palette, Oracle offre un large choix aux développeurs Windows. Vous devez choisir la technologie qui correspond le mieux aux caractéristiques techniques recherchées et aux performances attendues. Les développeurs seront plus productifs en utilisant une technique d'accès qu'ils maîtrisent déjà, plutôt que de devoir en apprendre une nouvelle.

Pour chaque type d'accès, des exemples sont proposés par Oracle. Pour Oracle Objects for OLE, des documents Word et Excel situés dans le répertoire `c:\oracle\oracle92\oo4o` utilisent OO4O pour accéder à Oracle. Ces exemples permettent de réaliser très rapidement des maquettes opérationnelles.

Avant de choisir un accès, identifiez clairement les contraintes liées à l'environnement et aux outils de développement :

- Quels sont les accès à Oracle proposés par l'outil de développement retenu ?
- Quelles sont les compétences de vos équipes de développement ? Connaissent-elles déjà un moyen d'accès ?
- Quelles sont les limites liées aux versions des logiciels (Oracle, Windows, Office...) ?
- Si vous utilisez des ADO, quel est l'accès imposé ?

Ensuite, pour accéder à Oracle, définissez les fonctionnalités avancées dont vous avez besoin :

- performance, stabilité, facilité d'installation, maîtrise des versions ;
- quel est l'impact des changements de version Oracle sur l'accès ;
- support Unicode ;
- accès à des données binaires (BLOB, CLOB, etc.) ;
- support du PL/SQL et des curseurs ;
- multiplexage des sessions dans le cas d'un serveur d'application Web, etc.

Pour avoir un retour sur l'utilisation d'un accès dans un cas de figure bien précis, utilisez les forums de discussion présents sur le Web (ceux d'otn.oracle.com ou ceux mentionnés en annexe) et n'hésitez pas à réaliser des maquettes basées sur les exemples fournis par Oracle.

Migrer des données provenant d'Access, Excel vers Oracle 10g

Oracle propose des utilitaires qui permettent de migrer des données provenant de bases de données concurrentes vers Oracle. Vous pouvez ainsi migrer des données provenant d'Excel, Access, SQL Server, DB2, Informix, etc. à l'aide d'un kit technique gratuit adapté à chaque base : le *Migration Workbench*.

Il est nécessaire de s'enregistrer sur le site <http://otn.oracle.com> pour accéder aux kits ou *Migration Workbench*.

Oracle COM Automation

Dans ce chapitre, tous les accès présentés sont à l'initiative d'un programme appelant Oracle. Or, il est possible d'appeler des logiciels **depuis la base de données**. Ainsi, une procédure PL/SQL ou Java peut appeler et manipuler des objets COM. Cela suppose bien sûr que la base de données fonctionne dans un environnement Microsoft.

Oracle COM Automation étant une technologie Microsoft, vous ne la retrouvez pas sous Unix ou Linux, cependant, Oracle permet d'appeler des procédures externes depuis des bases sous Unix ou Linux, ce qui s'approche très fortement d'un appel à COM Automation.

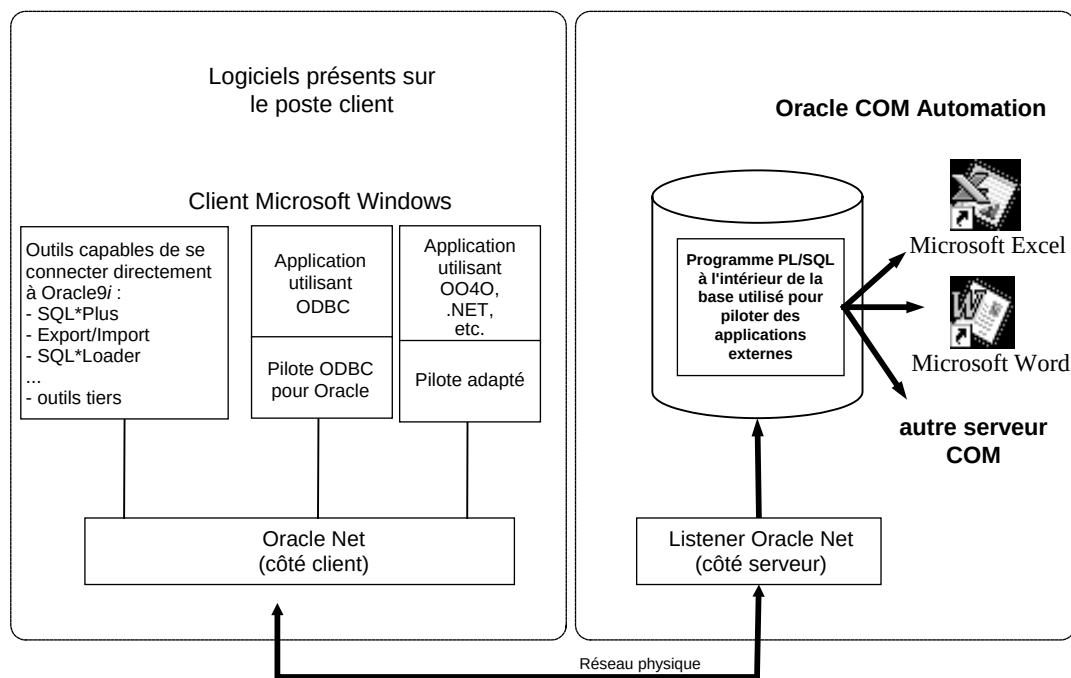


Figure 16-11

Principe d'Oracle COM Automation

Depuis la base de données, vous pouvez appeler des serveurs COM Automation. Des développements complémentaires permettent la réalisation d'applications telles que :

- transmettre des données à Microsoft Word, Excel, Powerpoint et utiliser toutes les possibilités de programmation de ces outils ;
- générer des éditions à partir de Crystal Reports (Seagate) ;
- envoyer et recevoir des e-mails depuis des applications compatibles MAPI ;
- accéder à tout autre serveur COM, proposé au sein d'un logiciel ou développé pour vos propres besoins.

Oracle COM Automation pour PL/SQL ou Java propose une interface de programmation qui permet d'instancier des objets COM. Ainsi, les développeurs peuvent appeler ces interfaces COM depuis des procédures stockées, des fonctions ou des triggers.

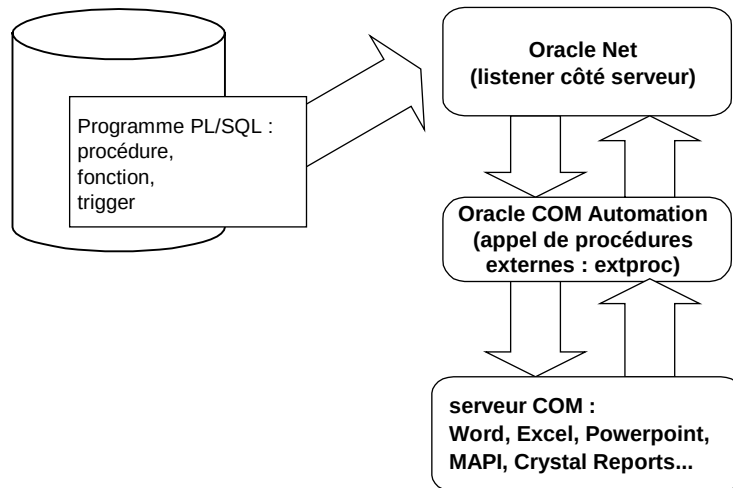
Il n'y a pas de restriction quant à l'emplacement de l'objet COM. Il peut être sur le même serveur que la base de données ou situé sur un serveur distant. Dans ce cas, l'appel est effectué via le Distributed Component Object Model (DCOM).

Des exemples de programmes PL/SQL qui appellent Microsoft Word, Excel, etc., sont situés en c:\oracle\ora92\com.

Comment s'effectue l'appel au serveur COM depuis la base Oracle ? L'architecture retenue par Oracle isole la base de données de l'environnement COM, de façon à ce qu'elle ne soit pas affectée par une erreur provoquée par un objet COM. Pour cela, Oracle COM Automation s'exécute à l'extérieur du processus oracle.exe.

Figure 16-12

Rôle du Listener dans COM Automation



Le listener sert de relais entre la base et Oracle COM Automation : il doit être configuré pour cela. L'exemple suivant présente la configuration des fichiers listener.ora et tnsnames.ora dans le cas d'utilisation de COM Automation :

```

Fichier listener.ora
LISTENER =
(ADDRESS_LIST =(ADDRESS= (PROTOCOL= IPC) (KEY= EXTPROC0)
)
)
STARTUP_WAIT_TIME_LISTENER = 0
CONNECT_TIMEOUT_LISTENER = 10
TRACE_LEVEL_LISTENER = OFF
SID_LIST_LISTENER =
(SID_LIST = (SID_DESC = (SID_NAME = ORCL)
)
(SID_DESC = (SID_NAME = extproc) (PROGRAM=extproc)
)
)
PASSWORDS_LISTENER = (oracle)
Fichier tnsnames.ora

extproc_connection_data.world = (DESCRIPTION = (ADDRESS =(PROTOCOL=IPC)
➡(KEY=EXTPROC0)
)
(CONNECT_DATA = (SID=extproc)
)
)
  
```

Pour configurer ces fichiers, nous vous conseillons d'utiliser Oracle Net Manager, présenté au chapitre précédent.

Oracle COM Automation ouvre de très nombreuses possibilités de programmation et d'automatisation en environnement Windows.

Résumé

Les possibilités d'accès à Oracle depuis Windows sont multiples. Au fil des ans, de plus en plus de techniques sont venues s'ajouter les unes aux autres. Ce chapitre a présenté l'ensemble des techniques d'accès proposées par Oracle et Microsoft pour faire communiquer des applications développées dans un environnement Windows avec une base Oracle locale ou distante.

Partie IV

Outils et langages de programmation

Chapitre 17 : Programmer avec PL/SQL

Chapitre 18 : Les outils d'Oracle 10g

Chapitre 19 : Pilotes JDBC et SQLJ

Chapitre 20 : Java et Oracle 10g

Chapitre 21 : XML et Oracle 10g

Chapitre 22 : Présentation d'Application Express

Programmer avec PL/SQL

Dans ce chapitre :

- les objectifs du PL/SQL ;
- la syntaxe du PL/SQL ;
- les procédures et les fonctions ;
- les fonctions standards ;
- les triggers.

PL/SQL est le langage procédural de troisième génération (L3G), extension du SQL, finement intégré au serveur Oracle 10g. Il permet de manipuler toutes les données Oracle 10g : relationnelles, relationnelles-objet, Java. C'est un langage de programmation à la fois puissant, simple et moderne destiné à exprimer facilement vos règles de gestion complexes sous forme de procédures et de triggers stockés.

Ce chapitre vous propose une prise en main progressive du PL/SQL, adaptée aux besoins des développeurs. De nombreux exemples sont présentés, les différentes techniques exposées et des règles de bonne programmation établies.

Nous évoquerons quelques procédures standard d'Oracle, les packages, dont il est utile de connaître l'existence car ils peuvent rendre de nombreux services.

Les avantages de PL/SQL

PL/SQL, langage de programmation éprouvé, offre de nombreux avantages :

- intégration parfaite du SQL ;

- support de la programmation orientée objet ;
- très bonnes performances ;
- portabilité ;
- facilité de programmation ;
- parfaite intégration à Oracle 10g et à Java.

Intégration parfaite du SQL

SQL est devenu le langage d'accès aux bases de données parce qu'il est standard, puissant et simple d'apprentissage. De très nombreux outils en ont popularisé l'utilisation. PL/SQL permet de réaliser des traitements complexes sur les données contenues dans une base Oracle 10g d'une façon simple, performante et sécurisée.

Support de la programmation orientée objet

Les types objet proposent une approche aisée et puissante de la programmation objet. En encapsulant les données avec les traitements, ils offrent au PL/SQL une programmation qui s'appuie sur des méthodes. Dans la programmation objet, l'implémentation des méthodes est indépendante de leur appel, ce qui constitue un avantage. On peut ainsi modifier des méthodes sans affecter l'application cliente. Ces points sont détaillés au chapitre 20, *Java et Oracle 10g*.

Très bonnes performances

Ce ne sont plus des ordres SQL qui sont transmis un à un au moteur de base de données Oracle 10g, mais un bloc de programmation. Le traitement des données est donc interne à la base, ce qui réduit considérablement le trafic entre celle-ci et l'application. Combiné à l'optimisation du moteur PL/SQL, cela diminue les échanges réseau et augmente les performances globales de vos applications. Le moteur PL/SQL d'Oracle 10g a été optimisé et les performances de traitement sont accrues.

Portabilité

Toutes les bases de données Oracle 10g comportent un moteur d'exécution PL/SQL. Comme Oracle 10g est présent sur un très grand nombre de plates-formes matérielles, le PL/SQL permet une grande portabilité de vos applications.

Facilité de programmation

PL/SQL est un langage simple d'apprentissage et de mise en œuvre. Sa syntaxe claire offre une grande lisibilité en phase de maintenance de vos applications. De nombreux outils de développement, autres que ceux d'Oracle 10g, autorisent la programmation en PL/SQL dans la base de données.

Parfaite intégration à Oracle 10g et à Java

PL/SQL, le langage L3G évolué, est étroitement intégré au serveur Oracle 10g. Il est également possible d'utiliser Java à la place de PL/SQL. Partout où PL/SQL peut être employé, Java peut l'être aussi. Java permet d'aller au-delà des possibilités de PL/SQL.

Oracle Application Express utilise massivement le PL/SQL

Oracle Application Express est une solution de développement d'applications Web à contenu dynamique. L'interface de développement et l'application utilisent des pages Web écrites en langage HTML. Oracle Application Express cible les applications Internet à développement rapide et à déploiement simplifié. Dans son fonctionnement interne, Application Express utilise massivement le langage PL/SQL. Il est décrit au chapitre 22, *Oracle Application Express*.

Survol de PL/SQL

Une manière simple d'aborder le PL/SQL est d'observer quelques-unes des utilisations qui en sont faites. Dans l'exemple 1, un programme PL/SQL consulte le salaire de l'employé SCOTT et en conserve la valeur dans une variable. Si cette variable est inférieure à 1 000, SCOTT bénéficie d'une augmentation de 10 %. Si cette variable est supérieure à 1 000, on ajoute 100 à la commission dont bénéficie SCOTT.

```
-- PL/SQL Exemple 1
DECLARE
    v_salaireNUMBER(7);-- variable locale
BEGIN
    SELECT sal INTO v_salaire FROM emp
        WHERE ename = 'SCOTT'
        FOR UPDATE OF sal;
    IF v_salaire < 1000 THEN -- test sur le salaire
        UPDATE emp SET sal = sal*1,1
            WHERE ename = 'SCOTT'; -- augmentation de 10%
    ELSE
        UPDATE emp SET comm = comm + 100
            WHERE ename = 'SCOTT' ;
    END IF;
    COMMIT;
END;
```

Ici, le langage SQL permet de rechercher et de mettre à jour des données tout en contrôlant les modifications apportées suivant les conditions exprimées par le PL/SQL. Le langage SQL est « ensembliste », c'est-à-dire qu'il ne manipule qu'un ensemble de données satisfaisant des critères de recherche. PL/SQL est « procédural » il permet de traiter de manière conditionnelle les données retournées par un ordre SQL. Vous ne pourriez pas résoudre aussi simplement, en SQL « pur », un traitement tel que celui de l'exemple 1.

Une structure de « bloc »

Le PL/SQL n'interprète pas une commande, mais un ensemble de commandes contenues dans un « bloc » PL/SQL. Ce bloc peut comporter plusieurs « sous-blocs ». Sa structure est simple d'apprentissage et lisible en termes de maintenance de programmes.

Les sections indispensables d'un bloc sont le BEGIN et le END. Elles peuvent être précédées d'une zone optionnelle de déclaration de variables, DECLARE, puis d'une zone optionnelle de traitement des erreurs, EXCEPTION.

```
DECLARE
    -- zone de déclaration de variables
BEGIN
    -- traitements ...
EXCEPTION
    -- traitement des erreurs rencontrées
END;
```

Les variables PL/SQL

PL/SQL permet de déclarer des variables et des constantes, puis de les utiliser dans des ordres SQL ou des traitements. Seul impératif président à l'utilisation d'une variable : elle doit d'abord être déclarée. Cette déclaration s'effectue dans la zone DECLARE.

```
DECLARE
    v_texte VARCHAR(30);
BEGIN
    ...
    v_texte := 'Texte que je manipule.';
    ...
END;
```

Les structures de contrôle

C'est l'une des plus importantes caractéristiques de PL/SQL. Ces structures permettent de contrôler l'exécution des ordres SQL et de traiter de façon procédurale les données. Les principales structures sont les suivantes :

- IF-THEN-ELSE ;
- FOR-LOOP ;
- WHILE-LOOP ;
- EXIT-WHEN ;
- GOTO (peu utilisé).

Les syntaxes des différents ordres sont développées dans ce chapitre.

Les curseurs

Oracle 10g utilise des espaces de travail pour exécuter les ordres SQL et en manipuler les données. Une instruction PL/SQL, le curseur, permet d'attribuer des noms à ces espaces de travail et d'accéder aux données contenues. Il existe deux sortes de curseurs : les curseurs *implicites* et *explicites*. Le PL/SQL crée de manière implicite un curseur pour chaque ordre SQL, même pour ceux qui ne renvoient qu'une ligne. Pour les curseurs qui renvoient plus d'un enregistrement, vous pouvez déclarer explicitement un curseur, afin de manipuler individuellement les lignes retournées. Les curseurs sont abordés dans le détail dans ce chapitre.

La modularité

PL/SQL permet d'écrire des programmes modulaires, faciles à réaliser et à maintenir. On peut scinder un problème complexe en un ensemble de sous-programmes simples, grâce aux procédures et aux fonctions. Par-delà les blocs et leurs sous-programmes, PL/SQL propose les *PACKAGES*, qui permettent de regrouper un ensemble de programmes liés sous une même appellation, avec une partie « privée » (accessible uniquement par les procédures et fonctions internes au package) et une partie « publique » (accessible par les procédures et fonctions externes au package).

Les attributs

Cette possibilité est fort utile. Elle fait référence aux caractéristiques d'une colonne de table (par exemple) et d'en récupérer automatiquement le type. Si le schéma de la base de données se trouve modifié (zone textuelle augmentée, zone numérique incorporant plus de décimales...), la variable en PL/SQL hérite d'emblée de ces modifications. Ce type a aussi le mérite de simplifier l'écriture de vos programmes. Par exemple, pour déclarer une variable `v_nom` destinée à contenir le nom d'un employé de la table *emp*, on utilise le type `%TYPE` :

```
■ v_nom emp.ename%TYPE ;
```

Pour déclarer un enregistrement, il faut avoir recours au type `%ROWTYPE`. Dans ce cas, comme l'on récupère dans un enregistrement l'ensemble des colonnes définies dans le curseur, il n'y a pas de référence à une colonne précise dans la déclaration : il convient donc de préciser l'identifiant de l'enregistrement ainsi que celui de la colonne considérée. Le nom de la colonne est identique à celui de l'ordre SQL

```
■ rec_employe emp.%ROWTYPE ;  
...  
v_nom_employe := rec_employe.ename ;
```

L'ouverture

PL/SQL permet d'utiliser des procédures externes (*EXTERNAL PROCEDURE*), c'est-à-dire un programme écrit en L3G, généralement le langage C. Les applications cibles sont très spécifiques et se rapprochent souvent des applications temps réel, scientifiques

ou de traitement d'images. L'appel de cette procédure externe peut s'effectuer à partir d'un bloc PL/SQL. C'est une possibilité très puissante : vous avez la possibilité de déclencher depuis la base de données des traitements extérieurs à la base.

Un accès aux données relationnelles-objet

Oracle 10g est une base relationnelle qui offre des possibilités de stockage d'objets. À ce titre, PL/SQL propose des syntaxes permettant de manipuler des objets.

Le traitement des erreurs

Avec PL/SQL, il est facile d'identifier et de traiter les erreurs qui peuvent survenir dans vos programmes, car elles déclenchent des exceptions. L'erreur est alors transmise à la partie EXCEPTION de votre bloc PL/SQL pour y être traitée.

La plupart des exceptions sont définies en standard : par exemple, l'exception NO_DATA_FOUND survient lorsqu'un ordre SQL ne retourne aucune donnée. Outre celles-ci, vous pouvez ajouter vos propres exceptions, ce qui confère une grande souplesse d'utilisation.

```
DECLARE
    ...
    excep_commissionEXCEPTION ;
    ...
DECLARE
    ...
    IF commssion > salaire THEN
        RAISEexcep_commission ; -- branche sur l'exception
    ELSE
        ...
    END IF ;
    ...
EXCEPTION
    WHEN excep_commission THEN -- traitement de l'exception
        BEGIN
            ... traitement
        END;
    WHEN NO_DATA_FOUND THEN-- branchement automatique
        ... traitement
END;
```

Une fois l'exception traitée, le bloc PL/SQL se termine.

Les exemples de programmes fournis par Oracle

PL/SQL est livré avec un ensemble d'exemples. En complément de ce chapitre, nous vous encourageons vivement à les consulter. Suivez les étapes de mise en place de cet environnement :

1. Lancez SQL*Plus et connectez-vous sous l'utilisateur SCOTT/TIGER :

```
C:\> cd C:\Oracle\ora92\plsql\demo
C:\> sqlplus scott/tiger
```

2. Pour charger l'environnement de démonstration, lancez le script SQL suivant :

```
SQL> @examp1d
```

3. Les exemples de programmes PL/SQL se trouvent dans le répertoire suivant :

```
C:\Oracle\ora92\plsql\demo
C:\> dir *.sql: programmes PL/SQL
C:\> dir *.pc : programmes en langage C comportant du PL/SQL
```

4. Connectez-vous sous SQL*Plus, puis lancez les programmes SQL :

```
C:\> sqlplus scott/tiger
SQL> start examp1
```

5. Compilez les programmes C avant de les lancer :

```
C:\> cd C:\Oracle\ora92\plsql\demo
...Compilation (nécessite la présence d'un compilateur C)
C:\> make -f demo_plsql.mk demos
...Exécution
C:\> examp9
```

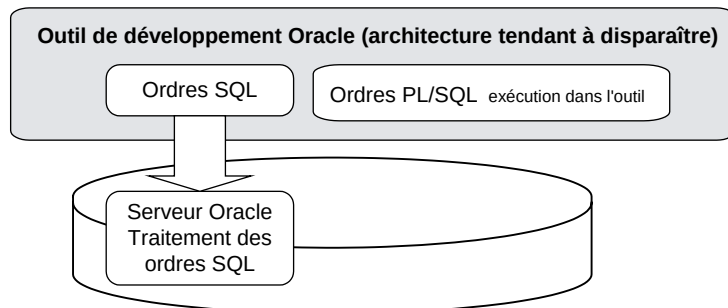
Architecture de PL/SQL

Le PL/SQL est un langage de programmation destiné à être exécuté. Ce rôle revient au « moteur » PL/SQL qui n'est pas un produit séparé. Il est embarqué dans le cœur de la base de données ou dans certains outils de développement (exclusivement ceux d'Oracle).

Un bloc PL/SQL peut être traité dans un outil de développement Oracle (SQL*Plus, Oracle Forms, Oracle Reports). Dans ce cas, seules les instructions conditionnelles sont traitées par le moteur PL/SQL embarqué dans l'outil de développement. Les ordres SQL incorporés dans les blocs PL/SQL sont toujours traités par la base de données.

Figure 17-1

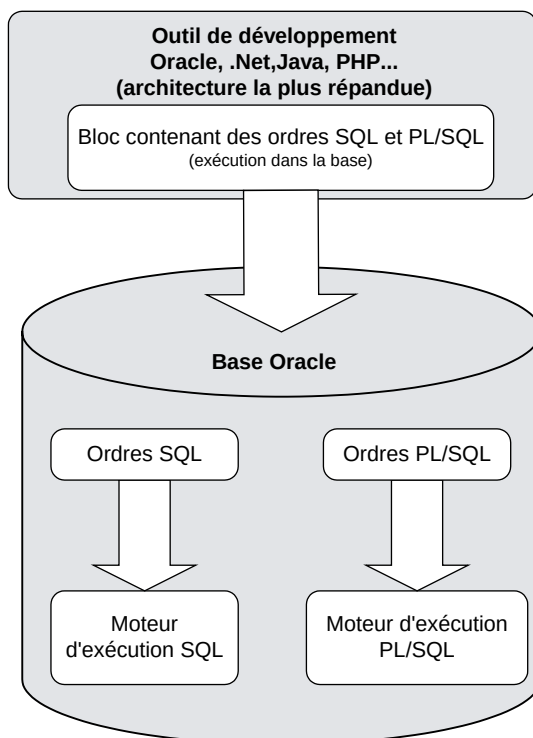
*Traitement d'un ordre
PL/SQL dans un outil
de développement Oracle*



Pour les blocs PL/SQL intégralement transmis à la base de données, la partie relative au traitement des instructions conditionnelles est effectuée par le moteur PL/SQL de la base ; le traitement des ordres SQL incombe au moteur d'exécution de la base.

Figure 17-2

*Traitement d'un ordre
PL/SQL dans la base
de données*



Développer avec PL/SQL

Avant d'aborder dans le détail la programmation des blocs PL/SQL, étudions leur fonctionnement. En effet, pour bien mesurer l'apport de PL/SQL, il est important de comprendre comment il s'insère dans une relation client-serveur.

Fonctionnement de PL/SQL

Le SQL et le PL/SQL comportent chacun un « moteur d'exécution » associé, respectivement le SQL STATEMENT EXECUTOR et le PROCEDURAL STATEMENT EXECUTOR. Le premier moteur se situe toujours dans la base de données ; quant au second, il peut se trouver avec le noyau Oracle 10g ou avec l'outil de développement, uniquement pour les outils appartenant à la famille Oracle Developer.

Cela ne signifie pas que le langage PL/SQL doive être exclusivement réservé aux outils de développement Oracle. Vous pouvez créer du code PL/SQL à partir de l'outil SQL*Plus

(ou d'autres outils du marché) pour réaliser des procédures, fonctions, triggers, packages stockés dans votre base Oracle 10g, et exécuter ces codes PL/SQL à partir de programmes réalisés avec presque tous les outils présents sur le marché.

La différence de fonctionnement entre le SQL STATEMENT EXECUTOR et le PROCEDURAL STATEMENT EXECUTOR tient à ce que le moteur SQL interprète les ordres SQL un par un, alors que le moteur PL/SQL interprète des blocs de commandes.

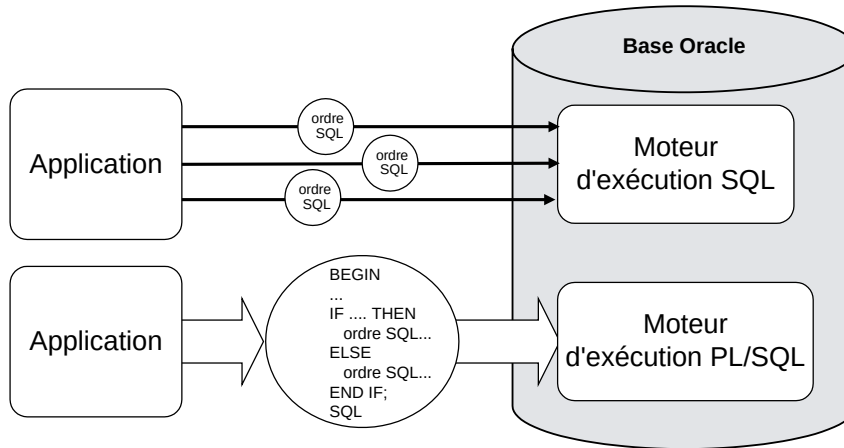


Figure 17-3

Traitement des ordres SQL et PL/SQL.

Si le moteur PL/SQL est incorporé aux outils Developer (Oracle Forms, Oracle Report), seule la partie procédurale du bloc PL/SQL est exécutée dans l'outil. Les ordres SQL sont transmis à la base de données.

Si le bloc PL/SQL est directement transmis à la base de données, le traitement procédural ainsi que les ordres SQL sont traités à l'intérieur de la base.

Une structure de « bloc »

Le PL/SQL n'interprète pas une commande, mais un ensemble de commandes contenues dans un « bloc » PL/SQL. Ce dernier peut contenir plusieurs « sous-blocs ». Les déclarations définies dans un bloc prennent fin en même temps que le bloc.

```

DECLARE
    -- zone de déclaration de variables, constantes,
    -- exceptions, curseurs
BEGIN
    -- traitements ...
    -- instructions SQL et PL/SQL
EXCEPTION
    -- traitement des erreurs rencontrées
END;
```

Vous pouvez placer des « sous-blocs » uniquement dans les sections BEGIN et EXCEPTION, pas dans la partie DECLARE.

```
DECLARE
    -- zone de déclaration de variables
BEGIN    -- bloc A
    -- traitements ...
    BEGIN -- bloc B
        -- traitements bloc B
    END;  -- fin du bloc B
    -- autres traitements du bloc A
EXCEPTION
    -- traitement des erreurs rencontrées
    BEGIN -- bloc C
        -- traitements bloc C
    END;  -- fin du bloc C
END;
```

Pour plus de lisibilité dans vos programmes, vous pouvez nommer les blocs afin d'en distinguer les différentes clauses BEGIN et END.

```
DECLARE
    -- zone de déclaration de variables
BEGIN BLOC_A -- bloc A
    -- traitements ...
    BEGIN BLOC_B -- bloc B
        -- traitements bloc B
    END BLOC_B; -- fin du bloc B
    -- autres traitements du bloc A
EXCEPTION
    -- traitement des erreurs rencontrées
    BEGIN BLOC_C -- bloc C
        -- traitements bloc C
    END BLOC_C; -- fin du bloc C
END BLOC_A; -- fin du bloc A
```

Remarques

- Les zones DECLARE et EXCEPTION sont facultatives.
- Chaque instruction, de n'importe quelle section, se termine par un « ; ».
- Il est possible d'insérer des commentaires :

```
-- commentaire sur une ligne
```

ou

```
/* commentaire sur
plusieurs lignes
*/
```

Exécution d'un programme PL/SQL avec SQL*Plus et iSQL*Plus

Pour exécuter un programme PL/SQL, l'outil le plus simple livré par Oracle est SQL*Plus. Il est abordé au chapitre 18, *Les outils d'Oracle 10g* car il permet d'accéder à l'ensemble des fonctionnalités d'Oracle 10g à partir d'une interface textuelle identique, quel que soit le système d'exploitation utilisé.

Pour exécuter un bloc PL/SQL à partir de SQL*Plus, il convient de saisir l'intégralité du bloc, puis d'en lancer l'exécution par le signe « / », suivi d'un retour chariot (appui sur la touche « entrée »).

```
DECLARE
    ...
BEGIN
    -- traitements ...

EXCEPTION
    -- traitement des erreurs rencontrées
END;
/
```

Par exemple, sous SQL*Plus :

```
SQL> declare
2   v_test number;
3   begin
4   v_test := 10;
5   end;
6   /
```

PL/SQL procedure successfully completed.

Remarque : observez la numérotation des lignes sous SQL*Plus. Seul le signe « / » permet de quitter le mode « saisie » de l'ordre SQL ou du bloc PL/SQL, pour en lancer l'exécution.

Les variables PL/SQL

PL/SQL permet de déclarer des variables et des constantes, puis de les utiliser dans des ordres SQL ou des traitements. Seul impératif président à l'utilisation d'une variable : elle doit d'abord être déclarée dans la zone DECLARE.

Pour simplifier le code, nous vous conseillons de toujours préfixer vos variables par une lettre, ce qui permet de différencier les variables des colonnes des tables. Dans les exemples, toutes les variables seront dotées du préfixe « v_ » et les colonnes « c_ ».

La déclaration de variables PL/SQL

Les variables PL/SQL doivent appartenir à un type précis, qui correspond à ceux du langage SQL. Par exemple, une variable `v_texte` destinée à accueillir une chaîne de 30 caractères et une

autre variable `v_vrai_faux` destinée à accueillir une valeur booléenne (TRUE, FALSE ou NULL) se déclareront comme suit :

```
v_texteVARCHAR2(30);  
v_vrai_fauxBOOLEAN;  
v_date_embaucheDATE ;
```

Toutes les déclarations de variables doivent s'effectuer dans la zone DECLARE de votre bloc PL/SQL.

Les différents types de variables en PL/SQL

Les types de variables les plus largement utilisés en PL/SQL sont les suivants :

```
v_numberNUMBER;-- longueur maxi 38  
v_numberNUMBER(7,2);  
  
v_charCHAR;-- longueur maxi 2 000  
v_charCHAR(20);  
  
v_varchar2VARCHAR2;-- longueur maxi 4 000  
v_varchar2VARCHAR2(30);
```

Si le texte d'une variable de type CHAR n'atteint pas la totalité de la taille de la variable, il est automatiquement « comblé » par des « blancs ». Au contraire, une variable VARCHAR2 ne stocke que les caractères utiles saisis dans la variable.

```
v_dateDATE;-- pas de précision  
v_date := '01-JAN-99';-- format par défaut JJ-MON-YY  
  
v_booleanBOOLEAN;-- valeurs TRUE, FALSE, NULL  
v_booleanBOOLEAN := 'TRUE';
```

Nous vous recommandons vivement de toujours utiliser VARCHAR2 pour sa simplicité et le gain de place qu'il procure dans la base de données.

D'autres types de variables moins usités existent. Nous n'en citons ici que quelques-uns et vous conseillons de vous référer à la documentation Oracle 10g pour disposer de la liste exhaustive.

```
v_long_rawLONG_RAW;-- longueur maxi 32760 bytes en PL/SQL  
v_rowidROWID;-- identifiant unique d'une colonne de table  
v_bfileBFILE;-- fichiers binaires extérieurs à la base
```

Variable faisant référence à la colonne d'une table

Cette façon de nommer les variables permet de récupérer automatiquement le type de la donnée tel qu'il est défini dans le dictionnaire de la base.

Par exemple, pour déclarer la variable `v_nom_employé` du même type que la colonne `NOM` de la table `EMP`, on utilisera la syntaxe suivante :

```
v_nom_employeemp.ename%TYPE;
```

Nous vous conseillons d'utiliser le plus souvent possible cette notation. Toute modification de la structure de la base de données (taille des colonnes de vos tables, de leur type...) sera alors sans effet sur les programmes PL/SQL.

Variables faisant référence à la ligne d'une table

Tout comme l'on peut déclarer une variable du même type qu'une colonne de table, il est possible de procéder ainsi pour un enregistrement, c'est-à-dire une ligne d'une table.

Par exemple, pour déclarer la variable `v_nom_employé` du même type que la colonne `NOM` de la table `EMP`, on utilisera la syntaxe suivante :

```
v_enregnom_de_ma_table%ROWTYPE;
```

Chaque variable de la structure de l'enregistrement comporte le même nom et le même type que la colonne associée.

Si la table `EMPLOYEE` se compose des colonnes `IDENTIFIANT`, `NOM`, `PRENOM`, on accédera aux données des différentes colonnes de l'enregistrement par la syntaxe suivante :

```
variable.nom_de_la_colonne_de_la_table

DECLARE
    v_enregEMPLOYEE%ROWTYPE;
    ...
BEGIN
    IF v_enreg.NOM = 'DURAND' THEN ...
    ...
    IF v_enreg.PRENOM = 'PAUL' THEN...
    ...
    IF v_enreg.IDENTIFIANT = '123456' THEN...
    ...
END;
```

Les types de variables définis par l'utilisateur

Vous pouvez définir vos propres types de données en PL/SQL : l'ordre `SUBTYPE` permet de les créer, même s'ils sont rarement utilisés.

```
DECLARE
    SUBTYPE mon_soustype_number IS NUMBER;
    v_salairemon_soustype_number(7,2);
    ...
```

Affecter des valeurs aux variables PL/SQL

Il existe deux façons d'affecter des valeurs à des variables. La première utilise l'opérateur d'assignation, le signe « := ». La variable doit être placée à gauche de l'opérateur et la valeur à droite. Les fonctions et opérations du langage SQL sont applicables.

```
v_texte := UPPER('texte saisi');  
v_vrai_faux := TRUE ;  
v_nombre := v_prix * v_taux_tva ;  
v_date_embauche := TO_DATE('01/01/2003','DD/MN/YYYY') ;
```

Il est aussi possible, dans une même opération, de déclarer une variable **CONSTANT** et d'y affecter une valeur immuable :

```
v_valeur_pi CONSTANT REAL := 3.14116 ;
```

La deuxième façon d'attribuer des valeurs à des variables consiste à effectuer un **SELECT** de valeurs en provenance de la base de données. La syntaxe utilisée se présente comme suit :

```
SELECT colonne1, colonne2  
INTO variable1, variable2  
FROM table  
[WHERE conditions] --le signe [] indique des clauses optionnelles
```

La clause **INTO** est obligatoire et l'ordre **SELECT** doit rapporter une seule ligne, sinon une erreur est générée. Si l'ordre SQL retourne plusieurs lignes, on utilisera obligatoirement un curseur (traité plus loin).

```
DECLARE  
    v_id_employe      employe.identifiant%TYPE := '1000';  
    v_nom_employe      employe.nom%TYPE ;  
    v_salaire_employe  employe.salaire%TYPE ;  
BEGIN  
    SELECT nom, salaire  
    INTO v_nom_employe, v_salaire_employe  
    FROM employe  
    WHERE empno := v_id_employe ;  
    ...  
END;
```

Dans l'exemple précédent, remarquez l'intérêt de suivre la règle pour nommer les variables locales. L'identification des variables par rapport aux colonnes des tables est plus aisée.

Visibilité des variables dans des blocs imbriqués

La structure par bloc des programmes PL/SQL oblige à déterminer si les variables définies dans un bloc sont visibles dans les autres blocs. L'exemple ci-dessous présente les règles à suivre :

```
DECLARE  
    v_identifiant      NUMBER(5);  
    v_valeur           NUMBER(9,2);
```

```
BEGIN
...;                | v_identifiant (NUMBER)
...;                | v_valeur (NUMBER)
  DECLARE
    v_identifiant  VARCHAR2(10);
    v_texte        VARCHAR2(30);
  BEGIN
...;                | v_identifiant (VARCHAR2)
...;                | v_valeur (NUMBER)
...;                | v_texte (VARCHAR2)
  END;
...;                | v_identifiant (NUMBER)
...;                | v_valeur (NUMBER)
END;
```

Utiliser en PL/SQL des variables définies sous SQL*Plus

Il est possible de définir des variables sous SQL*Plus par la commande ACCEPT. Cette variable peut être utilisée directement dans un bloc PL/SQL en attribuant à la variable le préfixe « & ».

```
SQL> PROMPT Saisissez votre texte :
SQL> ACCEPT v_texte
Je saisis mon texte...
SQL> declare
  2   v_test number;
  3   begin
  4     v_test := &v_texte ;
  5   end;
  6   /

PL/SQL procedure successfully completed.
```

Les structures de contrôle

C'est l'une des plus importantes caractéristiques de PL/SQL. Ces structures permettent de contrôler l'exécution des ordres SQL et de traiter de façon procédurale les données. Ces clauses trouvent toute leur utilité dans l'exécution des curseurs en permettant le traitement ligne à ligne des enregistrements retournés par un curseur. Les principales structures sont les suivantes :

- IF-THEN-ELSE ;
- FOR-LOOP ;
- WHILE-LOOP ;
- EXIT-WHEN ;
- GOTO (peu utilisé).

IF-THEN-ELSE

Cette syntaxe permet d'exécuter des instructions en fonction du résultat d'une condition. La syntaxe générale se présente comme suit :

```
BEGIN
...
IF condition1 THEN traitement1 ;
ELSIF condition2 THEN traitement2 ;
ELSE traitement3 ;
END IF;
...
END;
```

Les opérateurs utilisés dans les conditions sont les mêmes que dans les ordres SQL :

```
=, <, >, !=, >=, <=,
IS NULL, IS NOT NULL, BETWEEN,
LIKE, AND, OR, etc...
```

- Dès que l'une des conditions est avérée, il y a exécution du traitement qui suit le THEN.
- Si aucune condition n'est vérifiée, le traitement qui suit le ELSE est exécuté.
- Seules les clauses IF, THEN, END IF sont obligatoires.

```
-- PL/SQL Exemple 1 modifié
DECLARE
    v_salaireNUMBER(7);-- variable locale
BEGIN
    -- on récupère le salaire de l'employé SCOTT
    SELECT sal INTO v_salaire FROM emp
        WHERE ename = 'SCOTT'
        FOR UPDATE OF sal;
    IF v_salaire <= 1000 THEN -- test sur le salaire
        UPDATE emp SET sal = sal*1,1 -- augmentation de 10%
            WHERE ename = 'SCOTT';
    ELSIF v_salaire BETWEEN (1000 and 5000) THEN
        UPDATE emp SET sal = sal*1,05 -- augmentation de 5%
            WHERE ename = 'SCOTT';
    ELSE
        UPDATE emp SET comm = comm + 100
            WHERE ename = 'SCOTT' ;
    END IF;
    COMMIT;
END;
```

La boucle LOOP

PL/SQL permet d'effectuer des traitements répétitifs grâce à la clause LOOP, qui trouve toute son utilité dans le traitement des curseurs : les enregistrements retournés par un curseur peuvent en effet être traités ligne par ligne.

```

BEGIN
...
    LOOP [<<nom_boucle>>]
        instructions ;
    END LOOP [nom_boucle];
...
END;

```

Telle qu'elle est écrite, cette boucle s'exécute indéfiniment. C'est l'instruction EXIT qui stoppe l'exécution entre le LOOP et le END LOOP.

```
EXIT [nom_boucle] [WHEN condition]
```

Voici l'exemple qui permet de calculer les puissances de 2 inférieures à 100 :

```

DECLARE
    v_puissance_deux NUMBER := 2;
BEGIN
    LOOP
        INSERT INTO table_resultat
            VALUES (v_puissance_deux) ;
        v_puissance_deux := v_puissance_deux * v_puissance_deux;
        EXIT WHEN v_puissance_deux >= 100 ;
    END LOOP ;
END;

```

La boucle FOR-LOOP

La boucle FOR-LOOP simplifie l'écriture des boucles LOOP et garantit la prise en compte de la condition de sortie d'une boucle LOOP seule.

```

BEGIN
...
    FOR variable_indice IN valeur1 .. valeur2
    LOOP
        instructions ;
    END LOOP ;
...
END;

```

- Avec cette syntaxe, il est inutile de déclarer la variable *variable_indice*, car ce processus est automatique.
- *valeur1* et *valeur2* sont des expressions, des variables ou des constantes.
- Le pas d'incrément de *variable_indice* est de 1, compris entre *valeur1* et *valeur2*.

```

-- calcul de la table de 5
DECLARE
    v_table_cinq CONSTANT := 5;
    v_resultat NUMBER;
BEGIN
    FOR i IN 1 .. 9
    LOOP
        v_resultat := v_table_cinq * i ;
    END LOOP ;
END;

```

```

        -- stockage du résultat dans une colonne VARCHAR2
        INSERT INTO table_resultat
        VALUES (v_table_cinq||'*'||i||'='||v_resultat);
    END LOOP ;
END;
```

La boucle WHILE-LOOP

La boucle WHILE-LOOP s'exécute tant que la condition de la clause WHILE est vérifiée.

```

BEGIN
...
    WHILE condition
    LOOP
        instructions ;
    END LOOP ;
...
END;
```

Les opérateurs utilisés dans la condition sont les mêmes que dans les ordres SQL :

```

=, <, >, !=, >=, <=,
IS NULL, IS NOT NULL, BETWEEN,
LIKE, AND, OR, etc...
```

D'où une autre possibilité d'écrire l'exemple précédent :

```

-- calcul de la table de 5
DECLARE
    v_table_cinq CONSTANT := 5;
    v_resultatNUMBER;
    v_incrementNUMBER := 1;
BEGIN
    WHILE v_increment <= 9
    LOOP
        v_resultat := v_table_cinq * v_increment ;
        -- stockage du résultat dans une colonne VARCHAR2
        INSERT INTO table_resultat
        VALUES (v_table_cinq||'*'||i||'='||v_resultat);
        -- ne pas oublier l'increment
        v_increment := v_increment + 1 ;
    END LOOP ;
END;
```

Les curseurs

Oracle 10g utilise des espaces de travail pour exécuter les ordres SQL et en manipuler les données. Une instruction PL/SQL, le curseur, permet d'attribuer un nom à cet espace de travail et d'accéder aux données qu'il contient. Il existe deux sortes de curseurs : les curseurs *implicites* et les curseurs *explicites*. Le PL/SQL crée de manière implicite un curseur pour chaque ordre SQL, même pour ceux qui ne retournent qu'une ligne.

Les curseurs explicites

Pour les curseurs qui renvoient plus d'un enregistrement, vous pouvez déclarer explicitement un curseur, ce qui permet de traiter individuellement les lignes retournées.

```
-- recherche de tous les employés d'un département
DECLARE
  CURSOR mon_curseur IS
    SELECT ename, sal, job FROM emp WHERE deptno = 10 ;
  ...
```

L'ensemble des lignes renvoyées par une interrogation multiligne s'appelle un *result set* (jeu de résultats). Le nombre de colonnes, ainsi que les lignes retournées dépendent de l'ordre SQL traité. Comme le montre la figure suivante, un curseur explicite « pointe » vers une ligne courante du *result set*. Cela permet au programme de traiter ligne à ligne les données retournées par l'ordre SQL.

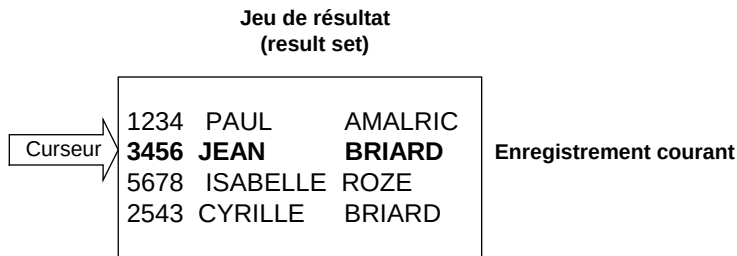


Figure 17-4

Traitement d'un ordre SQL

Le traitement d'un ordre SQL multiligne s'apparente à celui de données contenues dans un fichier. Par exemple, un programme Cobol ouvre un fichier, traite les données ligne à ligne puis ferme le fichier. De façon similaire, un programme PL/SQL ouvre un curseur, traite les enregistrements retournés par l'ordre SQL, puis ferme le curseur. De même qu'un pointeur dans un fichier identifie l'enregistrement courant, le curseur permet d'isoler l'enregistrement courant d'un *result set*.

Les étapes de la vie d'un curseur

Les instructions de déclaration d'un curseur puis OPEN, FETCH et CLOSE permettent le contrôle du début à la fin de son existence. OPEN exécute l'ordre SQL associé au curseur, identifie le *result set* et positionne le curseur **avant** le premier enregistrement. L'instruction FETCH retourne l'enregistrement courant puis place le curseur sur l'enregistrement suivant. Lorsque le dernier enregistrement est atteint, l'instruction CLOSE permet de fermer le curseur et libère les ressources allouées. Les étapes sont les suivantes :

- déclaration du curseur ;
- ouverture du curseur ;
- traitement des lignes du curseur ;
- fermeture du curseur.

La déclaration d'un curseur

Tout curseur utilisé dans un bloc PL/SQL doit obligatoirement être déclaré dans la section DECLARE du bloc, en précisant son nom et l'ordre SQL associé.

```
CURSOR nom_curseur IS ordre_select ;
```

Par exemple :

```
DECLARE
    -- déclaration de variables
    ...;
    -- déclaration de curseurs
    CURSOR c_employe IS SELECT ename, deptno, sal FROM emp
    WHERE job = 'MANAGER'
    ORDER BY ename ;
BEGIN
    ...;
    ...;
END;
```

Pour une meilleure lisibilité des programmes, nous vous conseillons de systématiquement préfixer attribuer aux curseurs le préfixe « c_ ». C'est la notation adoptée dans nos exemples.

Ouverture d'un curseur

Dès que vous ouvrez le curseur, l'exécution de l'ordre SQL est lancée. Cette phase d'ouverture s'effectue dans la section BEGIN du bloc.

```
OPEN nom_curseur ;
```

Par exemple :

```
DECLARE
    -- déclaration de variables
    ...;
    -- déclaration de curseurs
    CURSOR c_employe IS SELECT ename, deptno, sal FROM emp
    WHERE job = 'MANAGER'
    ORDER BY ename ;
BEGIN
    ...;
    OPEN c_employe ;
    ...;
END;
```


Traitement des lignes d'un curseur

L'ordre OPEN a forcé l'exécution de l'ordre SQL associé au curseur. Il faut maintenant récupérer les lignes de l'ordre SELECT et les traiter une par une, en stockant la valeur de chaque colonne de l'ordre SQL dans une variable réceptrice.

```
■ FETCH nom_curseur INTO liste_de_variables;
```

La commande FETCH ne retourne qu'un enregistrement. Pour récupérer l'ensemble des enregistrements de l'ordre SQL, il faut prévoir une boucle.

Par exemple :

```
DECLARE
    -- déclaration de variables
    v_nom          emp.ename%TYPE;
    v_departement  emp.job%TYPE;
    v_salaire      emp.sal%TYPE;

    -- déclaration de curseurs
    CURSOR c_employe IS SELECT ename, deptno, sal FROM emp
    WHERE job = 'MANAGER'
    ORDER BY ename ;
BEGIN
    OPEN c_employe ;
    LOOP
        FETCH c_employe INTO
            v_nom,v_fonction, v_salaire ;
        IF (v_fonction = 'CLERC' AND v_sal < 1000) THEN
            INSERT INTO table_resultat
            VALUES ( v_nom, v_departement, v_salaire );
        END IF;
        EXIT WHEN v_nom = 'SCOTT';
    END LOOP;
    ...
END;
```

Dans cet exemple, la condition exprimée par EXIT WHEN est loin d'être idéale car nous ne sommes pas certains de la rencontrer! Une instruction spécifique nous permettra de sortir de la boucle LOOP lorsque le dernier enregistrement du curseur aura été traité.

Fermeture d'un curseur

L'instruction CLOSE ferme le curseur ouvert par l'instruction OPEN et libère les ressources consommées.

```
■ CLOSE nom_curseur ;
```

Par exemple :

```
DECLARE
    -- déclaration de variables
    v_nom          emp.ename%TYPE;
    v_departement  emp.job%TYPE;
    v_salaire      emp.sal%TYPE;

    -- déclaration de curseurs
    CURSOR c_employe IS SELECT ename, deptno, sal FROM emp
    WHERE job = 'MANAGER'
    ORDER BY ename ;
BEGIN
    OPEN c_employe ;-- ouverture du curseur
    LOOP
        -- traitement des lignes du curseur
        FETCH c_employe INTO
            v_nom,v_fonction, v_salaire ;
        IF (v_fonction = 'CLERC' AND v_sal < 1000) THEN
            INSERT INTO table_resultat
            VALUES ( v_nom, v_departement, v_salaire );
        END IF;
        EXIT WHEN v_nom = 'SCOTT';
    END LOOP;
    CLOSE c_employe ;-- fermeture du curseur
END;
```

Les attributs d'un curseur

Les attributs permettent de connaître l'état de vos curseurs :

- %FOUND et %NOTFOUND déterminent si toutes les lignes retournées par le curseur ont été traitées ;
- %ISOPEN précise si le curseur est ouvert ;
- %ROWCOUNT indique le nombre de lignes déjà traitées.

Les attributs du curseur : %FOUND

Selon l'ordre SQL, dans le cas d'un curseur implicite, l'attribut SQL %FOUND prend la valeur :

- INSERT, UPDATE, DELETE : SQL%FOUND = TRUE, si l'ordre a traité au moins une ligne ;
- SELECT... INTO : SQL%FOUND = TRUE, si l'ordre SELECT a renvoyé une seule ligne.

Dans le cas d'un curseur explicite :

- nom_du_curseur%FOUND = TRUE, si le dernier FETCH a retourné une ligne.

Les attributs du curseur : %NOTFOUND

C'est le pendant de %FOUND. Suivant l'ordre SQL, dans le cas d'un curseur implicite, l'attribut SQL %NOTFOUND prend la valeur :

- INSERT, UPDATE, DELETE : SQL%NOTFOUND = TRUE, si l'ordre n'a traité aucune ligne ;
- SELECT... INTO : SQL%NOTFOUND = TRUE, si l'ordre SELECT n'a retourné aucune ligne.

Dans le cas d'un curseur explicite : nom_du_curseur%NOTFOUND = TRUE, si le dernier FETCH n'a pas renvoyé de ligne.

« nom_curseur%NOTFOUND » est la condition idéale très largement utilisée pour sortir d'une boucle FETCH.

Les attributs d'un curseur : %ISOPEN

Cet attribut permet de vérifier si le curseur considéré a été ouvert.

Les attributs du curseur : %ROWCOUNT

L'attribut %ROWCOUNT renvoie dans une valeur numérique le nombre de lignes traitées.

Dans le cas d'un curseur implicite INSERT, UPDATE, DELETE, SQL%ROWCOUNT retourne le nombre de lignes traitées par votre ordre SQL ;

- SELECT... INTO : SQL%ROWCOUNT prend les valeurs suivantes :
- -SQL%ROWCOUNT = 0, si le SELECT... INTO n'a renvoyé aucune ligne ;
- -SQL%ROWCOUNT = 1, si le SELECT... INTO n'a renvoyé qu'une seule ligne ;
- -SQL%ROWCOUNT = 2, si le SELECT... INTO a renvoyé plus d'une ligne.

Dans le cas d'un curseur explicite : nom_curseur%ROWCOUNT traduit l'énième ligne retournée par le FETCH.

Exemple d'utilisation d'un curseur explicite

Dans l'exemple suivant, nous recherchons le salaire de chaque employé ainsi que la moyenne des salaires des employés qui font le même métier que lui.

```
DECLARE
    -- déclaration de variables
    v_nom          emp.ename%TYPE;
    v_salaire      emp.sal%TYPE;
    v_fonction     emp.job%TYPE;
    v_moyenne      NUMBER(7,2);

    -- déclaration de curseurs
    CURSOR c_employe IS SELECT ename, sal, job FROM emp
    ORDER BY ename ;
```

```
BEGIN
  IF NOT (c_employe%ISOPEN) THEN
    OPEN c_employe ;
  END IF;
  LOOP
    FETCH c_employe INTO
      v_nom,v_salaire , v_fonction ;

    -- tester %NOTFOUND juste après le FETCH !
    EXIT WHEN c_employe%NOTFOUND ; --sortie boucle

    -- l'ordre SQL suivant est implicite
    -- pas de curseur car une seule valeur retournée
    SELECT avg(sal) INTO v_moyenne FROM emp
    WHERE job = v_fonction ;

    -- stockage des résultats dans une table
    INSERT INTO table_resultat
    VALUES ( v_nom, v_salaire, v_moyenne );

  END LOOP;
  CLOSE c_employe ;
END;
```

Dans cet autre exemple, nous interrompons le traitement d'un curseur si plus de 1 000 enregistrements sont traités :

```
DECLARE
  -- déclaration de variables
  v_nom          emp.ename%TYPE;
  v_salaire      emp.sal%TYPE;
  v_fonction     emp.job%TYPE;
  v_moyenne      NUMBER(7,2);

  -- déclaration de curseurs
  CURSOR c_employe IS SELECT ename, sal, job FROM emp
  ORDER BY ename ;
BEGIN
  OPEN c_employe ; -- ouverture du curseur
  LOOP
    FETCH c_employe INTO
      v_nom,v_salaire , v_fonction ;

    -- tester %NOTFOUND juste après le FETCH !
    EXIT WHEN ( c_employe%NOTFOUND
    OR c_employe%ROWCOUNT > 1000); --sortie boucle
    -- traitement à effectuer
    -- ...

  END LOOP;
  CLOSE c_employe ;
END;
```

Simplification d'écriture des curseurs

La syntaxe de base est délicate, dans la mesure où vous devez manipuler des variables et des noms quasi identiques à plusieurs endroits :

- nommer la liste des colonnes de l'ordre SQL dans le zone DECLARE lors de la déclaration du curseur ;
- déclarer les variables récupérant les valeurs du SELECT... INTO dans la section DECLARE ;
- effectuer une correspondance entre les variables déclarées et les colonnes de votre table lors du FETCH nom_curseur INTO variables.

Par exemple :

```
DECLARE
    -- déclaration du curseur
    CURSOR mon_curseur IS
        SELECT col1,
               col2 ,
               ...,
               coln FROM table ;
    -- déclaration des variables
    v_col1      table.col1%TYPE ;
    v_col2      table.col1%TYPE ;
    ...         ...
    v_coln      table.coln%TYPE ;

BEGIN
    OPEN mon_curseur ;
    ...
    FETCH mon_curseur INTO
        v_col1 ,
        v_col2 ,
        ...
        v_coln ;

    ...
    EXIT WHEN mon_curseur%NOTFOUND ;
    ...
    -- suite du traitement ...
    -- on utilise les variables par :
    ... v_col1 ...
    ... v_col2 ...
    ...
    ... v_coln ...
    ...
    END LOOP;
    CLOSE mon_curseur ;
END;
```

La simplification proposée déclare implicitement une structure dont les éléments sont du même type que les colonnes retournées par le curseur.

Vous devez utiliser la syntaxe :

```
DECLARE
    CURSOR mon_curseur IS mon_ordre_select ;
    nom_structure mon_curseur%ROWTYPE ;
```

Pour utiliser les colonnes de la structure :

```
nom_structure.nom_colonne_definie_dans_mon_ordre_select
```

La structure est renseignée par le FETCH :

```
FETCH mon_curseur INTO nom_structure ;
```

L'exemple précédent s'en trouve simplifié :

```
DECLARE
    -- déclaration du curseur
    CURSOR mon_curseur IS
        SELECT col1,
               col2 ,
               ...,
               coln FROM table ;

    -- déclaration de ma structure
    ma_structure mon_curseur%ROWTYPE;

BEGIN
    OPEN mon_curseur ;
    LOOP
        -- il n'est plus nécessaire de nommer
        -- les colonnes lors du FETCH
        FETCH mon_curseur INTO ma_structure ;
        ...
        EXIT WHEN mon_curseur%NOTFOUND ;
        ...
        -- suite du traitement ...
        -- on utilise les variables par :
        ... ma_structure.col1 ...
        ... ma_structure.col2 ...
        ...
        ... ma_structure.coln ...

    END LOOP;
    CLOSE mon_curseur ;
END;
```

Simplification d'écriture des curseurs : FOR... LOOP

La syntaxe FOR... LOOP permet de simplifier les étapes OPEN, FETCH, CLOSE de la vie d'un curseur. Elle se positionne sur un « enregistrement » contenant les colonnes définies dans l'ordre SQL, ouvre le curseur, « fetch » un enregistrement dans le « record ». Le curseur est automatiquement fermé à la fin du traitement des enregistrements.

```
DECLARE
    CURSOR mon_curseur IS
        SELECT ename, job, sal FROM emp ;
    -- il n'est plus nécessaire de déclarer votre structure !
    ...
BEGIN
    FOR rec_employe IN mon_curseur LOOP
        -- ouverture automatique du curseur !
        -- FETCH automatique du curseur !
        -- condition de sortie %NOTFOUND automatique !
        ...
        -- votre traitement
        ...
    END LOOP;
    -- fermeture automatique du curseur !
END;
```

L'utilisation de cette syntaxe génère implicitement le code suivant :

```
DECLARE
    CURSOR mon_curseur IS
        SELECT ename, job, sal FROM emp ;
    rec_employe mon_curseur%ROWTYPE ;
    ...
BEGIN
    OPEN mon_curseur ;
    LOOP
        FETCH mon_curseur INTO rec_employe ;
        EXIT WHEN mon_curseur%NOTFOUND ;
        ...
        -- votre traitement
        ...
    END LOOP;
    CLOSE mon_curseur ;
END;
```

L'exemple précédent devient alors :

```
DECLARE
    -- déclaration du curseur
    CURSOR mon_curseur IS
        SELECT col1,
               col2 ,
               ....
               coln FROM table ;

    -- pas de déclaration de structure
```

```

BEGIN

    FOR ma_structure IN mon curseur LOOP
    ...
    -- le FETCH automatique !
    -- la condition de sortie est automatique !
    ...
    -- suite du traitement ...
    -- on utilise les variables par :
    ... ma_structure.col1 ...
    ... ma_structure.col2 ...
    ...
    ... ma_structure.coln ...

    END LOOP;
    -- fermeture automatique !
END;
```

Nous vous conseillons de bien maîtriser cette structure avant de la mettre en place dans vos codes. Si vous hésitez, restez au premier niveau de simplification de l'écriture des curseurs.

Simplification d'écriture des curseurs : FOR... IN

La syntaxe FOR... IN est encore plus concise. Elle évite de déclarer le curseur dans la section DECLARE.

```

-- plus de déclaration du curseur en section DECLARE
BEGIN
    FOR rec_employe IN (SELECT ename, job, sal FROM emp )
    LOOP
    -- ouverture automatique du curseur !
    -- FETCH automatique du curseur !
    -- condition de sortie %NOTFOUND automatique !
    ...
    -- votre traitement
    ...
    END LOOP;
    -- fermeture automatique du curseur !
END;
```

L'exemple précédent devient alors :

```

DECLARE
    -- pas de déclaration de curseur
    -- pas de déclaration de structure
BEGIN

    FOR ma_structure IN ( SELECT col1,
                           col2 ,
                           ...,
                           coln FROM table );
```



```
    LOOP
    ...
    -- 1e FETCH automatique !
    -- la condition de sortie est automatique !
    ...
    -- suite du traitement ...
    -- on utilise les variables par :
    ... ma_structure.col1 ...
    ... ma_structure.col2 ...
    ...
    ... ma_structure.coln ...

    END LOOP;
    -- fermeture automatique !
END;
```

Pour des questions de clarté du code, nous préférons la syntaxe FOR... IN... LOOP qui conserve les ordres SQL de curseurs en zone de déclaration.

Les curseurs paramétrés

Dans tous les curseurs précédents, les ordres SQL étaient fixes et n'acceptaient aucun paramètre. Il est souvent utile de réutiliser un même curseur avec des valeurs différentes, dans un même bloc PL/SQL :

```
DECLARE
    CURSOR mon_curseur (param1 TYPE,
                        param2 TYPE,
                        ... )
    IS mon_ordre_select ;
    -- l'ordre select utilise les paramètres param1, param2...
BEGIN
    OPEN mon_curseur (val1, val2,...) ;
    ...
    CLOSE mon_curseur;
END;
```

Les types des paramètres sont les suivants : CHAR, NUMBER, DATE, BOOLEAN, leur longueur n'est pas spécifiée. Le passage des valeurs des paramètres s'effectue à l'ouverture du curseur.

Mettre à jour des données avec un curseur : CURRENT-OF

Jusqu'à présent, tous les exemples de curseurs étaient en lecture seule. Aucune modification des données retournées par un curseur n'a été effectuée. C'est l'objet de la clause CURRENT-OF.

Elle permet d'accéder directement, en modification ou en suppression, à la ligne que vient de renvoyer l'ordre FETCH. Au préalable, dans la déclaration du curseur, il faut

réserver les lignes qui seront modifiées par la pose d'un verrou d'intention (...FOR UPDATE OF colonne_à_modifier...).

```
--
-- Exemple d'un curseur mettant à jour les données retournées
--
DECLARE
    CURSOR mon_curseur IS SELECT ename, sal, comm
                          FROM emp
                          FOR UPDATE OF sal ;

BEGIN
    FOR ma_structure IN mon_curseur
    LOOP
        IF ma_structure.comm IS NOT NULL THEN
            -- si pas de commission, le salaire augmente de 5%
            UPDATE emp SET sal = sal*1.05
            WHERE CURRENT OF mon_curseur;
        END IF;
    END LOOP;
    -- ne pas oublier de valider les modifications
    COMMIT;
END;
```

Gestion des erreurs

Dans l'introduction de ce chapitre, nous avons mentionné que le PL/SQL disposait d'un mécanisme simple pour traiter les erreurs et les exceptions rencontrées dans les blocs PL/SQL. C'est l'objet de la section EXCEPTION.

La section EXCEPTION

Dans un bloc PL/SQL, la section EXCEPTION permet de traiter les erreurs survenues lors de l'exécution du bloc PL/SQL. Il existe deux grandes familles d'erreurs :

- celles retournées par Oracle 10g (table inaccessible, conflit de mise à jour...);
- celles dues à l'écriture du programme utilisateur.

Un certain nombre d'erreurs Oracle 10g (les plus courantes) sont prédéfinies. Pour des erreurs utilisateur, il convient de déclarer un nom d'erreur dans la section DECLARE pour l'utiliser dans la section EXCEPTION.

Dans tous les cas, il faut écrire le code PL/SQL traitant l'erreur dans la section EXCEPTION du bloc PL/SQL.

```
--
-- Syntaxe d'utilisation des exceptions
--
DECLARE
    ...
    nom_erreur      EXCEPTION ;
    ...
```

```
BEGIN
    ...
    IF (problème rencontré)
        THEN RAISE nom_erreur ;
    ...
EXCEPTION
    WHEN nom_erreur THEN
        (traitement de l'erreur);

END;
```

Attention : vous sortez du bloc PL/SQL après le traitement de l'erreur.

Pour les erreurs Oracle 10g prédéfinies, vous n'avez pas besoin de placer une instruction d'appel à l'exception dans le code PL/SQL :

```
--
-- Syntaxe d'utilisation des exceptions Oracle 10g prédéfinies
--
DECLARE
    ...
BEGIN
    ...
EXCEPTION
    WHEN ZERO_DIVIDE THEN
        (traitement de l'erreur);

    WHEN NO_DATA_FOUND THEN
        (traitement de l'erreur);
    ...
END;
```

Nous vous conseillons de toujours utiliser l'instruction WHEN OTHERS dans le traitement des blocs PL/SQL. Cela vous permet de conserver une trace de l'erreur rencontrée.

```
--
-- Utilisation de WHEN-OTHERS
--
DECLARE
    v_code_erreur          NUMBER ;
    v_message_erreur       VARCHAR2(50);

BEGIN
    ...
EXCEPTION
    WHEN ZERO_DIVIDE THEN
        (traitement de l'erreur);

    WHEN NO_DATA_FOUND THEN
        (traitement de l'erreur);
```

```
...
WHEN OTHERS THEN
    v_code_erreur := SQLCODE;
    v_message_erreur := SQLERRM ;
    INSERT INTO table_debug
    VALUES ( sysdate, v_code_erreur, v_message_erreur );
    COMMIT;
END;
```

SQLCODE et SQLERRM permettent de connaître le code et le message associé à l'erreur rencontrée.

Procédures et fonctions

Dans ce chapitre, tous les exemples précédents concernaient des programmes PL/SQL « anonymes », conservés à l'extérieur de la base de données. Les procédures et les fonctions sont des sous-programmes hébergés par la base de données : ils peuvent être appelés à partir d'une autre procédure, fonction ou encore depuis un programme exécutable extérieur à la base de données. Un programme Java peut ainsi lancer un programme PL/SQL.

Qu'est-ce qu'un sous-programme ?

Un sous-programme est un bloc PL/SQL nommé, qui peut accepter des paramètres et être appelé. Le PL/SQL possède deux types de sous-programmes : les procédures et les fonctions. Généralement, les fonctions sont utilisées pour effectuer un calcul et les procédures pour exécuter un traitement.

Reprenons l'exemple de l'augmentation de salaire d'un employé qui a déjà été réalisé sous forme d'un bloc PL/SQL anonyme. Utilisons maintenant une procédure pour effectuer la même tâche.

```
--
-- Exemple d'une procédure
--
PROCEDURE augmente_employe (nom_employe IN VARCHAR2) IS
-- utilisation d'un curseur de mise à jour qui accepte un paramètre
    CURSOR mon_curseur (nom_employe CHAR)
        IS SELECT sal FROM emp
        WHERE upper(ename) = upper(nom_employe)
        FOR UPDATE OF sal, comm ;
BEGIN
    FOR ma_structure IN mon_curseur (nom_employe)
    LOOP
        IF ma_stucture.sal < 1000 THEN      -- test sur le salaire
            UPDATE emp SET sal = sal*1,1
            WHERE CURRENT OF mon_curseur; -- augmentation de 10%
```

```
ELSE
    UPDATE emp SET comm = comm + 100
    WHERE CURRENT OF mon_curseur;
END IF;
COMMIT;
END LOOP;
EXCEPTION
    WHEN OTHERS THEN
        -- traitement des exceptions...
        null;
END;
```

Pour réaliser cette procédure, il faut l'appeler en lui transmettant le ou les paramètres requis.

Avantages d'un sous-programme

De trop nombreux programmes utilisant Oracle 10g comme base de données ne profitent pas des capacités de programmation offertes par les sous-programmes. Pourtant, ces derniers présentent bien des avantages :

- ils offrent une *modularité* des applications. Les programmes complexes peuvent être découpés en sous-programmes plus accessibles et plus faciles à écrire et à *maintenir* ;
- une fois validé, un sous-programme peut être *réutilisé* par d'autres applications ;
- dans la logique applicative de vos programmes, les sous-programmes vous fournissent un niveau d'*abstraction* en isolant des pièces de code.

Les procédures

Une procédure est un sous-programme utilisant la syntaxe suivante :

```
--
-- Syntaxe d'une procédure
--
PROCEDURE ma_procedure (parametre1,
                        parametre2,
                        ...parametre_n) IS
    -- zone de déclaration :
    -- * des variables locales
    -- * des curseurs
    -- * des exceptions
BEGIN
    -- traitements
EXCEPTION
    -- traitement des exceptions...
END;
```

Une variable *paramètre* doit se conformer à la syntaxe suivante :

```
nom_parametre [ IN | OUT | IN OUT ] type_du_parametre
```

Le type_du_parametre doit être NUMBER, CHAR, BOOLEAN... Ainsi, un type NUMBER(7,2) n'est pas accepté.

Les fonctions

Une fonction est un sous-programme destiné à calculer une valeur. Sa syntaxe est identique à celle d'une procédure, mais elle retourne une valeur par la clause RETURN. Cette clause est obligatoire.

Attention, une fonction se nomme FUNCTION. Cette proximité d'orthographe est source de nombreuses erreurs difficiles à corriger...

```
--
-- Syntaxe d'une fonction
--
FUNCTION ma_fonction (parametre1,
                      parametre2,
                      ...parametre_n)
    RETURN type_de_la_variable_retournée IS
    -- zone de déclaration :
    -- * des variables locales
    -- * des curseurs
    -- * des exceptions
BEGIN
    -- traitements
    -- clause RETURN
EXCEPTION
    -- traitement des exceptions...
    -- clause RETURN
END;
```

Une variable *paramètre* doit se conformer à la syntaxe suivante :

```
nom_parametre [ IN | OUT | IN OUT ] type_du_parametre
```

Le type_du_parametre doit être NUMBER, CHAR, BOOLEAN... Ainsi, un type NUMBER(7,2) n'est pas accepté.

Pour la clause RETURN, il suffit d'indiquer son type : NUMBER, CHAR, BOOLEAN, sans lui donner de nom.

```
--
-- Exemple d'une fonction
--
FUNCTION francs_vers_euro (v_valeur_en_francs IN NUMBER )
    RETURN REAL IS
    v_valeur_en_euro REAL ;
```

```
BEGIN
    v_valeur_en_euro := v_valeur_en_francs / 6.55957 ;
    RETURN v_valeur_en_euro ;
END;
```

Création, suppression des procédures et fonctions

Comme tout objet SQL, une procédure ou une fonction peut être créée, modifiée, supprimée.

- Création

```
CREATE PROCEDURE ...
```

```
CREATE FUNCTION ...
```

- Modification

```
CREATE OR REPLACE PROCEDURE ...
```

```
CREATE OR REPLACE FUNCTION ...
```

- Suppression

```
DROP PROCEDURE ...
```

```
DROP FUNCTION ...
```

Nous venons d'aborder le principe et la syntaxe des procédures et fonctions. Comment à présent les appeler pour les exécuter ? C'est ce que traite le paragraphe suivant.

Utilisation de SQL*Plus avec les procédures et les fonctions

Pour créer, modifier, supprimer des sous-programmes, vous avez besoin d'un environnement de développement. L'outil standard livré avec toutes les versions d'Oracle 10g est SQL*Plus. Il existe de nombreux autres environnements de développement, mais nous préférons effectuer un exemple complet à partir de celui-ci, de sorte que vous puissiez facilement créer vos propres applications.

Création d'une fonction comportant une erreur

Connectez-vous à SQL*Plus et étudions la création d'une fonction comportant une erreur :

```
SQL> --
SQL> -- Création de la fonction f_euro comportant une erreur
SQL> --
SQL> CREATE OR REPLACE FUNCTION f_euro ( v_valeur_en_francs IN NUMBER )
2   RETURN REAL IS
3       v_valeur_en_euro REAL;
4   BEGIN
5       v_valeur_en_euro := v_valeur_en_francs / 6,55957 ;
```

```

6      return v_valeur_en_euro ;
7  END;
8  /

```

Warning: Function created with compilation errors.

Même si une erreur est survenue lors de la compilation, le code de la fonction est maintenant présent dans la base de données. Nous expliquons comment visualiser ce code plus loin dans ce paragraphe.

Pour repérer l'erreur, utilisez la commande SQL*Plus `SHOW ERROR`, qui affiche la dernière erreur rencontrée :

```

SQL> --
SQL> -- Visualisation des erreurs
SQL> --
SQL> -- Optimise l'affichage des messages d'erreur
SQL> set CHARWIDTH 132
SQL>
SQL> Show error
Errors for FUNCTION F_EURO:

LINE/COL ERROR
-----
5/44      PLS-00103: Encountered the symbol "," when expecting one of
          the following:
          * & = - + ; < / > in mod not rem an exponent (**)
          <> or != or ~= >= <= <> and or like between is null is not ||
          is dangling
          The symbol "*" was inserted before "," to continue.

```

Sont indiqués la ligne et la colonne où se situe l'erreur. Ici, l'erreur SQL vient de l'utilisation d'une virgule à la place du point, pour séparer les décimales d'un chiffre (rassurez-vous, cette caractéristique par défaut peut être changée pour respecter les standards français).

Compilation d'une fonction sans erreur

```

SQL> --
SQL> -- Fonction corrigée ( 6.55957 au lieu de 6,55957 )
SQL> --
SQL> CREATE OR REPLACE FUNCTION f_euro ( v_valeur_en_francs IN NUMBER )
2  RETURN REAL IS
3      v_valeur_en_euro REAL;
4  BEGIN
5      v_valeur_en_euro := v_valeur_en_francs / 6.55957 ;
6      return v_valeur_en_euro ;
7  END;
8  /

```

Function created.

Votre fonction est créée et compilée sans erreur.

Exécution d'une fonction compilée sans erreur

La fonction est correctement créée. Pour la tester, utilisons un bloc PL/SQL anonyme qui l'exécute.

```
SQL> --
SQL> -- Exécution de la fonction à partir d'un bloc PL/SQL anonyme
SQL> --
SQL> DECLARE
2     v_valeur_francs REAL := '1000' ;
3     v_valeur_euro   REAL;
4 BEGIN
5     v_valeur_euro := f_euro ( v_valeur_francs );
6 END;
7 /
```

PL/SQL procedure successfully completed.

La fonction s'est exécutée sans erreur, sans que nous percevions la transformation de la valeur de francs en euro. Pour visualiser des résultats et des variables, il est possible grâce à SQL*Plus de les afficher à partir de procédures, fonctions et blocs PL/SQL anonymes.

Affichage de variables contenues dans un bloc PL/SQL

Pour cela, il faut inclure le package standard *dbms_output.put_line* dans vos blocs PL/SQL. Cette procédure permet d'afficher le contenu de variables que vous lui transmettez. Pour activer la visualisation des variables transmises à DBMS_OUTPUT.PUT_LINE il faut auparavant déclarer *set serveroutput on* à partir de l'environnement SQL*Plus :

```
SQL> --
SQL> -- Visualisation des variables
SQL> --
SQL> set serveroutput on
SQL>
SQL> DECLARE
2     v_valeur_francs REAL := '1000' ;
3     v_valeur_euro   REAL;
4 BEGIN
5     dbms_output.put_line('Valeur en francs : '||to_char(v_valeur_francs));
6     v_valeur_euro := f_euro ( v_valeur_francs );
7     dbms_output.put_line('Valeur en euros  : '||to_char(v_valeur_euro));
8 END;
9 /
```

Valeur en francs : 1000
Valeur en euros : 152.449017237410379033991557373425392213

PL/SQL procedure successfully completed.

Retrouver la liste des sous-programmes créés dans une base

Dès qu'une procédure, une fonction, un package ou un trigger a été créé dans une base Oracle 10g, son code original y est conservé, même si la compilation de ce code a produit des erreurs. Pour retrouver la liste des procédures, des triggers et des fonctions présents dans une base Oracle 10g, connectez-vous à SQL*Plus sous un compte ayant les privilèges administrateur et lancez l'ordre SQL :

```
SQL> --
SQL> -- Affichage de la liste des procédures, fonctions, triggers
SQL> -- d'une base de données
SQL> --
SQL> col object_name for a20
SQL> col owner for a20
SQL>
SQL> select owner, object_name, object_type, status from dba_objects
2  where object_type in ('PROCEDURE','FUNCTION','TRIGGER','PACKAGE')
3  and owner not in ('SYS','SYSTEM')
4  order by owner, object_name
5  /
```

OWNER	OBJECT_NAME	OBJECT_TYPE	STATUS
GB_SITE	CORNERS_IU	TRIGGER	VALID
GB_SITE	HOME	PROCEDURE	VALID
GB_SITE	PERSPECTIVES_I	TRIGGER	VALID
GB_SITE	TOPICS_I	TRIGGER	VALID
GB_SITE	TRAVELER	PROCEDURE	VALID
GB_SITE	WWV_URLENCODE	FUNCTION	VALID
SCOTT	F_EURO	FUNCTION	VALID
SCOTT	GILLES	PROCEDURE	VALID
SCOTT	GIVE_RAISE	PROCEDURE	INVALID
SCOTT	TESTJDBC	PROCEDURE	INVALID
....			

Vous pouvez adapter cet ordre suivant vos besoins. Dans cette liste figurent des triggers, des procédures et des fonctions. Le statut de chaque objet vous précise si sa compilation a été effectuée avec ou sans erreur. Cet ordre SQL fait partie de l'annexe 3, *Procédures pour le DBA*.

Retrouver le code d'un sous-programme créé dans une base

Une fois que le nom et le propriétaire d'un objet sont déterminés par l'ordre SQL précédent, vous pouvez récupérer le code de cet objet. Par exemple, pour la fonction F_EURO qui a été créée précédemment, il faut procéder comme suit :

```
SQL> --
SQL> -- Récupérer le source du code d'un objet
SQL> --
1  select text from dba_source
```

```
2 where owner = 'SCOTT'
3 and name = 'F_EURO'
4* order by line;

TEXT
-----FUNCTION
➡f_euro ( v_valeur_en_francs IN NUMBER )
RETURN REAL IS
    v_valeur_en_euro REAL;
BEGIN
    v_valeur_en_euro := v_valeur_en_francs / 6.55957 ;
    return v_valeur_en_euro ;
END;

7 rows selected.
```

Bien entendu, le code du programme PL/SQL n'est visible que si le « wrapper » PL/SQL n'a pas été utilisé. Dans ce cas, le code est crypté et vos techniques de programmation restent confidentielles.

Les packages

Un package PL/SQL Oracle 10g permet de regrouper un ensemble de procédures, de fonctions et de variables au sein d'un ensemble cohérent. Nous présentons succinctement les packages et nous nous attachons plus particulièrement à leur utilisation, par le biais de packages standard fournis avec Oracle 10g.

Qu'est-ce qu'un package ?

Toutes les procédures et fonctions d'une application peuvent être conservées dans un même package. Comme les procédures et les fonctions, les packages sont stockés dans la base de données Oracle 10g. Ils sont composés de deux parties bien distinctes :

- la partie *spécification* : où sont déclarées l'ensemble des procédures, des fonctions et des variables accessibles aux applications utilisant votre package ;
- la partie *body* ou *corps* : lieu de programmation du code des procédures et des fonctions visibles dans la partie spécification. De plus, il est possible d'écrire des procédures et fonctions internes à la partie body, qui ne peuvent être atteintes par les applications accédant au package.

Vous disposez ainsi de code « privé » (interne à la partie body du package) et de code « public » (déclaré dans la partie spécification du package).

Appel d'un objet d'un package

Pour utiliser un objet de package, il faut nommer ce dernier puis l'un des objets publics déclarés dans sa partie spécification.

Par exemple, le package standard DBMS_OUTPUT contient dans sa spécification la procédure PUT_LINE. Son utilisation est NOM_PACKAGE.NOM_OBJET :

```
SQL> --
SQL> -- Visualisation des variables
SQL> --
SQL> set serveroutput on
SQL>
SQL> DECLARE
2     v_valeur_francs REAL := '1000' ;
3     v_valeur_euro   REAL;
4 BEGIN
5     dbms_output.put_line('Valeur en francs : '||to_char(v_valeur_francs));
6     v_valeur_euro := f_euro ( v_valeur_francs );
7     dbms_output.put_line('Valeur en euros  : '||to_char(v_valeur_euro));
8 END;
9 /

Valeur en francs : 1000
Valeur en euros  : 152.44901

PL/SQL procedure successfully completed.
```

Les packages PL/SQL standards

Toutes les bases de données Oracle 10g sont livrées avec des packages spécifiques qui les aident à construire des applications. PL/SQL est un langage puissant qui permet d'écrire des applications transactionnelles. Néanmoins, comment réaliser des opérations telles que soumettre des travaux (*job scheduling* en anglais), créer des Workflows (circuits de validation), lire/écrire dans des fichiers du système d'exploitation ou encore créer des tables ou des utilisateurs de vos bases Oracle 10g ?

C'est ce que proposent les packages DBMS_STANDARD, DBMS_OUTPUT, UTL_FILE, DBMS_JOB, DBMS_LOB, UTL_HTTP, DBMS_SQL, DBMS_ALERT, DBMS_PIPE, DBMS_AQ, DBMS_LDAP, DBMS_LOGMNR, DBMS_XML, UTL_MAIL, DBMS_UTILITY et bien d'autres encore.

On rencontre trop souvent le cas d'applications complexes qui ne font que reconstruire des fonctions déjà présentes en standard dans ces packages. Consultez la documentation Oracle pour découvrir la palette proposée par les packages commençant par DBMS_, UTL_, SDO_, HTMLDB_. Il en existe plus d'une centaine !

Les principales tâches dévolues à ces packages sont les suivantes :

- lire et écrire dans des fichiers ;
- manipuler de larges objets binaires (LOB) ;
- créer des structures de données (DDL) et construire dynamiquement des ordres SQL ;

- communiquer entre processus ;
- soumettre des travaux.

Il existe d'autres packages qui servent à appeler des procédures distantes ou encore à gérer le verrouillage des données. Les packages que nous abordons ici sont les plus répandus et les plus utiles. Les développeurs et administrateurs d'Oracle 10g doivent se familiariser avec ces packages pour les utiliser au mieux.

Pour plus de clarté, ils sont groupés par fonctions réalisées, ce qui simplifie la recherche lorsque, pour une réalisation précise, on ignore le package nécessaire. Ce paragraphe présente succinctement la syntaxe de ces packages ainsi que les grands principes de leur utilisation.

Comment créer ces packages ?

Au fil de la parution de nouvelles versions d'Oracle 10g, la liste des packages s'est allongée. Pour l'ensemble d'entre eux, seule la partie publique ou *spécification* est visible. La partie privée ou *body* est cryptée pour préserver les copyrights d'Oracle. Nous vous engageons à visualiser la spécification de ces packages avant de les utiliser. Vous y trouverez de nombreux commentaires ainsi que des exemples.

L'emplacement des codes de création de ces packages dépend du système d'exploitation. Sous Windows, ils se situent dans \$ORACLE_HOME\rdbms\admin et sous Linux dans \$ORACLE_HOME/rdbms/admin. Ils apparaissent sous l'utilisateur SYS lors de la création de votre base de données par le lancement du script *catproc.sql*. Celui-ci est chargé de démarrer l'ensemble des scripts PL/SQL d'installation des packages. L'utilisateur de ces packages doit disposer de la permission EXECUTE. Des synonymes publics sont créés, il n'est donc pas nécessaire de les doter comme préfixe du nom de leur propriétaire, SYS, lorsque vous les utilisez.

Les fonctions standards : DBMS_STANDARD

Un grand nombre de commandes que vous utilisez dans le langage PL/SQL sont écrites en PL/SQL et fournies par le package DBMS_STANDARD. Par exemple, dans le traitement des erreurs, la clause WHEN_OTHERS rencontrée dans la section EXCEPTION d'un bloc PL/SQL est une procédure de DBMS_STANDARD. Nous n'abordons pas le contenu de ce package, traité dans les paragraphes précédents concernant la syntaxe et l'utilisation du langage PL/SQL.

Émettre des messages d'un bloc PL/SQL : DBMS_OUTPUT

DBMS_OUTPUT est un package à la fois simple et important : simple par son utilisation et important par les services qu'il rend. Il est largement utilisé pour envoyer des messages contrôlant l'exécution d'un programme ou encore lors du débogage. Les informations

envoyées sont conservées dans la zone mémoire allouée à l'instance Oracle 10g. Elles sont donc perdues dès que la base de données est arrêtée.

Il est possible de visualiser les messages adressés par le package DBMS_OUTPUT dans les outils SQL*Plus et Server Manager en positionnant au préalable SET SERVEROUTPUT ON. Tous les messages en provenance de vos blocs PL/SQL, procédures et fonctions, envoyés par DBMS_OUTPUT.PUT_LINE sont alors visualisés dans SQL*Plus ou Server Manager.

L'exemple suivant, déjà abordé, présente la syntaxe et l'utilisation de DBMS_OUTPUT.

```
SQL> --
SQL> -- Visualisation des variables
SQL> --
SQL> set serveroutput on
SQL>
SQL> DECLARE
2     v_valeur_francs REAL := '1000' ;
3     v_valeur_euro   REAL;
4 BEGIN
5     dbms_output.put_line('Valeur en francs : '||to_char(v_valeur_francs));
6     v_valeur_euro := f_euro ( v_valeur_francs );
7     dbms_output.put_line('Valeur en euros  : '||to_char(v_valeur_euro));
8 END;
9 /

Valeur en francs : 1000
Valeur en euros  : 152.44901723741037903399155737

PL/SQL procedure successfully completed.
```

Écrire et lire dans des fichiers : UTL_FILE

Le package PL/SQL UTL_FILE permet à des fonctions ou procédures qui s'exécutent dans la base de données de créer, lire et écrire dans des fichiers situés sur le serveur hébergeant la base. Il est alors aisé pour les clients distants de créer des fichiers contenant des données en provenance de la base ou encore de récupérer des informations situées dans des fichiers. Une autre utilisation possible consiste à insérer des balises HTML pour créer des pages Web incorporant des données de la base Oracle 10g.

Paramétrage du fichier *init<SID>.ora*

Les fichiers créés doivent se situer dans des répertoires indiqués par la variable UTL_FILE_DIR du fichier *init<SID>.ora* de démarrage de votre base. Par exemple :

```
UTL_FILE_DIR=C:\oracle\admin\TEST\temp
UTL_FILE_DIR=C:\temp
```

Seuls les répertoires cités dans cette liste sont accessibles par UTL_FILE. Tous les autres répertoires ou sous-répertoires sont interdits. Il faut aussi que l'utilisateur *oracle* en possède les droits de lecture/écriture.

Syntaxe du package UTL_FILE

Le tableau suivant résume les principales procédures disponibles dans le package UTL_FILE :

Fonction/Procédure	Description
FOPEN	Ouvre un fichier en lecture/écriture. Si le fichier n'existe pas, il est créé.
IS_OPEN	Détermine si le fichier considéré est ouvert.
FCLOSE	Ferme le fichier.
FCLOSE_ALL	Ferme tous les fichiers ouverts.
GET_LINE	Lit la ligne courante d'un fichier ouvert.
PUT	Écrit une ligne dans un fichier. N'ajoute pas de caractère de fin de ligne.
PUT_LINE	Écrit une ligne dans un fichier. Ajoute un caractère de fin de ligne. Ce caractère dépend du système d'exploitation (CR sous Linux, CR-LF sous Windows).
PUTF	Écrit une ligne formatée.
NEW_LINE	Écrit un caractère de fin de ligne (dépendant du système d'exploitation).
FFLUSH	Force l'écriture physique dans le fichier.

Création d'une procédure de test

Dans l'exemple suivant, nous créons une procédure destinée à écrire dans un fichier présent sous le système d'exploitation du serveur.

```
--
-- Test du package UTL_FILE
--
CREATE OR REPLACE PROCEDURE Test_util_file IS
  ref_fichier UTL_FILE.FILE_TYPE;
BEGIN
  ref_fichier := UTL_FILE.FOPEN('C:\temp\fichier','resultat.txt','w');
  UTL_FILE.PUT_LINE(ref_fichier,'Première ligne...');
  -- insérez ici des données provenant de la base...
  UTL_FILE.PUT_LINE(ref_fichier,'C'est fini...');
  UTL_FILE.FFLUSH(ref_fichier);
  UTL_FILE.FCLOSE(ref_fichier);
EXCEPTION
  -- traitement des erreurs rencontrées
  WHEN NO_DATA_FOUND THEN
    DBMS_OUTPUT.PUT_LINE ('Pas de donnée trouvée.');
```

```
UTL_FILE.FCLOSE(ref_fichier);
WHEN UTL_FILE.WRITE_ERROR THEN
  DBMS_OUTPUT.PUT_LINE ('UTL_FILE.WRITE_ERROR invalide. ');
UTL_FILE.FCLOSE(ref_fichier);
WHEN OTHERS THEN
  DBMS_OUTPUT.PUT_LINE ('Autre erreur. ');
UTL_FILE.FCLOSE(ref_fichier);
END; -- fin de la procedure
-- le signe / execute la procedure sous SQL*Plus
/
```

Test de la procédure

La création de la procédure s'effectue à partir de SQL*Plus. On l'exécute par une commande de type :

```
BEGIN
  test_utl_file;
END; -- le signe / execute la procedure sous SQL*Plus
/
```

Résultat : le fichier *C:\temp\fichier\resultat.txt* contient les lignes suivantes :

```
Première ligne...
C'est fini...
```

L'utilisation de cette méthode permet de créer très rapidement des pages HTML statiques contenant des données extraites des bases Oracle 10g.

Envoyer des e-mails depuis Oracle : UTL_MAIL

Le package PL/SQL UTL_MAIL remplace l'antique UTL_SMTP. Il permet à des fonctions ou procédures qui s'exécutent dans la base de données d'envoyer des e-mails. Il est ainsi facile d'organiser une campagne d'e-mails utilisant des données et des contacts conservés dans Oracle. Toutes vos applications peuvent enfin expédier des e-mails, avec des pièces jointes, directement depuis la base de données ! Combiné avec des triggers ou des procédures, cela ouvre un vaste champ de possibilités. Toutes les possibilités d'un e-mail sont accessibles : destinataire, copie, copie cachée, pièce jointe, accusé de réception.

Prérequis à l'utilisation de UTL_SMTP

Le package UTL_MAIL n'est pas installé par défaut car il nécessite que le serveur puisse envoyer des e-mails, ce qui peut être considéré comme une faille de sécurité. Le port de communication SMTP_OUT_SERVER qui doit être ouvert à cet effet sur le serveur doit être sécurisé. Pour installer UTL_MAIL :

```
sqlplus sys/mot_de_passe

SQL> @ORACLE_HOME\rdbms\admin\utlmail.sql
SQL> @ORACLE_HOME\rdbms\admin\prvmail.sql
```


SMTP_OUT_SERVER précise le serveur SMTP et le numéro de port que UTL_MAIL doit utiliser pour adresser des e-mails sortants. Comme les serveurs e-mail sont souvent redondants, plusieurs serveurs peuvent être précisés, séparés par des virgules. Dans ce cas, si le premier serveur est indisponible, le second est contacté et ainsi de suite.

SMTP_OUT_SERVER doit être précisé dans le fichier init.ora (fichier statique) ou le pfile (fichier dynamique) d'initialisation de la base.

Utilisation de UTL_MAIL

Les procédures UTL_MAIL.SEND, UTL_MAIL.SEND_ATTACH_RAW et UTL_MAIL.SEND_ATTACH_VARCHAR2 permettent d'envoyer un message simple, avec un texte attaché ou tout type de fichier binaire attaché.

Le package UTL_MAIL dispose de nombreuses options. Nous vous engageons à consulter sa documentation. Vous bénéficiez maintenant de toute la puissance du langage PL/SQL pour envoyer des messages. C'est une option très intéressante pour toutes vos applications.

Soumettre des travaux : DBMS_JOB

Le package DBMS_JOB permet de soumettre des travaux à une heure et une fréquence déterminée. C'est le principe des « schedulers » ou ordonnanceurs. Si vous connaissez des utilitaires comme *cron* (Linux, Unix) ou *at* (Windows), vous possédez déjà une vision d'ensemble de ce que propose DBMS_JOB.

L'intérêt de DBMS_JOB est d'automatiser les travaux exécutés au sein de la base Oracle 10g, quel que soit le système d'exploitation qui l'héberge. Pour les autres tâches ne concernant pas la base, telles que lancer des programmes de sauvegarde, manipuler des fichiers, redémarrer la machine, etc., vous pouvez aussi vous appuyer sur l'ordonnanceur offert par le système d'exploitation ou un produit tiers.

Oracle Enterprise Manager propose une interface graphique pour manipuler simplement DBMS_JOB.

Accéder aux données d'un annuaire LDAP : DBMS_LDAP

Les packages DBMS_LDAP et DBMS_LDAP_UTL permettent d'accéder aux données d'un annuaire LDAP. Cela permet, par exemple, de ne pas décrire les droits d'accès à une application dans l'application mais de s'appuyer sur une infrastructure existante (Microsoft Active, Open LDAP) pour autoriser l'accès à une application Oracle et en contrôler les droits d'usage.

Avec des simples ordres SQL, vous pouvez interroger, manipuler les données d'un annuaire LDAP.

Travailler avec des données binaires : DBMS_LOB

Les LOB (*Large Objects*) sont destinés à contenir des données binaires, jusqu'à quatre giga-octets par enregistrement. Avec Oracle7, les possibilités offertes pour manipuler de tels objets stockés dans la base étaient très réduites. Il fallait écrire des applications (en langage C) pour extraire et manipuler ces données.

Principes d'utilisation du package DBMS_LOB

Oracle 10g permet de créer des LOB internes ou externes. Cette possibilité est largement utilisée dans Oracle 10g, notamment par *Internet File System* (iFS), le système de fichiers Internet. Du point de vue de l'utilisateur, iFS se présente simplement comme un autre volume du système de fichiers en réseau. L'utilisateur accède à ses fichiers au moyen de l'explorateur Windows, d'un navigateur Web ou encore d'un client FTP ; les fichiers ont le même aspect. Du point de vue de la base Oracle 10g, iFS est une application Java qui manipule des objets LOB conservés à l'intérieur ou à l'extérieur de la base de données Oracle 10g. Cette application Java utilise massivement les possibilités offertes par le package DBMS_LOB pour manipuler ces objets binaires.

Le package DBMS_LOB permet de lire et de modifier les objets BLOB, CLOB et NLOB qui sont stockés dans la base Oracle 10g, et autorise uniquement la lecture sur les BFILES conservés hors de la base.

L'utilisation de LOB dans une base de données impose quelques limites d'utilisation. Par exemple, on ne peut pas créer d'index unique sur une colonne comportant un LOB. Ces limites sont peu contraignantes et facilement contournables.

Toutes les procédures et fonctions de DBMS_LOB se fondent sur l'utilisation de *locators*. Les locators doivent pointer vers des LOB déjà créés et existants dans votre base de données ou sur un fichier externe. Pour les opérations en amont de la recherche de colonnes LOB, d'insertion d'un LOB dans cette colonne, etc., il faut créer les tables contenant ces LOB et y insérer les données. En résumé, DBMS_LOB permet la manipulation d'objets binaires existants.

Syntaxe du package DBMS_LOB

Le tableau ci-dessous résume les principales procédures disponibles dans le package DBMS_LOB.

Les routines qui permettent de modifier des LOB, BLOB, CLOB sont répertoriées ci-après :

Fonction/Procédure	Description
APPEND	Remplace le contenu d'un LOB cible par un LOB source.
COPY	Copie complètement ou partiellement un LOB source dans un LOB cible.
ERASE	Efface complètement ou partiellement le contenu d'un LOB.
LOADFROMFILE	Permet de charger un BFILE externe à la base Oracle 10g dans un LOB interne à la base Oracle 10g.
TRIM	Réduit la taille du LOB à la valeur spécifiée.
WRITE	Force une écriture dans un LOB.

Les routines qui permettent de lire et examiner des LOB sont les suivantes :

Fonction/Procédure	Description
GETLENGTH	Récupère la taille du LOB.
INSTR	Recherche d'occurrences de chaînes binaires dans un LOB.
READ	Lecture complète d'un LOB après un « saut » d'une taille précisée.
SUBSTR	Lecture partielle d'un LOB. Un paramètre précise la taille à lire.

Voici les routines qui permettent de lire des BFILES (externes à la base Oracle 10g) :

Fonction/Procédure	Description
FILECLOSE	Ferme le fichier BFILE.
FILECLOSEALL	Ferme tous les fichiers BFILE ouverts.
FILEEXISTS	Vérifie l'existence d'un fichier BFILE.
FILEGETNAME	Récupère le répertoire et le nom du fichier.
FILEISOPEN	Vérifie si le fichier est déjà ouvert.
FILEOPEN	Ouvre un fichier BFILE.

Un BFILE est stocké à l'extérieur de la base *Oracle 10g*. Avant de l'utiliser, il faut créer un alias (en majuscules !) associé au répertoire du BFILE, au moyen de la commande SQL CREATE DIRECTORY. La fonction BFILENAME récupère le répertoire d'un BFILE. Ces éléments sont indispensables pour manipuler des fichiers BFILE.

Exemple d'une procédure utilisant DBMS_LOB

L'exemple suivant est un bloc PL/SQL anonyme, destiné à manipuler des LOB.

```
--
-- Création d'une table contenant une colonne de type LOB
--
CREATE TABLE test_lob (
  cle_primaire    NUMBER PRIMARY KEY,
  col_clob        CLOB,
  col_blob        BLOB,
  col_bfile       BFILE );
--
-- Insertion de données dans la table
--
-- Colonne CLOB
INSERT INTO test_lob (cle_primaire, col_clob )
VALUES( 1, ' Une phrase tout à fait normale.' );

-- Colonne BLOB
INSERT INTO test_lob (cle_primaire, col_blob )
VALUES( 2, HEXTORAW('01ABCFF12DD3'));
```

```
-- Colonne BFILE (conservé hors de la base Oracle !)  
-- * placez un fichier binaire (le fichier Word TEST.DOC par exemple)  
-- * dans le répertoire C:/TEMP  
  
--  
-- Création d'un alias pour le répertoire BFILE  
--  
CREATE DIRECTORY rep_bfile AS 'C:\temp' ;  
  
--  
-- Bloc PL/SQL destiné à manipuler des objets LOB  
--  
set serveroutput on  
  
DECLARE  
    LocatorClob      CLOB;  
    LocatorBlob      BLOB;  
    LocatorBfile     BFILE;  
    -- Buffer destinés à manipuler les données  
    BufferClob        VARCHAR2(255);  
    BufferBlob        RAW(32767);  
    BufferBfile       RAW(32767);  
  
    Offset           NUMBER;  
BEGIN  
    -- Les colonnes BLOB et CLOB sont remplies  
    -- dans la table test_lob.  
    -- Reste à récupérer le fichier BFILE pour  
    -- l'insérer dans la table test_lob.  
    INSERT INTO test_lob (cle_primaire, col_bfile )  
    VALUES( 3, BFILENAME('REP_BFILE','TEST.DOC') );  
  
    -- Récupération des 'locators' des données.  
    -- Seules les données CLOB et BLOB peuvent être modifiées.  
    -- Comme chaque ordre SELECT ne retourne qu'un seul  
    -- enregistrement, il n'est pas nécessaire d'utiliser  
    -- des curseurs.  
    SELECT col_clob INTO LocatorClob FROM test_lob  
    WHERE cle_primaire = 1 FOR UPDATE ;  
  
    SELECT col_blob INTO LocatorBlob FROM test_lob  
    WHERE cle_primaire = 2 FOR UPDATE ;  
  
    SELECT col_bfile INTO LocatorBfile FROM test_lob  
    WHERE cle_primaire = 3 ;  
  
    -- Taille de la donnée CLOB:  
    Offset := 1 + DBMS_LOB.GETLENGTH(LocatorClob) ;  
  
    -- Lecture de la donnée CLOB et récupération dans un buffer:  
    DBMS_LOB.READ(LocatorClob, Offset, 1, BufferClob);
```

```

-- Affichage de la donnée textuelle :
DBMS_OUTPUT.PUT_LINE('**** Donnée CLOB *****');
DBMS_OUTPUT.PUT_LINE('Taille CLOB : '||Offset||' octets');
DBMS_OUTPUT.PUT_LINE('Texte CLOB : '||BufferClob);
DBMS_OUTPUT.NEW_LINE;

-- Ajout de texte en fin du CLOB:
DBMS_LOB.WRITE(LocatorClob,27,Offset+1,'...Texte ajouté au CLOB !!!');
Offset := 1 + DBMS_LOB.GETLENGTH(LocatorClob) ;
DBMS_LOB.READ(LocatorClob, Offset, 1, BufferClob);
DBMS_OUTPUT.PUT_LINE('**** Modification du texte CLOB *****');
DBMS_OUTPUT.PUT_LINE('Taille CLOB : '||Offset||' octets');
DBMS_OUTPUT.PUT_LINE('Texte CLOB : '||BufferClob);
DBMS_OUTPUT.NEW_LINE;

-- On peut manipuler la donnée BLOB de la même façon
-- mais il n'est pas possible de l'afficher ...

-- Manipulons le fichier BFILE extérieur à la base Oracle.
DBMS_LOB.FILEOPEN(LocatorBfile);
Offset := 1 + DBMS_LOB.GETLENGTH(LocatorBfile) ;
DBMS_OUTPUT.PUT_LINE('**** Fichier BFILE *****');
DBMS_OUTPUT.PUT_LINE('Taille BFILE : '||Offset||' octets');

-- Lecture du fichier BFILE :
DBMS_LOB.READ(LocatorBfile, Offset, 1, BufferBfile);
-- Attention, la taille de votre fichier ne doit pas dépasser
-- celle du Buffer la contenant...
-- impossible d'afficher ce fichier sous SQL*Plus...

-- Ne pas oublier de fermer le fichier BFILE :
DBMS_LOB.FILECLOSE(LocatorBfile);

-- C'est fini...
DBMS_OUTPUT.PUT_LINE('Fin normale du test...');

EXCEPTION
  WHEN OTHERS THEN
    DBMS_OUTPUT.PUT_LINE('..erreur rencontrée...');
    DBMS_LOB.FILECLOSE(LocatorBfile);
END;
/

--
-- Nettoyage
--
DROP TABLE test_lob ;
DROP DIRECTORY rep_bfile ;

```

Test du script précédent

Saisissez le script précédent et exécutez-le sous SQL*Plus. Vous obtenez ce qui suit :

```
Table created.
1 row created.
1 row created.
Directory created.

**** Donnée CLOB ****
Taille CLOB : 32 octets
Texte CLOB : Une phrase tout à fait normale.
**** Modification du texte CLOB ****
Taille CLOB : 59 octets
Texte CLOB : Une phrase tout à fait normale....Texte ajouté au CLOB !!!
**** Fichier BFILE ****
Taille BFILE : 2199 octets
Fin normale du test...

PL/SQL procedure successfully completed.

Table dropped.
Directory dropped.
```

Les limites de DBMS_SQL

DBMS_SQL apporte une grande souplesse dans la manipulation des données binaires. Il n'en demeure pas moins quelques limites liées à la taille des objets. En effet, les variables PL/SQL d'accueil des données de type LOB sont d'une taille réduite, limitée à 32 767 octets. Si l'on souhaite manipuler des données plus volumineuses, PL/SQL n'est plus le langage adéquat : il faut avoir recours à un langage de type C qui utilise des routines d'accès à la base Oracle 10g, identiques à celles proposées par DBMS_LOB. Les limites des variables d'accueil sont alors celles du langage C. Néanmoins, pour le traitement d'objets d'une taille inférieure à 32 767 octets, DBMS_LOB est d'une souplesse remarquable.

Générer des scripts de création des objets : DBMS_METADATA

Le package DBMS_METADATA (depuis Oracle9i) permet de générer les scripts de création des objets contenus dans la base de données.

Par exemple, pour obtenir les requêtes de création de toutes les tables du schéma utilisateur courant :

```
select DBMS_METADATA.GET_DDL('TABLE', u.table_name)
from USER_ALL_TABLES u
where u.nested='NO'
and (u.iot_type is null or u.iot_type='IOT') ;
```

L'habillage au sein d'un ordre SQL pour récupérer le script dans un fichier serait :

```
# Purge de l'exécution précédente
begin
DBMS_METADATA.SET_TRANSFORM_PARAM (DBMS_METADATA.SESSION_TRANSFORM, 'STORAGE', false);
end ;
/
# paramétrage de l'environnement SQL*Plus pour
# obtenir une édition « propre »
set long 30000
set echo off
set heading off
set linesize 1000
set trimspool on
spool c:\temp\sql\creation_table.sql
# écrit dans le fichier spool l'ordre de création des tables
select DBMS_METADATA.GET_DDL('TABLE', u.table_name)
from USER_ALL_TABLES u
where u.nested='NO'
and (u.iot_type is null or u.iot_type='IOT') ;
begin
DBMS_METADATA.SET_TRANSFORM_PARAM (DBMS_METADATA.SESSION_TRANSFORM, 'DEFAULT') ;
end ;
/
```

Le package DBMS_METADATA autorise aussi de nombreuses manipulations des schémas au format XML.

Construire dynamiquement des ordres SQL : DBMS_SQL

Dans tous les exemples précédents, seuls des ordres SQL de type langage de manipulation de données (DML ou *Data Manipulation Language*) ont été utilisés. Les syntaxes abordées ne permettent pas d'écrire un ordre SQL avec le nom de la table accédé transmis sous forme de paramètre. Pour écrire des programmes contenant des commandes utilisant le langage de définition de données (DDL ou *Data Definition Language*), il faut obligatoirement utiliser le package DBMS_SQL.

En effet, tout bloc PL/SQL est compilé sous forme de pseudo-code (p-code) avant son exécution. Cette phase de compilation vérifie si les ordres SQL sont corrects, s'il n'y a pas d'erreur de syntaxe, si les variables d'échange de données sont de même type, etc. Le pseudo-code permet d'obtenir de bien meilleures performances, mais il est nécessaire de connaître le nom des tables, des colonnes avant l'étape de compilation. Grâce à DBMS_SQL, il est possible de s'affranchir de ces limites en fournissant une méthode de compilation des blocs PL/SQL, en reportant certaines phases lors de l'exécution du bloc, quand toutes les variables sont connues.

Utilisation du package DBMS_SQL

Pour utiliser DBMS_SQL, les étapes suivantes sont nécessaires :

1. Placez l'ordre SQL à exécuter dans une chaîne de caractères.

2. Ouvrez un curseur avec l'ordre OPEN_CURSOR.
3. La commande PARSE permet d'analyser le bloc sans la chaîne.
4. BIND_VALUE permet d'assigner des valeurs à des variables.
5. Pour les ordres autres que des SELECT, exécutez l'ordre SQL par EXECUTE ou encore VARIABLE_VALUE puis fermez le curseur avec CLOSE_CURSOR.
6. Pour les ordres SELECT, DEFINE_COLUMN permet de définir les colonnes résultat de l'interrogation.
7. EXECUTE lance l'interrogation, FETCH_ROW, COLUMN_VALUE et VARIABLE_VALUE recueillent les données.
8. CLOSE_CURSOR ferme le curseur.

Syntaxe du package DBMS_SQL

Le tableau suivant résume les principales procédures disponibles dans le package DBMS_SQL. D'autres procédures existent pour traiter des données de type LONG ou encore manipuler des tableaux sous forme d'ARRAY.

Fonction/Procédure	Description
OPEN_CURSOR	Ouvre un curseur et retourne un identifiant.
PARSE	Analyse syntaxique l'ordre SQL.
BIND_VARIABLE	Affecte un identifiant à une variable.
DEFINE_COLUMN	Détermine pour les ordres SELECT les colonnes qui seront visualisées.
EXECUTE	Exécute le curseur choisi.
FETCH_ROWS	Retourne les enregistrements d'un ordre SELECT.
IS_OPEN	Teste le statut d'un curseur.
DESCRIBE_COLUMNS	Décrit les colonnes d'un ordre SQL contenu dans un curseur ouvert et « parsé » (analyse sémantique).
CLOSE_CURSOR	Ferme le curseur et libère les ressources.

Exemples d'utilisation du package DBMS_SQL

Cet exemple permet d'exécuter un ordre SQL qui est transmis comme paramètre à la procédure. Les ordres de type SELECT ne sont pas admis dans ce cas.

```
--  
-- Exemple: Exécution d'un ordre SQL indéfini  
--  
-- L'ordre SQL est transmis comme paramètre à la procédure.  
--  
CREATE OR REPLACE PROCEDURE ordre_sql (v_ORDRE_SQL IN varchar2) AS  
    mon_curseur    integer;  
    v_retour        integer;
```



```
BEGIN
    mon_curseur := DBMS_SQL.OPEN_CURSOR ;
    --
    -- L'ordre SQL est exécuté lors de la phase de PARSE.
    -- Le COMMIT est implicite après le PARSE.
    --
    DBMS_SQL.PARSE(mon_curseur,v_ordre_sql,dbms_sql.v7);
    v_retour := DBMS_SQL.EXECUTE(mon_curseur);
    DBMS_SQL.CLOSE_CURSOR(mon_curseur);
END;
/
-- utiliser la ligne suivante en phase de deboguage
-- show error

-- Test de la procédure
begin
    ordre_sql('create table Oracle9 (colonneA varchar2(10) )');
end;
/

-- suppression de la table Oracle9
execute ordre_sql ( 'drop table Oracle9');
```

La syntaxe et les utilisations possibles de DBMS_SQL sont très riches. Nous vous conseillons de consulter la documentation Oracle 10g.

Communiquer entre processus : DBMS_ALERT et DBMS_PIPE

DBMS_ALERT et DBMS_PIPE font partie de la famille d'outils mis à votre disposition par Oracle pour communiquer entre processus. Ce sont des outils rudimentaires, souvent anciens. Le véritable outil de communication inter-processus est DBMS_AQ, beaucoup plus riche fonctionnellement.

DBMS_ALERT

DBMS_ALERT permet d'envoyer un message vers plusieurs sessions utilisateur. Il s'agit d'une communication unidirectionnelle. Pour lire le message envoyé, il faut interroger l'alerte.

C'est l'ordre COMMIT qui effectue l'envoi d'une alerte. Si un ROLLBACK intervient, aucune alerte n'est envoyée.

Plusieurs alertes peuvent être envoyées les unes à la suite des autres. Lorsque l'utilisateur interrogera l'alerte, il ne lira que la dernière.

Lorsqu'un utilisateur scrute une alerte, cela ne la supprime pas. Plusieurs utilisateurs peuvent donc interroger la même alerte.

DBMS_PIPE

DBMS_PIPE permet d'envoyer un message vers une seule session utilisateur : c'est une communication unidirectionnelle. Pour lire le message adressé, il faut interroger le *pipe* (pipe signifie « canal de communication » comme pipeline).

Dès qu'une information est expédiée sur un *pipe*, les utilisateurs peuvent interroger le pipe pour recevoir le message.

Plusieurs messages peuvent être envoyés les uns à la suite des autres. Lorsque l'utilisateur scrute le pipe, il recevra l'ensemble des messages.

À la différence de l'alerte, le pipe n'est pas protégé par un mécanisme de COMMIT. Les informations envoyées sur un pipe sont conservées dans la mémoire allouée à la base Oracle 10g. Lorsque celle-ci est fermée ou que le serveur est relancé, l'ensemble des messages reçus sur les pipes est perdu. Cela limite fortement l'utilisation.

Oracle Advance Queuing : DBMS_AQ

DBMS_AQ est un package extrêmement riche qui permet de construire des applications sophistiquées. Il se fonde sur des mécanismes de file d'attente de messages ; il a été construit en partie pour répondre à des besoins afférents aux modules d'Oracle Application (logiciel couvrant tous les domaines de l'entreprise paie, comptabilité, gestion, stocks...).

DBMS_AQ intervient dans des applications complexes, dont la logique de traitement consiste à enchaîner des tâches. C'est le cas de nombreuses applications dans l'entreprise. Ce package permet de faire circuler des informations, de les lier à des actions tout en contrôlant l'ensemble des opérations. C'est le principe du *workflow*.

Les messages sont, envoyés dans une file d'attente et y sont conservés. Ils sont ensuite traités de manière asynchrone ou synchrone. Le principe de traitement asynchrone est largement utilisé dans l'industrie informatique, le fonctionnement des systèmes d'exploitation ou encore des moniteurs transactionnels y font largement appel.

Placer des messages dans une file d'attente permet de gérer les conflits et d'arbitrer les priorités de traitement. Il est aussi possible de regrouper les messages d'un même type pour en accélérer le traitement.

Nous ne détaillerons pas davantage le fonctionnement de DBMS_AQ, que vous retrouverez, si nécessaire, dans la documentation Oracle 10g.

La bibliothèque DBMS_AQ permet de construire des applications de type workflow au sein de la base de données, avec toutes les garanties offertes par la portabilité, la stabilité et les possibilités de sauvegarde d'une base Oracle 10g.

Oracle Log Miner : DBMS_LOGMNR

Le Log Miner permet d'exploiter le contenu des fichiers redo-log pour les transcrire en une suite d'ordres SQL. Vous pouvez extraire de cette suite d'ordres SQL ceux qui ont

porté sur une table particulière ou qui ont été effectués par un utilisateur identifié. Le point a été abordé au chapitre 23, *Gestion de l'espace disque et des fichiers*.

Pour réaliser cette analyse, Log Miner exploite les fichiers redo-log et en affiche le contenu sous forme de texte et de valeurs hexadécimales. La présence du dictionnaire de la base permet de traduire les données hexadécimales en valeurs textuelles compréhensibles.

L'utilisation de Log Miner repose sur les packages PL/SQL DBMS_LOGMNR et DBMS_LOGMNR_D. Ils permettent de commencer une session Log Miner, d'en préciser la portée, le champ d'étude et enfin de terminer la session. Les vues V\$LOGMNR_CONTENTS et V\$LOGMNR* permettent d'interroger et de filtrer les éléments retournés.

Récupérer le schéma des objets : DBMS_METADATA

Le package DBMS_METADATA permet d'extraire depuis une base de données existante la définition de ses objets. Vous pouvez aussi utiliser Oracle Enterprise Manager à cet effet, mais le package DBMS_METADATA offre des possibilités de programmation et de filtrage.

Un regroupement d'utilitaires : DBMS_UTILITY

Le package DBMS_UTILITY regroupe tout un ensemble de procédures et fonctions utiles. Parmi elles figurent :

- COMPILE_SCHEMA : automatise la compilation de tous les objets d'un schéma ;
- ANALYZE_SCHEMA : automatise l'analyse de tous les objets d'un schéma ;
- GET_TIME : permet de mesurer le temps écoulé entre deux mesures ;
- GET_PARAMETER_VALUE : récupère les valeurs fixées dans le fichier *init.ora*, même si vous n'avez pas les droits d'accès à la vue V\$PARAMETER ;
- DB_VERSION : numéro de version de la base de données.

Manipuler le XML : DBMS_XML

Oracle propose tout un ensemble de packages pour manipuler le XML : DBMS_XMLPARSER, DBMS_XMLQUERY, DBMS_XMLSCHEMA, DBMS_XSPPROCESSOR, etc. En liaison avec XML DB, ce sont de très puissants outils de manipulation, d'interrogation et de transformation de données XML. Ils peuvent vous faire gagner un temps précieux dans vos applications.

Manipuler des données géolocalisées : SDO

De nombreuses données peuvent faire l'objet de données de géolocalisation : des routes, bâtiments, clients, etc. La géolocalisation permet de répondre aux questions du type : quels sont les clients situés dans un rayon de 15 km de mon agence ? Vous pouvez aussi

utiliser ces données pour les restituer sur des cartes. C'est ce que proposent des outils tiers comme MAP Server, MAPINFO, Essbase, etc. Des projets à base de logiciels libres existent aussi dans ce domaine.

Les triggers

Oracle 10g vous permet de définir des procédures, appelées *triggers*, de la base de données. Ces procédures sont associées à une table et sont exécutées implicitement lorsque vous lancez des ordres SQL de type insertion, modification, suppression (INSERT, UPDATE, DELETE).

Principes d'utilisation des triggers

Les triggers permettent d'exécuter automatiquement des procédures centralisées dans la base de données, quel que soit l'utilisateur ou l'application qui les déclenche. Ils sont souvent utilisés pour construire des interfaces interapplications et s'avèrent très simples et très souples dans leur maniement.

Les triggers se déclenchent sur des actions modifiant les données de la table sur laquelle porte le trigger. Si cette table est accessible en lecture uniquement, aucun trigger ne sera déclenché.

N'utilisez pas les triggers pour effectuer des traitements trop complexes, qui nuiraient aux performances de votre base Oracle 10g en cas de modification des données. Préférez des triggers simples, qui appellent des procédures plus complexes.

N'utilisez pas les triggers pour effectuer des contrôles d'intégrité des données qui peuvent être réalisés par des contraintes d'intégrité. Ces contraintes positionnées au niveau des tables sont puissantes et très optimisées.

Veillez à ne pas créer de triggers fonctionnant « en boucle », s'exécutant sans contrôle possible. La seule possibilité est alors de supprimer la session Oracle qui rencontre cette erreur.

Exemple d'écriture d'un trigger

Dans l'exemple suivant, le trigger sert à constituer un fichier historique de toutes les modifications apportées au salaire des employés :

```
SQL> --
SQL> -- Création de la table trace
SQL> --
SQL>
SQL> CREATE TABLE trace_salaire (
  2  empno NUMBER(5),
  3  ancienNUMBER(7,2),
  4  nouveauNUMBER(7,2),
  5  dat_modifDATE,
```

```
6 qui VARCHAR2(30),
7 commentaireVARCHAR2(20) )
8 ;
```

Table created.

```
SQL> --
SQL> -- Création du trigger sur la table des employés
SQL> --
SQL> CREATE OR REPLACE TRIGGER trig_trace_salaire
2 AFTER INSERT OR UPDATE OR DELETE ON emp
3 FOR EACH ROW
4 DECLARE
5     quand DATE := SYSDATE ;
6 BEGIN
7     -- insertion d'un employé
8     IF INSERTING THEN
9         INSERT INTO trace_salaire
10 VALUES(:new.empno,null,:new.sal,quand,user,'NOUVEAU' );
11     -- si il y a modification du salaire
12     ELSIF UPDATING ('SAL') THEN
13         INSERT INTO trace_salaire
14 VALUES(:old.empno,:old.sal,:new.sal,quand,user,'MODIFIE');
15     -- suppression d'un employé
16     ELSIF DELETING THEN
17         INSERT INTO trace_salaire
18 VALUES(:old.empno,:old.sal,null,quand,user,'SUPPRIME');
19     END IF;
20 END;
21 /
```

Trigger created.

```
SQL> show error
No errors.
```

```
SQL> --
SQL> -- Test du trigger
SQL> --
SQL> -- Création d'un nouvel employé
SQL> insert into emp values (123,'GILLES','CLERK',null,sysdate,1000,0,10);
```

1 row created.

```
SQL>
SQL> -- Modification de cet employé
SQL> update emp set sal = sal*2 where empno = 123 ;
```

1 row updated.

```
SQL>
SQL> -- Suppression de l'employé
```

```
SQL> delete from emp where empno = 123 ;
```

```
1 row deleted.
```

```
SQL>
```

```
SQL> -- Vérification dans la table trace_salaire
```

```
SQL> select * from trace_salaire ;
```

EMPNO	ANCIEN	NOUVEAU	DAT_MODIF	QUI	COMMENTAIRE
123		1000	09-FEB-99	SCOTT	NOUVEAU
123	1000	2000	09-FEB-99	SCOTT	MODIFIE
123	2000		09-FEB-99	SCOTT	SUPPRIME

La table trace contient ainsi une trace de toutes les modifications apportées à la table des employés.

Comment créer et gérer un trigger ?

Un trigger est créé par la clause CREATE TRIGGER ou CREATE OR REPLACE TRIGGER. Un trigger existant peut être détruit par DROP TRIGGER.

Il est à noter qu'un trigger existant peut temporairement être activé ou désactivé :

```
ALTER TRIGGER nom_du_trigger ENABLE ; -- active le trigger
ALTER TRIGGER nom_du_trigger DISABLE; -- désactive le trigger
```

Cette opération peut être étendue en un seul ordre SQL à l'ensemble des triggers qui portent sur une table :

```
ALTER TABLE table_ou_sont_des_triggers DISABLE ALL TRIGGERS ;
ALTER TABLE table_ou_sont_des_triggers ENABLE ALL TRIGGERS ;
```

Quand se déclenchent les triggers ?

Le type d'action qui déclenche le trigger détermine son moment d'exécution ; il est précisé au début de l'écriture du code du trigger.

Les actions sont les suivantes :

- INSERT ;
- UPDATE ;
- DELETE.

Le trigger peut être déclenché avant ou après cette action : BEFORE ou AFTER.

Dans l'exemple, il est prévu que le trigger se déclenche après toute action modifiant les données de la table des employés :

```
...
AFTER INSERT OR UPDATE OR DELETE ON emp
...
```

La clause *FOR EACH ROW*

Pour un ordre SQL modifiant plusieurs enregistrements simultanément, la clause *FOR EACH ROW* détermine le déclenchement du trigger pour chaque enregistrement ou pour l'ensemble des lignes modifiées.

Dans l'exemple, même si l'ordre SQL lancé sur la table des employés modifie plusieurs enregistrements à la fois, le trigger sera exécuté pour chacun d'entre eux.

Les conditions d'exécution

À l'intérieur du code du trigger, vous pouvez contrôler avec précision les conditions d'exécution. Dans l'exemple, les clauses :

```
...  
IF INSERTING THEN  
...  
ELSIF UPDATING ('SAL') THEN  
...  
ELSIF DELETING THEN  
...
```

permettent d'effectuer des traitements différents suivant l'action qui a déclenché le trigger. On peut même, dans le cas de la mise à jour d'un enregistrement, n'effectuer un traitement que si une colonne spécifique a été modifiée :

```
...  
ELSIF UPDATING ('SAL') THEN  
...
```

Accéder aux valeurs d'un enregistrement dans un trigger

Un trigger traitant des enregistrements en cours de modification, il est possible d'accéder à la valeur des données avant et après modification. Pour chaque colonne de la table, vous pouvez utiliser les variables suivantes :

```
:old.nom_de_la_colonne  
:new.nom_de_la_colonne
```

Cela permet, à l'intérieur du trigger, de manipuler les valeurs modifiées :

```
IF :old.sal > 5000 ...  
IF UPPER(:new.nom) = 'GILLES'...
```

Les valeurs *new* et *old* apparaissent en fonction du type d'action. Dans le cas d'une insertion (*INSERT*), seule la valeur *new* existe, la valeur *old* n'ayant pas de sens pour un enregistrement en cours de création. Pour une modification (*UPDATE*), les valeurs *new* et *old* co-existent, alors que pour une suppression (*DELETE*), on ne retrouvera que la valeur *old*.

Par exemple :

```
...  
IF UPDATING THEN  
  INSERT INTO trace_salaire  
  VALUES(:old.empno,:old.sal,:new.sal,quand,user,'SUPPRIME');  
...
```

Peut-il y avoir plusieurs triggers sur une table ?

La réponse est oui. Vous pouvez créer, sur une même table, un trigger avant mise à jour, un autre après mise à jour, etc. Nous vous conseillons de ne pas créer trop de triggers sur une même table, car la maintenance est alors rendue difficile.

Comment visualiser les triggers portant sur une table et leur code ?

Différentes vues du dictionnaire de données Oracle 10g traitent des triggers. Vous pouvez y accéder en fonction des droits dont vous disposez :

- USER_TRIGGERS ;
- USER_TRIGGER_COLS ;
- ALL_TRIGGERS ;
- ALL_TRIGGER_COLS ;
- DBA_TRIGGERS ;
- DBA_TRIGGER_COLS.

Résumé

Ce chapitre traite de PL/SQL. Dans la stratégie Oracle, ce langage de programmation simple permet d'exploiter tout le potentiel d'Oracle 10g. Nous avons détaillé les éléments de base : la syntaxe, la programmation des procédures, les fonctions, les packages et les triggers. D'autres éléments, comme les packages standards, ont été abordés plus rapidement.

L'objectif de ce chapitre était de vous montrer l'importance fondamentale de PL/SQL. Les procédures, les fonctions, les packages et les triggers sont bien souvent ignorés et sous-employés, au profit de développements lourds et complexes.

PL/SQL vous offre une souplesse et une puissance de programmation que vous auriez tort de négliger.

Les outils d'Oracle 10g

Dans ce chapitre :

- présentation des outils d'Oracle 10g ;
- Oracle Migration Workbench, pour migrer des bases vers Oracle ;
- Oracle Export/Import ;
- Oracle Data Pump ;
- Oracle SQL*Loader ;
- Oracle SQL*Plus ;
- utilitaires complémentaires aux outils Oracle.

Depuis des années, chaque version d'Oracle est livrée avec un ensemble d'outils standard. Ils sont puissants et performants et on les retrouve sur toutes les machines où Oracle 10g existe. Livrés gratuitement, ils rendent l'utilisation de produits complémentaires facultative. Avant de les étudier séparément, nous vous les présentons rapidement.

Présentation des outils d'Oracle 10g

Oracle Migration Workbench n'est pas installé avec Oracle. C'est un « kit de migration » permettant de migrer toutes sortes de bases de données vers Oracle. Vous pouvez ainsi migrer des bases Access, Informix, DB2, Sybase, SQL Server, etc. vers Oracle. Vous pouvez télécharger gratuitement le kit concernant le fournisseur et la version exacte de la base à migrer vers Oracle à partir du site <http://technet.oracle.com>. Nous ne développerons pas plus cet utilitaire dans ce chapitre, mais il était important d'en mentionner l'existence.

Export/Import sont deux utilitaires qui fonctionnent l'un avec l'autre. Le premier permet d'exporter une partie ou la totalité du contenu d'une base Oracle 10g, le second importe la partie exportée dans une base de données Oracle 10g. Export/Import sont principalement utilisés lors de sauvegardes, de réorganisations et de déplacements du contenu d'une base Oracle 10g de machine à machine. La portabilité des fichiers exportés permet à un Export réalisé sur une base Oracle 10g fonctionnant sous Windows, d'être importé dans une base Oracle 10g sous Linux et inversement.

Un nouvel utilitaire d'Oracle 10g, Oracle Data Pump, reprend les mécanismes d'import/export. Il offre de meilleures performances qu'import/export et procure plus de souplesse, notamment la possibilité d'interrompre et de reprendre un Data Pump Export ou un Data Pump Import. Oracle Data Pump ne permet des échanges qu'entre des bases 10g, alors que l'import/export « classique » permet aussi de transférer des données entre des versions différentes d'Oracle.

SQL*Loader est un utilitaire très puissant de chargement des bases Oracle, à partir de fichiers contenant des données. Le format des données contenues dans les fichiers et leur destination dans la base cible sont très facilement paramétrables. SQL*Loader est fréquemment utilisé lors de l'alimentation de bases d'infocentre pour charger, dans Oracle, des données extraites d'autres bases.

SQL*Plus est l'interface mode caractère d'accès à Oracle 10g ; certaines versions, comme iSQL*Plus, peuvent se présenter sous forme semi-graphique via un accès Web. Cet outil est indispensable à tout développeur ou tout administrateur Oracle 10g, car c'est l'interface SQL privilégiée pour accéder aux bases Oracle 10g.

SQL*Plus permet maintenant de créer, démarrer, arrêter, administrer et effectuer des opérations complexes de restauration sur les bases Oracle. Il remplace l'ancien Server Manager qui assurait ces fonctions. SQL*Plus devient donc l'interface privilégiée d'Oracle par laquelle la totalité des ordres SQL concernant l'administration et l'utilisation d'Oracle peuvent être exécutés. Oracle Enterprise Manager est un outil graphique qui reprend une grande partie des possibilités d'administration offertes par SQL*Plus. Pour plus de détails à son sujet, reportez-vous au chapitre 25, *Oracle Enterprise Manager*.

Outre les outils standard, d'autres sont dédiés au développement et à l'administration. Ils sont proposés par Oracle ou par d'autres sociétés. Nous vous en présentons quelques-uns en distinguant les principales familles.

Oracle Export / Oracle Import

Ce paragraphe traite de l'utilisation pratique de l'utilitaire d'Export/Import. C'est un outil fort apprécié et pourtant négligé lors des formations Oracle. Ses principales utilisations sont les suivantes :

- le déplacement du contenu d'une base vers une autre (regroupement, migrations...) ;
- des sauvegardes partielles.

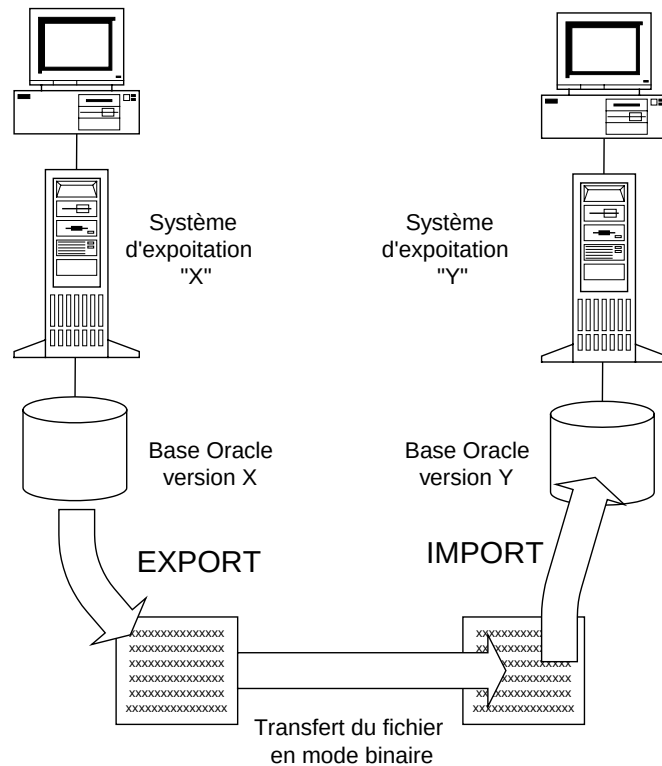
Principes de l'Export/Import

Le principe de l'Export/Import est d'extraire des données d'une base Oracle dans un format indépendant de tout système d'exploitation et de les réintégrer ailleurs, pour ce qui concerne l'Import. Cet objectif se réalise en deux étapes :

- Extrayez tous les objets (tablespaces, tables, index, procédures...) de la base de données. Ce sont les vues interrogeant le dictionnaire de la base Oracle qui réalisent cette fonction. Les objets sont extraits dans un ordre particulier.
- Extrayez les données contenues dans les tables. Ce sont des ordres SELECT qui interrogent les tables et enregistrent les données dans un format d'Export particulier.

Figure 18-1

Principe de l'Export/Import



L'opération d'Export est graduelle. Elle permet de transférer, par étapes, le contenu de la base :

- full : exportation complète de la base ;
- user : exportation de l'ensemble des objets propriétés d'un utilisateur ;
- table : exportation d'une table particulière.

À l'inverse, l'Import insère le contenu d'un Export dans une base de données. De même que l'Export, il fonctionne suivant plusieurs modes :

- full : importation complète de tous les objets présents dans le fichier export en entrée ;
- user : importation de l'ensemble des objets propriétés d'un utilisateur ;
- table : importation d'une table particulière.

L'Export/Import est-il un outil de sauvegarde ?

Force est de reconnaître que l'on ne peut pas considérer l'Export/Import comme un moyen de sauvegarde fiable à 100%. En effet, dans certains cas, vous pouvez effectuer une exportation complète sans exporter la totalité du contenu de la base. Par exemple, si l'un des tablespaces est inactif (OFFLINE), aucun des objets de ce tablespace ne sera exporté. L'exportation sera partielle au vu du contenu global de la base, ce qui peut être grave.

L'Export/Import est complémentaire des sauvegardes classiques, c'est-à-dire de la totalité des fichiers physiques qui constituent la base de données. Ce type de sauvegarde très puissant peut être étendu et inclure les journaux d'activité (archivage des redo-log). L'Export/Import est intéressant si, par exemple, une table et ses données sont supprimées par erreur. Il est alors possible d'importer cette table (dont les données sont à J-1 si l'exportation a eu lieu la veille), sans perturber la production ni arrêter la base.

Avant de choisir une stratégie de sauvegarde, reportez-vous au chapitre 26, *La sauvegarde d'une base Oracle 10g*.

Où se situe le jeu d'essai de l'Export/Import ?

À la différence de SQL*Loader, PL/SQL, Pro*C, il n'y a pas de fichier d'exemple livré par Oracle concernant l'Export/Import.

Configuration préalable

Pour rechercher dans le dictionnaire de données Oracle l'ensemble des objets existants dans la base, l'Export s'appuie sur des vues. Elles sont créées par le script `catexp.sql`, sous le compte utilisateur SYS (propriétaire du dictionnaire de la base). Il est exécuté durant la création de la base de données, à la fin de l'exécution du script `catalog.sql`, qui crée le catalogue de la base. Toutefois, le script `catexp.sql` peut être lancé manuellement, quand vous le souhaitez. Malgré son nom, (*catalog export*) il est aussi utilisé par le logiciel d'Import.

Si le script de création des vues n'est pas exécuté, l'Export/Import cherchera à accéder à des vues inexistantes et générera une erreur.

Les programmes d'Export/Import sont indépendants de la base de données. Ils peuvent être lancés en client-serveur pour accéder à des bases locales ou distantes. Sous Windows, les exécutables se nomment `exp` (Export) et `imp` (Import). Ils sont situés dans `ORACLE_HOME\bin` soit généralement `C:\oracle\product\10.1.0\db_1\BIN`.

Compatibilité des Export/Import avec les précédentes versions d'Oracle

Puisque les exécutables d'Export/Import sont des outils clients, ils ne se situent pas forcément sur la même machine et au même niveau de version que la base de données qu'ils interrogent. C'est pourquoi certaines versions d'Export/Import incorporent dans leur nom le numéro de la base de données à laquelle ils peuvent accéder (imp, exp, imp7, exp7...).

Une version prévue pour exporter des données d'une base Oracle8i permet, par rapport à la version Oracle7, d'exporter les nouvelles possibilités de cette base (extensions objet, partitionnement, etc.). Réaliser l'exportation complète d'une base Oracle8i avec la version Export d'Oracle7 conduit à une exportation incomplète, ce qui est grave.

En conséquence, il sera toujours plus facile de réaliser une exportation sur une base Oracle7 et d'en importer le contenu dans une base Oracle 10g. Dans ce cas, la compatibilité ascendante est assurée. Dans le cas d'une exportation Oracle 10g importée dans une base Oracle7, des difficultés peuvent survenir si vous avez utilisé dans la nouvelle version des possibilités inexistantes dans l'ancienne. Ces difficultés se traduisent par des erreurs lors des opérations.

Pour éviter tout inconvénient, la règle de base est d'utiliser la même version d'Export/Import que celle livrée avec la base de données. Réservez exclusivement ces opérations aux serveurs qui hébergent vos bases.

Si vous devez exporter vos données d'une base Oracle7, 8, 9i pour les importer dans une base Oracle 10g, utilisez la version correspondante pour l'exportation et la version 10g pour l'importation. Ce cas est fréquemment rencontré lors de l'évolution des versions Oracle.

Que contient un fichier export ?

Un fichier export est d'une taille beaucoup plus réduite que celle de la base de données exportée. Il ne contient que les ordres SQL de création des index, rollback segments, tablespaces, utilisateurs, droits, etc. Pour les autres objets, il contient à la fois leur définition, leur données et le code des programmes (tables, triggers, fonctions, etc.). Cette technique fait gagner beaucoup d'espace disque par rapport au volume de l'ensemble des fichiers qui composent la base de données.

Le format interne d'un fichier export n'est pas lisible facilement. Il faut toujours passer par les programmes d'Export/Import pour en manipuler le contenu. Nous vous déconseillons d'utiliser des éditeurs pour en visualiser le contenu. En effet, leur codage interne est binaire afin d'éviter les problèmes de conversion lors d'un changement de système d'exploitation.

Comment transférer des fichiers Export ?

Il faut toujours utiliser un outil (par exemple ftp) qui permet de transférer les fichiers d'Export de machine à machine sous forme binaire. Cette erreur (transfert en mode texte et non binaire) fréquente se manifeste par l'impossibilité de charger le fichier dans la base cible.

Attention : un fichier export peut être l'un des éléments de sauvegarde de votre base de données. Transférez-le en mode binaire et soyez prudent avant de supprimer les fichiers Export d'origine : assurez-vous qu'ils n'ont pas été corrompus ou transformés par leur transfert.

Comment lancer un Export/Import ?

Vous pouvez lancer un Export/Import par la commande `exp` (ou `imp`), suivie du nom d'utilisateur et du mot de passe. Par exemple :

```
exp SCOTT/TIGER
```

Dans ce cas, l'Export propose un mode interactif et vous guide pour les opérations à effectuer. Vous pouvez aussi commander l'Export en précisant dans sa ligne de commande le paramètre piloté et sa valeur :

```
exp PARAMETRE=valeur ou PARAMETRE=(valeur1, valeur2, ..., valeurN)
```

Par exemple :

```
exp userid=scott/tiger file=14_02_2005.dmp tables=(emp,dept)
```

Le mode interactif ne propose pas tous les paramètres possibles. Nous vous conseillons d'utiliser le passage de paramètres qui permet d'utiliser toutes les options d'Export/Import.

L'Export/Import en mode incrémental

Vous pouvez créer des exportations incrémentales. Pour cela, il faut procéder comme suit :

1. Effectuer une exportation totale.
2. Exporter les incréments.

Cela permet de synchroniser le contenu des bases de données distantes, sans avoir à importer la totalité de la base à chaque fois.

Les options d'un Export

Pour obtenir les options du programme Export, appelez la commande suivante :

```
exp HELP=Yes
```

Il est regrettable que chaque outil Oracle ait sa propre syntaxe pour appeler l'aide.

```
Export: Release 9.2.0.1.0 - Production on Ve Aou 23 08:31:41 2002  
Copyright (c) 1982, 2002, Oracle Corporation. All rights reserved.
```

Vous pouvez laisser l'export vous demander les param. en tapant la commande EXP suivie de votre nom utilisateur/mot de passe :

Exemple : EXP SCOTT/TIGER

Pour contrôler l'exécution de l'export, entrer la commande EXP suivie de divers arguments.

➔ Pour indiquer les paramètres, utiliser des mots-clés :

Format : EXP KEYWORD=valeur

ou KEYWORD=(valeur1,valeur2,...,valeurN)

Exemple : EXP SCOTT/TIGER GRANTS=Y TABLES=(EMP,DEPT,MGR)

ou TABLES=(T1:P1,T1:P2), si T1 est une table partitionnée

USERID doit être le premier paramètre de la ligne de commande.

Mot-clé	Description (Valeur par défaut)
USERID	nom utilisateur/mot de passe
FULL	export du fichier entier (N)
BUFFER	taille du tampon de données
OWNER	listes des noms utilisateur propriétaire
FILE	fichiers de sortie (EXPDAT.DMP)
TABLES	liste des noms de table
COMPRESS	import dans un extent (Y)
RECORDLENGTH	longueur de l'enregistrement d'E/S
GRANTS	export des autorisations d'accès (Y)
INCTYPE	type d'export incrémentiel
INDEXES	export des index (Y)
RECORD	suivi de l'export incr. (Y)
DIRECT	chemin direct (N)
TRIGGERS	export des déclencheurs (Y)
LOG	fichier journal de sortie d'écran
STATISTICS	analyse des objets (ESTIMATE)
ROWS	export des lignes de données (Y)
PARFILE	nom du fichier de paramètres
CONSISTENT	cohérence entre les tables(N)
CONSTRAINTS	export des contraintes (Y)
OBJECT_CONSISTENT	transaction définie en lecture seule pendant l'export de l'objet (N)
FEEDBACK	afficher la progression toutes les x lignes (0)
FILESIZE	taille maximale de chaque fichier de vidage
FLASHBACK_SCN	SCN utilisé pour rétablir le cliché de session
FLASHBACK_TIME	temps d'obtention du SCN le plus près du temps indiqué
QUERY	clause SELECT utilisée pour exporter un sous-ensemble d'une table
RESUMABLE	suspension lorsqu'une erreur associée à l'espace se produit(N)
RESUMABLE_NAME	chaîne de texte utilisée pour identifier l'instruction RESUMABLE
RESUMABLE_TIMEOUT	temps d'attente pour RESUMABLE
TTS_FULL_CHECK	exécuter un contrôle de dépendance complet ou partiel pour TTS
TABLESPACES	liste des tablespaces à exporter
TRANSPORT_TABLESPACE	- export des métadonnées des tablespaces transportables (N)
TEMPLATE	nom de modèle appelant l'export du mode iAS

Procédure d'export terminée avec succès.

Les options les plus importantes sont commentées dans ce paragraphe.

Paramètre	Usage	Valeur par défaut
BUFFER	Taille de tampon de données.	Dépend de l'OS
COMPRESS	Exporte la définition d'un objet (table, index...) en une seule extension contiguë.	Yes
CONSISTENT	Cohérence entre les tables exportées.	Sans
CONSTRAINTS	Exporte la définition des contraintes d'intégrité.	Yes
DIRECT	Chemin d'accès direct.	No
FEEDBACK	Affiche l'avancement de l'exportation toutes les n lignes.	0
FILE	Nom du fichier de sortie. Si le fichier est trop volumineux, vous pouvez en créer plusieurs d'une taille inférieure grâce au paramètre FILESIZE.	EXPDAT.DMP
FILESIZE	Pour limiter la taille du fichier exporté, vous pouvez en créer plusieurs de taille réduite. FILESIZE indique la taille maximale au-delà de laquelle un nouveau fichier d'export sera créé.	Sans
FULL	Exportation complète.	Yes
GRANTS	Exporte les autorisations d'accès.	Yes
INCTYPE	Type d'exportation incrémentale.	Sans
INDEXES	Exporte la définition des index.	Yes
LOG	Fichier de journalisation des sorties écran. Utile pour conserver une trace de l'exécution d'un export.	Sans
PARFILE	Fichier de commande contenant les paramètres d'exportation.	Sans
OWNER	Liste des utilisateurs à exporter.	Sans
QUERY	Il est possible d'exporter une partie des enregistrements d'une table en filtrant les lignes désirées par la commande QUERY.	Sans
RECORD	Suivi d'exportation incrémentale.	Sans
RECORDLENGTH	Longueur d'enregistrement.	Dépend de l'OS
RECOVERY_TABLESPACES	Liste des tablespaces à récupérer.	Sans
ROWS	Exporte les lignes de données.	Yes
STATISTICS	Analyse (ANALYZE) des objets exportés.	Sans
TABLES	Liste des tables à exporter.	Sans
TABLESPACE	Tous les objets hébergés dans ce tablespace seront exportés.	Sans
USERID	Nom utilisateur / mot de passe.	Sans

Les possibilités de l'Export en mode USER

Un utilisateur peut exporter plus ou moins d'éléments selon les droits qu'il possède :

- un utilisateur exporte ses propres objets :

```
exp userid=scott/tiger file=scott_export.dmp
```

- un administrateur DBA peut exporter plusieurs utilisateurs :

```
exp userid=system/manager owner=(scott,toto) file=export.dmp
```

Dans les deux cas, les exportations sont considérées de type utilisateur (USER), mais les utilisateurs possèdent des droits différents qui permettent d'exporter les objets de plusieurs d'entre eux ou d'un seul.

Les possibilités de l'Export en mode TABLE

D'une façon similaire à l'Export en mode USER, tout dépend des droits de l'utilisateur qui réalise l'exportation. Certains n'accéderont qu'à leurs propres objets :

```
exp userid=scott/tiger tables=(scott.emp, scott.dept, scott.customer)
```

D'autres, de droits plus étendus, exportent des tables propriétés de plusieurs utilisateurs :

```
exp userid=system/manager tables=(scott.emp, toto.travail)
```

Dans les deux cas, les exportations sont considérées comme des exportations en mode TABLE.

Qui peut réaliser une exportation complète ?

Une exportation complète (FULL) est réalisable par tout utilisateur possédant le rôle EXP_FULL_DATABASE. Par défaut, c'est le cas de tous les utilisateurs DBA. On peut aussi attribuer ce rôle à d'autres utilisateurs pour réaliser des exportations complètes. Toutefois, par prudence, nous vous conseillons de réserver les Exports FULL aux seuls administrateurs DBA de la base de données.

```
exp userid=system/manager full=Yes
```

Ne vous laissez pas pirater le contenu de votre base en autorisant l'Export à un trop grand nombre de personnes.

Dans quel ordre sont exportés les objets ?

Voici l'ordre dans lequel les objets sont exportés. Suivant les versions Oracle, il peut évoluer :

- tablespaces ;
- profiles ;
- utilisateurs ;

- rôles ;
- System Privilege Grants ;
- autorisations sur les rôles ;
- rôles par défaut ;
- quotas sur les tablespaces ;
- rollback segments;
- database links;
- séquences (avec leurs droits) ;
- snapshots, logs, queues, refresh ;
- cluster ;
- tables (avec droits, autorisations, commentaires, index, contraintes, audit) ;
- intégrité référentielle ;
- synonymes ;
- vues ;
- procédures stockées ;
- triggers.

L'ordre d'Export est très important. Souvent, les objets dépendent les uns des autres. Par exemple, une table est contenue dans un tablespace et elle possède des contraintes liées à d'autres tables situées dans d'autres tablespaces. Si vous importez une table avant son tablespace, il en résultera une erreur. C'est pourquoi l'Import obéit lui aussi à cet ordre très précis.

Qu'est-ce que l'ARRAY FETCH ?

Pour extraire les données des tables, l'Export utilise des ordres SQL SELECT. Les données doivent être transférées de la base au programme Export, qui n'est pas forcément situé sur la même machine. Si l'Export extrait les données ligne à ligne, cela génère une forte consommation réseau. Pour éviter cela, le mécanisme d'ARRAY FETCH extrait les données par paquet. Pour l'Export et l'Import, la taille des paquets dépend des paramètres BUFFER et RECORDLENGTH.

Utilisation des paramètres BUFFER et RECORDLENGTH

Lors d'un Export, les données sont extraites de la base Oracle 10g, transmises à Export qui les conserve en mémoire avant de les enregistrer dans le fichier d'Export. La variable BUFFER détermine la taille de cette zone mémoire tampon. On retrouve le même paramètre BUFFER lors d'une importation.

La variable RECORDLENGTH détermine la taille des paquets de données extraits de la zone mémoire fixée par BUFFER, pour les écrire dans le fichier disque.

Dans un premier temps, nous vous conseillons de laisser les valeurs par défaut.

Comment sont exportés les champs de type LONG ?

Les champs de type LONG peuvent atteindre 2 Go par enregistrement. L'Export/Import traite les champs de type LONG au même titre que les autres champs.

Comment utiliser un fichier de commande ?

Pour éviter une ligne de commande trop complexe ou écrire des scripts réutilisables, nous vous conseillons d'utiliser un fichier de commande. Par exemple :

```
exp PARFILE=fichier_de_commande.txt
```

Le fichier de commande contient les paramètres sous la forme PARAMETRE=valeur. Par exemple :

```
USERID=scott/tiger
FULL=yes
FILE=mon_fichier_export.dmp
GRANTS=yes
INDEXES=yes
```

En environnement Windows, si le nom du fichier cible comporte un ou plusieurs espaces, vous devez l'encadrer avec trois parenthèses : FILE= ""C:\temp\mon fichier avec des espaces.dmp""

Comment utiliser l'option QUERY ?

Pour réaliser un environnement de test, il est souvent nécessaire d'extraire juste une partie d'une table. La méthode « classique » consiste à réaliser une table temporaire, insérer les données provenant de la table complète à l'aide d'un ordre INSERT...AS SELECT puis d'exporter / importer cette table temporaire.

Oracle Export propose une option très intéressante qui évite la création de cette table temporaire. Vous pouvez n'extraire qu'une partie d'une table. Par exemple :

```
exp scott/tiger TABLE=emp QUERY="WHERE job='CLERK' and sal>1000"
```

Dans ce cas, les données extraites depuis la table EMP correspondront au résultat de l'ordre SQL :

```
SELECT * from emp WHERE job='CLERK' and sal>1000
```

En environnement Windows, pour que le caractère « " » soit correctement interprété, il est nécessaire de le faire précéder par un « \ ». Par exemple :

```
exp scott/tiger TABLE=emp QUERY="\WHERE job='CLERK' and sal>1000\"
```

Comment limiter la taille d'un fichier export ?

Prenons un exemple : vous devez envoyer rapidement un fichier export à un utilisateur distant. Le fichier est très volumineux, ce qui exclu un transfert par le réseau. Vous disposez d'un graveur de CD-Rom et d'un lecteur de cartouches. Dans le cas d'un CD-Rom ou d'une cartouche de sauvegarde, leur taille respective donne la taille maximale du fichier d'export qu'il serait simple de manipuler.

```
exp user/mot_de_passe FILE=exp1.dmp,exp2.dmp,exp3.dmp FILESIZE=600MB
```

Trois fichiers d'export seront créés d'une taille maximale de 600 Mo. Si la taille du fichier export est supérieure, un nouveau nom de fichier vous sera demandé. Vous pouvez graver les fichiers sur trois CD-Rom et les envoyer. Cette technique est la plus simple.

Une autre technique consiste à compresser le fichier export, le copier puis le décompresser à l'arrivée. Cela impose que l'expéditeur et le destinataire soient équipés d'outils identiques.

Comment récupérer les erreurs rencontrées lors d'un export ?

Vous pouvez rediriger les sorties écran pour conserver dans des fichiers la trace de l'exécution d'un export :

```
exp user/mot_de_passe PARFILE=fichier.txt >fichier1.log 2>fichier2.err
```

Dans cet exemple, les messages d'erreur sont séparés des messages d'exécution de l'export. Le fichier fichier1.log contient le résultat des messages écran et le fichier fichier2.err contient les erreurs.

Les options d'un Import

Pour obtenir les options du programme Import, appelez la commande suivante :

```
imp HELP=Yes
```

```
Import: Release 9.2.0.1.0 - Production on Ve Aou 23 08:32:24 2002  
Copyright (c) 1982, 2002, Oracle Corporation. All rights reserved.
```

Pour que l'import vous demande les paramètres., entrez la commande IMP suivie de votre nom utilisateur/mot de passe :

Exemple : IMP SCOTT/TIGER

Pour contrôler l'exécution de l'import, entrez la commande IMP suivie de divers arguments. Pour indiquer paramètre, utiliser des mots-clés :

```
Format : IMP KEYWORD=valeur ou KEYWORD=(valeur1,valeur2,...,valeurN)  
Exemple : IMP SCOTT/TIGER IGNORE=Y TABLES=(EMP,DEPT) FULL=N  
ou TABLES=(T1:P1,T1:P2), si T1 est une table partitionnée
```

USERID doit être le premier paramètre de ligne de commande.

Mot-clé Description (Valeur par défaut)

```

-----
USERID  nom utilisateur/mot de passe
FULL    import du fichier entier (N)
BUFFER  taille du tampon de données
FROMUSER liste noms utilisateur propriétaire
TOUSER  liste des noms utilisateur
FILE    fichier d'entrée (EXPDAT.DMP)
SHOW    afficher seulement contenu du fichier (N)
TABLES  liste des noms de table
IGNORE  ignorer erreurs de création (N)
RECORDLENGTH taille d'enreg. ES
GRANTS  import d'autorisations d'accès (Y)
INCTYPE type d'import incrémentiel
INDEXES import d'index (Y)
COMMIT  valider l'insertion de tableau (N)
ROWS    import de lignes de données (Y)
PARFILE nom fichier de paramètres
LOG      fichier journal de la sortie écran
CONSTRAINTS import de contraintes (Y)
DESTROY  écraser le fichier de données de tablespace (N)
INDEXFILE écriture d'infos de table/index dans le fichier indiqué
SKIP_UNUSABLE_INDEXES sauter la maintenance des index inutilisables (N)
FEEDBACK afficher la progression toutes les x lignes (0)
TOID_NOVALIDATE sauter la validation des ID de type indiqués
FILESIZE  taille maximale de chaque fichier de vidage
STATISTICS importer les statistiques pré-calculées (toujours)
RESUMABLE suspension lorsqu'une erreur associée à l'espace se produit(N)
RESUMABLE_NAME chaîne de texte utilisée pour identifier l'instruction RESUMABLE
RESUMABLE_TIMEOUT temps d'attente pour RESUMABLE
COMPILE   compiler procédures, packages et fonctions (Y)
STREAMS_CONFIGURATION import des métadonnées générales des flux de données (Y)
STREAMS_INSTANTITATION import des métadonnées d'instanciation des flux de données (N)

```

Les mots-clés suivants ne s'appliquent qu'aux tablespaces transportables

TRANSPORT_TABLESPACE - import des métadonnées des tablespaces transportables (N)

TABLESPACES - tablespaces à transporter dans la base de données

DATAFILES - fichiers de données à transporter dans la base de données

TTS_OWNERS - utilisateurs auxquels appartiennent les données contenues dans l'ensemble
 ➡ de tablespaces transportables

Procédure d'import terminée avec succès.

Paramètre	Usage	Valeur par défaut
BUFFER	Taille de tampon de données.	Dépend de l'OS
CHARSET	Jeu de caractères du fichier sous lequel a été réalisée l'exportation.	Sans
COMMIT	Valide l'insertion des données.	Yes
COMPILE	Compile tous les objets lors de l'import.	Yes
CONSTRAINTS	Importe les contraintes liées aux tables.	Yes
DESTROY	Remplace le tablespace s'il existe déjà (attention !).	No
FEEDBACK	Affiche l'avancement de l'importation toutes les n lignes.	0
FILE	Nom du fichier d'entrée. Vous pouvez indiquer plusieurs fichiers résultat d'un seul export.	EXPDAT.DMP
FROMUSER/ TOUSER	Permet de changer le propriétaire des données lors de l'importation.	Sans
FULL	Importation complète du fichier d'Export.	Yes
GRANTS	Importation des autorisations d'accès.	Yes
HELP	Aide en ligne.	Sans
IGNORE	Ignore les erreurs lors de la création.	No
INCTYPE	Type d'Export incrémental.	Sans
INDEXES	Importation et création (si inexistant) des index.	Yes
INDEXFILE	Lit le fichier d'entrée et en extrait la définition DDL de création des tables et des index qu'il contient.	No
LOG	Fichier de journalisation des sorties écran. Utile pour conserver une trace de l'exécution d'un import.	Sans
PARFILE	Fichier de commandes contenant les paramètres d'un Import.	Sans
RECORDLENGTH	Longueur d'enregistrement.	Dépend de l'OS
SHOW	Affiche le contenu du fichier sans importer les objets.	No
STATISTICS	Analyse (ANALYZE) les objets exportés.	Sans
TABLES	Liste des tables à importer.	Sans
USERID	Nom utilisateur/mot de passe.	Sans

Les options disponibles lors d'une importation sont plus nombreuses que celles proposées par l'exportation. Les plus importantes sont commentées dans ce paragraphe.

Quelle action réalise le paramètre **DESTROY** ?

Si vous importez un tablespace, il est créé par la commande CREATE TABLESPACE qui permet de réutiliser les fichiers spécifiés dans la commande s'ils existent déjà sur le disque. DESTROY = Yes peut effacer un fichier déjà existant. Nous vous déconseillons fortement d'utiliser cette option. La valeur par défaut est DESTROY = No.

Comment créer le clone d'une base ?

Si vous voulez effectuer une copie d'une base existante, il faut créer une base cible « minimale » puis procéder comme suit :

- exportez complètement la base source :

```
exp system/manager full=Y file=expfull.dmp
```

- importez ce fichier dans la base cible :

```
imp system/manager full=Y file=expfull.dmp
```

Si vous copiez la base sur la même machine ou si le système d'exploitation change, il faut au préalable créer les tablespaces manuellement sur la base cible, car l'Import cherchera à en créer les fichiers sous le même nom et aux mêmes endroits que les fichiers des tablespaces source.

Peut-on remplacer des données existantes par celles issues d'une importation ?

La réponse est non, avant de réaliser l'importation, il faut supprimer manuellement les données que vous souhaitez mettre à jour. Par défaut, l'Import ajoute les données importées à celles qui existent déjà. C'est le principe de l'importation incrémentale.

Quelles sont les conséquences d'une importation pour les objets propriétés de SYS ?

L'utilisateur SYS est le propriétaire du dictionnaire de données qui gère la vie de chaque base Oracle 10g. Sauf à lancer des scripts SQL fournis par Oracle, on ne doit jamais travailler ni créer d'objets sous le compte utilisateur SYS. Toute base cible possède, elle aussi, son dictionnaire interne de données, qui ne peut pas être effacé par celui issu d'une autre base. C'est pourquoi les objets de l'utilisateur SYS ne sont ni exportés, ni importés. De la même façon, les autorisations (GRANT) attribuées aux objets de SYS ne sont pas exportées.

C'est une autre raison pour laquelle l'Export/Import est l'un des éléments d'une sauvegarde logique, différente de la sauvegarde physique réalisée par la copie des fichiers composant la base.

Les mots de passe des utilisateurs sont-ils automatiquement exportés/importés ?

Les mots de passe des utilisateurs autres que SYS et SYSTEM ne sont pas affectés par un Export/Import. En revanche, les mots de passe SYS et SYSTEM sont mis à jour par un Import en mode FULL. Pour éviter cela, il faut se connecter pour modifier les mots de passe de SYS et SYSTEM.

```
# Connexion sous le compte OS propriétaire d'Oracle  
su - oracle  
  
sqlplus /nolog  
connect / as sysdba
```

Par sécurité, les mots de passe sont cryptés lorsqu'ils figurent dans un fichier d'Export.

Exporter tous les utilisateurs, est-ce identique à un Export en mode FULL ?

La réponse est non. Lorsque vous exportez le contenu d'une base, les objets sont exportés dans un ordre particulier. Lorsque l'on exporte uniquement les objets des utilisateurs, certains éléments exportés lors d'un Export FULL manquent.

Comment sont importées les contraintes d'intégrité référentielle ?

Les contraintes d'intégrité référentielles sont créées lorsque toutes les tables et leurs données ont été importées.

Si les contraintes d'intégrité existent déjà dans la base cible, l'ordre d'Import des tables, géré lors de l'Export, permet d'éviter tout problème.

Peut-on changer le nom d'une table ?

La réponse est non. Il n'est pas possible d'exporter une table sous un nom et de l'importer sous un autre. Pour cela, il faut renommer la table avant l'Export ou après l'Import.

Quelle répercussion à l'Import sur les index, contraintes et triggers d'une table ?

Aucun, en ce sens que l'Import crée les nouveaux objets s'ils n'existent pas, mais ne supprime pas ceux qui existent déjà.

Des problèmes peuvent survenir si un objet importé entre en conflit avec un autre de même nom, existant déjà dans la base de données cible. Par exemple, deux tables de structure différente portant le même nom.

Comment sont gérés les droits et les synonymes ?

Les droits (GRANT) et les synonymes sont importés suivant l'ordre dans lequel ils ont été créés.

À quoi sert l'option *COMPRESS=Yes* ?

Cette option est très utile pour réorganiser vos bases de données en évitant toute fragmentation des tables et des index. Une table ou un index possède une taille initiale. Lorsque cet espace alloué est plein, un extent (une extension) est automatiquement créé. La taille du segment initial, du segment suivant et de l'incrément donné aux autres segments sont des paramètres liés à chaque table et index. Ils sont précisés lors de la création de la table et peuvent être modifiés ultérieurement. Par exemple :

```
CREATE TABLE scott.emp (
    empno NUMBER(4),
    ...
    deptno NUMBER(2) )
STORAGE (
    INITIAL 1 M-- 1 Mo pour l'extent initial
    NEXT 100k-- taille du premier extent
    PCTINCREASE 10 -- % d'incrément des extents suivants
)
```

Si des extents sont alloués lorsque la taille de l'objet augmente, ils perdurent lorsque sa taille diminue.

Il est plus facile et plus rapide pour la base Oracle 10g de lire des données contenues dans un même segment, car des segments dispersés au sein d'un tablespace obligent le bras du disque dur à de fréquents déplacements. Le gain est particulièrement sensible pour les index.

Une optimisation de premier niveau de la base de données consiste à regrouper une table ou un index en un seul extent.

L'exportation d'une table ou d'un index contient son ordre SQL de création avec les paramètres de STORAGE initiaux. L'option `EXP COMPRESS=Y` calcule la somme des différents extents et l'affecte au segment INITIAL.

L'importation ne modifie pas les paramètres STORAGE si l'objet existe déjà dans la base cible. Pour supprimer les extents d'une table ou d'un index il faut supprimer l'objet avant de l'importer pour, qu'à sa création, les nouveaux paramètres STORAGE soient pris en compte.

Comme cette opération s'effectue souvent sur des bases de production, nous vous conseillons de vous entraîner et de disposer d'une sauvegarde avant de supprimer vos tables et leurs données...

Notez que si la table importée nécessite un extent INITIAL de grande dimension, la base de données cible ne dispose peut-être pas d'autant de place en un seul segment contigu. L'annexe 3, *Procédures pour le DBA*, possède des ordres SQL qui vérifient ces aspects.

Comment diminuer la taille d'une table ou d'un index ?

Il est possible qu'une table soit de taille importante mais relativement vide car des enregistrements ont été supprimés. Il est alors préférable de la pré-créez manuellement. Pour cela procédez comme suit :

- exportez la table sans ses données :

```
exp userid=scott/tiger table=(emp)
```

- récupérez dans un fichier texte l'ordre SQL de création de la table en effectuant un Import avec le paramètre :

```
imp userid=scott/tiger table=(emp) INDEXFILE=fichier.txt
```

- modifiez manuellement la clause STORAGE de l'ordre de création ;
- créez la table sous SQL*Plus en exécutant l'ordre SQL modifié.

Il est aussi possible d'utiliser les procédures standard DBMS_DESCRIBE ou DBMS_METADATA.

Comment récupérer les ordres SQL de création de tous les objets d'une base ?

La méthode précédente, utilisant le paramètre INDEXFILE, permet de récupérer dans un fichier texte l'ensemble des ordres SQL de création des objets de la base de données (tables, vues, index, triggers...).

Ce paramètre doit être utilisé uniquement lors de l'importation. Son utilisation inhibe toute action sur la base de données cible.

Si une table et son index ont des propriétaires différents, que se passe-t-il ?

L'utilisateur SCOTT possède par exemple la table EMP et TOTO a créé un index sur cette table. L'Export suivant ne porte pas sur l'index qui a été créé par l'utilisateur TOTO :

```
exp userid=scott/tiger tables=(emp) indexes=Y
```

Les règles de bonne programmation imposent que tous les objets d'une application soient créés sous le même compte utilisateur. Des droits ou des rôles (lecture, lecture-écriture...) sont ensuite attribués aux autres utilisateurs.

Qu'en est-il des tablespaces OFFLINE ?

Un tablespace peut être actif (ONLINE) ou inactif (OFFLINE). Lors d'une exportation, les données et objets contenus dans un tablespace OFFLINE ne sont pas exportés. C'est

pourquoi un Export FULL ne peut pas être considéré comme une sauvegarde fiable et complète à 100 %.

Les informations du tablespace SYSTEM sont-elles automatiquement exportées ?

Non, les informations concernant les fichiers du tablespace SYSTEM (emplacement, nombre, taille...) ne sont pas exportées. En effet, pour importer une base, la base cible fonctionne, donc son tablespace SYSTEM et ses fichiers existent déjà.

Comment connaître le résultat d'un Import lancé sous at ?

at est l'outil d'ordonnancement standard de Windows qui permet de planifier des tâches. Il est l'outil idéal pour exécuter automatiquement des exportations et leurs sauvegardes durant la nuit. Pour disposer d'un compte-rendu d'exécution d'exportation, utilisez l'option LOG.

Quelles sont les erreurs rencontrées ?

Un Export/Import peut se terminer de trois façons :

- il peut aboutir avec succès ;
- il peut rencontrer un avertissement (warning) ;
- il peut comporter des erreurs.

Un avertissement peut être mineur : c'est le cas d'une erreur dans le nom de la table à exporter. Mais certaines erreurs peuvent provoquer l'arrêt brutal de Export/Import.

Vous pouvez récupérer dans des fichiers les erreurs rencontrées lors d'un import. Ce point est traité dans la partie « export » de ce chapitre.

Oracle Data Pump

Oracle Data Pump Export et Oracle Data Pump Import sont la nouvelle génération des utilitaires Export/Import apparus avec Oracle 10g. Ils offrent de meilleures performances que l'import/export et permettent l'échange à la fois des données et de la structure d'une base.

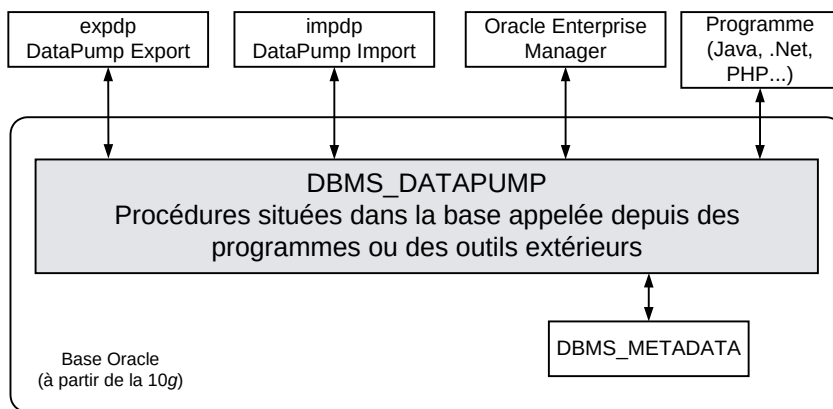
Les nouveautés apportées par Data Pump

Outre les performances, la nouveauté majeure apportée par Data Pump concerne la possibilité d'arrêter ou de relancer un Data Pump Export ou Import. Comme Data Pump cible les gros volumes de données, cette nouveauté est très intéressante.

Les différences entre Data Pump et Export/Import

Tout d'abord, seules des données extraites par Data Pump Export peuvent être importées par Data Pump Import. Il y a donc une rupture avec Export/Import qui permet d'effectuer des transferts de données d'une version d'Oracle vers une autre. Data Pump sera donc à utiliser à partir d'Oracle 10g et pour les futures versions.

Figure 18-2
Principaux composants
de Data Pump



Autre point, Export et Import peuvent être utilisés en mode client-serveur. Ce mode de fonctionnement fait transiter les données par un réseau et ralentit la durée du traitement : il convient de l'éviter lorsque l'on manipule de gros volumes de données. C'est pourquoi Data Pump ne travaille qu'en « local », sur le serveur où est située la base. Data Pump Export et Import manipulent exclusivement des données situées sur un serveur.

Pouvoir relancer des Data Pump Export ou Import est très intéressant pour les DBA qui manipulent de gros volumes de données ou lorsque les opérations sont longues. Les cas rencontrés peuvent être les manques d'espace disque (pour le Data Pump Export) ou un tablespace sous-dimensionné (pour le Data Pump Import). Lorsque le traitement Data Pump rencontre une erreur, le DBA peut intervenir, corriger l'erreur puis relancer le traitement. Les deux exemples suivants montrent comment relancer un Data Pump Export et Import.

Comment relancer Data Pump Export

Ce premier exemple montre comment relancer un Data Pump Export. Nous allons effectuer un Data Pump Export en précisant une taille de fichier bien inférieure à celle des données à exporter. Une fois l'erreur rencontrée, nous ajouterons un fichier au Data Pump Export puis nous continuerons l'opération.

Tout d'abord, préparons les répertoires où seront exportées les données :

```
create directory mon_repertoire_pump as 'c:\export\datapump';  
grant read, write on directory mon_repertoire to system;
```

Ensuite, utilisons expdp (*Export Data Pump*) pour lancer l'export, en précisant un nom de job facile à identifier : mon_job.

```
expdp system/manager schemas=gros_schema \
directory= mon_repertoire_pump \
logfile=mon_job.log filesize=100000 dumpfile=mon_job1.dmp \
job_name=mon_job
```

Le signe \ indique que la commande continue sur une autre ligne et qu'elle ne doit pas être immédiatement interprétée.

Le Data Pump Export va démarrer puis indiquer une erreur, le fichier cible étant trop petit. Une fois l'erreur rencontrée, connectons-nous sous SQL*Plus, sous l'utilisateur system.

```
SQL> select job_name, state from dba_datapump_jobs ;
```

```
JOB_NAME    STATE
-----
MON_JOBNOT  RUNNING
```

Pour corriger le problème, attachons-nous au job MON_JOB en cours et ajoutons un second fichier au fichier initial d'export.

```
expdp system/manager attach=MON_JOB

Export: Release 10.1.0.2.0 - Production le Samedi 4 Septembre 2004
...
...
Job : MON_JOB
  Owner : SYSTEM
  Operation : EXPORT
...
...
```

Nous sommes maintenant dans l'outil Export : ajoutons le fichier.

```
Export> add_file=mon_job2.dmp
```

La commande status nous permet de voir la modification apportée.

```
Export> status
Job=MON_JOB
Owner : SYSTEM
  Operation : EXPORT
  Mode=SCHEMA
  STATE= IDLING
  Bytes Processed: 60,453
  Percent done: 80
  Current Parallelism: 1
  Job Error Count: 0
  Dump file Size: F:\travail\oracle\mon_job1.dmp
    Size: 100,000
    Bytes written: 54,000
  Dump file Size: F:\travail\oracle\mon_job2.dmp
```

Il faut maintenant relancer le job par la commande `CONTINUE_CLIENT`.

```
Export> continue_client - on peut aussi utiliser START_JOB
```

Comment relancer Data Pump Import

Dans ce second exemple, importons les données provenant d'un Data Pump Export dans un tablespace trop petit pour les contenir. Tout d'abord, créons le petit tablespace.

```
SQL> create tablespace petit_tablespace
      datafile 'D:\database\TEST\petit_tablespace1.dbf'
      size 500k extent management local ;
```

Lançons le Data Pump Import.

```
impdp system/manager dumpfile=gros_fichier.dmp \
      remap_tablespace=system:petit_tablespace
      logfile=mon_job_import.log job_name=mon_job_import

Import: Release 10.1.0.2.0 - Production le Samedi 4 Septembre 2004
...
...
Processing object type SCHEMA_EXPORT/TABLE/TABLE
...
...
ORA-39171: Job is experiencing a resumable wait.
ORA-01658: unable to create INITIAL extent for segment in tablespace PETIT_TABLESPACE.
```

Le job est en attente. Par défaut, la durée d'attente est de deux heures. Passé ce délai, le job est tué. Cela laisse le temps au DBA d'intervenir et évite de laisser des jobs non terminés.

Le job étant bloqué, interrompons-le par un « control C ».

```
^C
Import> stop_job=immediate
```

Ajoutons un fichier au tablespace.

```
SQL> alter tablespace petit_tablespace
      Add datafile 'D:\database\TEST\petit_tablespace2.dbf'
      size 10M autoextent on maxsize 100M ;
```

Attachons-nous au job et relançons-le.

```
impdp system/manager attach=mon_job_import

Import: Release 10.1.0.2.0 - Production le Samedi 4 Septembre 2004
...
...
Job Error Count: 0
...
Worker 1 Status:
  State: UNDEFINED
  Object Schema: GROS_SCHEMA
  Object name: COMMANDES
```

```
Object type: SCHEMA_EXPORT/TABLE/TABLE  
Completed objects: 32  
Worker parallelism: 1
```

Il faut alors relancer le job.

```
Import> start_job
```

Tout comme pour l'export, nous pouvons vérifier l'avancement de l'import par la commande `status`. Lorsque le job est terminé, consultez les fichiers log.

Utiliser Data Pump pour récupérer le code d'une procédure

L'Export traditionnel permet de récupérer le code de procédures, mais le script généré contient aussi le code des tables, vues, index... que vous ne désirez peut-être pas. Par exemple, pour recréer une ou plusieurs procédures sur une autre base ou pour un utilisateur particulier, la commande suivante vous permet de récupérer uniquement le code des procédures et rien d'autre : ni tables, ni index, etc.

```
expdp system/manager directory=mon_repertoire_pump \  
dumpfile=exp_procedure.bmp include=PROCEDURE
```

Le fichier `exp_procedure.bmp` est créé au format Data Pump. Pour extraire le code SQL des procédures :

```
Impdp system/manager directory=mon_repertoire_pump \  
dumpfile=exp_procedure.dmp sqlfile=code_procedure.sql
```

Quelles sont les autres possibilités de Data Pump ?

Les possibilités offertes par Data Pump Export et Import sont en grande partie similaires à celles de l'Export et Import traditionnels. Pour découvrir les autres possibilités, je vous conseille d'utiliser en ligne `expdp help=Y` ou `impdp help=Y`.

Vous pouvez aussi accéder à toutes les possibilités de Data Pump depuis la base de données par la procédure `DBMS_DATAPUMP`. Vous pouvez ainsi imaginer construire des procédures en PL/SQL, appelables depuis un site Web (Java, .Net, PHP...) ou tout autre programme qui utilisera Data Pump.

Oracle SQL*Loader

Nous avons choisi de vous présenter SQL*Loader par le biais d'une série de questions/réponses et d'exemples.

Quelle utilisation pour SQL*Loader ?

SQL*Loader est un utilitaire servant à charger des données provenant de fichiers externes, dans une base Oracle 10g. Il permet d'intégrer de nombreux formats de données, à partir d'un ou de plusieurs fichiers, dans une ou plusieurs tables.

C'est l'outil fréquemment utilisé lors de migrations d'une base vers une autre ou lors de la construction d'un datawarehouse ou autre infocentre.

Comment fonctionne SQL*Loader ?

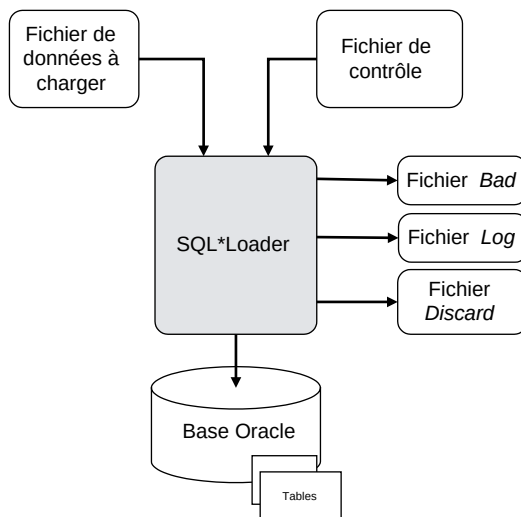
Pour piloter le chargement des données, SQL*Loader utilise un *fichier de contrôle*. Ce dernier détermine les fichiers de données à charger et leur table de destination.

Durant le chargement, SQL*Loader peut produire trois types de fichiers :

- le fichier log, qui effectue un compte rendu complet du chargement ;
- le fichier bad, qui conserve les enregistrements qui n'ont pu être insérés (rejetés lors de l'insertion dans la table cible) ;
- le fichier discard, qui stocke les enregistrements illisibles (provenant du fichier de données à charger, mais non reconnus par SQL*Loader).

Figure 18-3

Principe de fonctionnement de SQL*Loader



Où se situe le jeu d'essai de SQL*Loader ?

Oracle 10g est livré avec un jeu d'essai SQL*Loader. Les fichiers d'essai se terminent tous avec l'extension .ctl et se situent généralement dans :

■ ORACLE_HOME\rdbms\demo

Nous vous conseillons de les étudier en complément de ce chapitre.

Comment utiliser SQL*Loader ?

SQL*Loader est avant tout un utilitaire « mode caractère ». Il peut être piloté par un utilitaire graphique, qui ne fera que transmettre les différents paramètres à la commande en ligne.

Sous Windows, il se nomme sqlldr et se situe dans ORACLE_HOME\bin. Pour accéder à l'aide en ligne :

```
C:\> sqlldr
```

La commande usuelle se présente comme suit :

```
C:\> sqlldr utilisateur/mot_de_passe@alias control=controle.ctl
```

avec : *Utilisateur/mot_de_passe@alias* : connexion à la base Oracle 10g locale ou distante.

Le fichier de contrôle controle.ctl contient les données suivantes :

```
load data
1      infile *
2      replace
3      into table COURS
4      ( codeposition (02:05) char(4),
        nomposition (08:27) char(20)
        )
5      begindata
      FRAN  LANGUE FRANCAISE
      ENGL  LANGUE ANGLAISE
      MATH  MATHEMATIQUES
      PHYS  PHYSIQUE
```

La numérotation indiquée ne doit pas figurer, elle sert à commenter l'exemple.

En outre, il faut tenir compte du fait que :

1. les données ne proviennent pas d'un fichier mais sont présentes à la fin du fichier de contrôle ;
2. les données de la table COURS seront remplacées ;
3. la table cible, dans la base Oracle 10g, doit être accessible à l'utilisateur de connexion ;
4. il faut insérer des informations dans les colonnes CODE et NOM de la table COURS. Pour les données fournies en entrée, le CODE sera lu entre le deuxième et le cinquième caractère, le NOM entre le huitième et le vingt-septième ;
5. les données suivantes sont à insérer.

Comment générer les fichiers log, bad, discard ?

Ces fichiers sont créés soit à partir de la commande en ligne de SQL*Loader, soit à partir du fichier de contrôle :

```
C:\> sqlldr user/pwd control=test.ctl log=test.log bad=test.bad discard=test.dsc

load data
```

```
infile 'fichier_donnees.dat'  
badfile 'test.bad'  
discardfile 'test.dsc'  
truncate  
into table test  
  (colonne1,  
   colonne2,  
   colonne3)
```

Pour faciliter leur repérage, il est préférable d'utiliser le même nom pour les fichiers de contrôle, à savoir : *bad*, *log*, *discard*, plutôt que celui de la table que vous chargez.

De même, disposer du plus d'éléments possibles dans les fichiers de contrôle en facilite l'écriture, la mise au point et la maintenance.

Les données à insérer peuvent-elles être de longueur variable ?

La réponse est oui. Dans l'exemple suivant, la première et la seconde colonne sont de longueur fixe :

```
LOAD DATA  
  INFILE *  
  INTO TABLE test  
  1 FIELDS TERMINATED BY "," OPTIONALLY ENCLOSED BY '''  
  2 TRAILING NULLCOLS  
    ( colonne1,  
      colonne2  
    )  
  BEGINDATA  
  11111,AAAAAAAAAA  
  222,"A,B,C,D,"
```

Les données seront séparées par des virgules et peuvent être entre guillemets (").

TRAILING NULLCOLS signifie qu'à la fin de chaque champ inséré, les « blancs » ou « espaces » seront supprimés. On évite de la sorte les colonnes qui contiennent des valeurs de type "XXX " avec le caractère " " qui « remplit » inutilement le champ.

N'oubliez pas qu'avec Oracle, une colonne de type VARCHAR2(5) ne contient que les données stockées et que "XXX" et "XXX " ne sont pas identiques.

Les données à insérer peuvent-elles être de longueur fixe ?

La réponse est oui. L'exemple suivant fait appel à un positionnement absolu des données lues dans le fichier d'entrée :

```
LOAD DATA  
INFILE *
```

```

    INTO TABLE positionnement_absolu
    ( data1 POSITION(1:5),
      data2 POSITION(6:15)
    )
    BEGINDATA
    11111AAAAAAAAA
    22222BBBBBBBBB

```

Attention ! Lors du comptage de la position des champs, il est facile de commettre une erreur en provoquant un décalage. Vérifiez toujours, après un chargement, que les données ont été insérées dans les colonnes souhaitées.

SQL*Loader crée-t-il la table dans la base Oracle 10g ?

La réponse est non. La table doit être créée avant le chargement des données.

Quels liens existe-t-il entre la table et le fichier de contrôle ?

Considérons l'exemple suivant. La structure de la table TEST est :

```

CREATE TABLE TEST (
  COLONNE1      NUMBER(5) PRIMARY KEY,
  COLONNE2      VARCHAR2(10),
  COLONNE3      VARCHAR2(10),
  COLONNE4      VARCHAR2(10)
);

```

Dans la colonne1, nous souhaitons insérer un nombre et dans les colonnes suivantes, les lettres de l'alphabet, dans l'ordre.

Le fichier de contrôle a pour rôle d'indiquer à SQL*Loader où placer les données lues dans le fichier de données :

```

LOAD DATA
  INFILE *
  INTO TABLE test
  FIELDS TERMINATED BY "," OPTIONALLY ENCLOSED BY '"'
  TRAILING NULLCOLS
  (colonne1,
   colonne2,
   colonne3,
   colonne4
  )
  BEGINDATA
  11111,AAAAAAAAA,BBB,CCCCCCC
  22222,AAAAAAAAA,BBBBBBBBBB,CCC
  333,AAAAA,BBBBBB,CCCC

```

Dans cet exemple, l'ordre des colonnes correspond à celui des données à insérer.

```

LOAD DATA
  INFILE *
  INTO TABLE test
  FIELDS TERMINATED BY "," OPTIONALLY ENCLOSED BY ''
  TRAILING NULLCOLS
  (colonne3,
   colonne2,
   colonne4,
   colonne1
  )
  BEGINDATA
  BBBB,AAAAA,CCCC,11111
  BBBB,AAAAAAA,CCCCC,222
  BBBBBB,AAAAAAAAA,CCCCC,3333333

```

Ce sont bien les données à intégrer qui définissent l'ordre des colonnes du fichier de contrôle.

```

LOAD DATA
  INFILE *
  INTO TABLE test
  FIELDS TERMINATED BY "," OPTIONALLY ENCLOSED BY ''
  TRAILING NULLCOLS
  (colonne2,
   colonne4,
   colonne1
  )
  BEGINDATA
  AAAA,CCCC,11111
  AAAAAA,CCCCC,222
  AAAAAAAAA,CCCCC,3333333

```

Dans l'exemple précédent, toutes les colonnes de la table ne sont pas insérées, car les valeurs « BBB... » sont absentes. En revanche, la colonne1 est obligatoire, car c'est une clé primaire, qui doit donc exister pour chaque enregistrement inséré.

Comment préciser le type des données en entrée ?

Le type des données dans la table cible est bien défini. Souvent, la difficulté est de préciser à SQL*Loader le type des données en entrée pour qu'il les lise correctement.

C'est grâce au fichier de contrôle que nous précisons les types :

```

LOAD DATA
  INFILE *
  INTO TABLE test
  (colonne1 INTEGER EXTERNAL,
   colonne2 CHAR,
   colonne3 INTEGER EXTERNAL,
   colonne4 DECIMAL EXTERNAL
  )
  ....

```

C'est en précisant ce type que vous pouvez lire n'importe quel format de données en entrée : signé, binaire packé décimal, etc.

Comment charger des données qui ne sont pas codées en ASCII ?

Si les données proviennent de systèmes informatiques dont le codage interne n'est pas en ascii, vous pouvez demander leur conversion dans le fichier de contrôle. Cela vous permet de récupérer correctement tous les caractères spécifiques à la langue française : àéèùç... C'est une possibilité très utile que propose SQL*Loader.

Par exemple, si les données proviennent d'un grand système IBM, elles sont souvent codées en EBCDIC. Transférez alors les fichiers jusqu'à la machine qui héberge la base Oracle 10g en mode binaire. Vous obtiendrez en local, un fichier qui sera l'image exacte (en binaire) de celui d'origine. Chargez-le en indiquant son type d'origine dans le fichier de contrôle, de sorte que SQL*Loader convertisse le fichier lors du chargement.

```
LOAD DATA
  CHARACTERSET 'WE8EBCDIC500'
  INFILE fichier_en_EBCDIC
  INTO TABLE test
  (
    colonne1
    ...
    colonneN
  )
```

Vous devrez peut-être effectuer quelques essais pour trouver le jeu de caractères convertissant correctement vos données, mais cela vaut la peine. Sinon, il faut modifier les données une fois chargées dans la base, opération longue et complexe.

Peut-on modifier les données lors de leur insertion ?

La réponse est oui. Vous disposez en outre de pratiquement toutes les fonctions SQL portant sur les nombres, chiffres, dates, etc.

```
LOAD DATA
  INFILE *
  INTO TABLE test
  1 ( numero                                "ma_sequence.nextval",
  2   heure_chargement                      "to_char(SYSDATE, 'HH24:MI')",
  3   colonne2      POSITION(1:5)  ":colonne1/100",
      colonne1      POSITION(6:15) "upper(:colonne2)"
  )
  BEGIN DATA
  11111AAAAAAAAA
  22222BBBBBBBBB
```

1. Insérez un numéro de séquence qui numérottera les enregistrements en entrée.
2. L'heure de chargement sera l'heure système.

3. Lors de l'utilisation des fonctions SQL, faites précéder la référence à la colonne de « : », en l'encadrant par des « " » .

Peut-on insérer des données dans plusieurs tables ?

La réponse est oui. C'est très utile lorsqu'on récupère des données provenant de fichiers qui ne sont pas normalisés au sens SQL :

```
LOAD DATA
INFILE *
REPLACE
INTO TABLE table1
    WHEN champ_test != ' '
( champ_test POSITION(1:4)  INTEGER EXTERNAL,
  colonne2  POSITION(6:15)  CHAR,
  colonne3  POSITION(17:18) CHAR,
  colonne4  POSITION(20:23) INTEGER EXTERNAL
)
INTO TABLE table2
    WHEN champ_test2 != ' '
( champ_test2 POSITION(25:27) INTEGER EXTERNAL,
  colonne2    POSITION(1:4)   INTEGER EXTERNAL
)
```

*Comment charger plus vite avec SQL*Loader ?*

Le paramètre `DIRECT=TRUE` dans la ligne de commande de SQL*Loader permet d'améliorer considérablement la vitesse de chargement de vos fichiers, en court-circuitant une bonne partie des opérations effectuées normalement par la base de données.

Certaines commandes présentes dans le fichier de contrôle (par exemple, "upper(:colonne2)") ne permettent pas d'utiliser le mode `DIRECT`, car elles font appel aux fonctions du SQL.

Lors du chargement en mode `DIRECT`, aucune indication ne s'affiche à l'écran. Seule la fin est indiquée, chose surprenante car le mode normal nous habitue à suivre la progression du chargement.

La recommandation générale est de mettre au point vos fichiers de chargement en mode normal puis de les exécuter en mode `DIRECT`.

*Quelle relation entre SQL*Loader et les index ?*

Au fur et à mesure du chargement d'une table, les index qui s'y rapportent se construisent : il faut donc disposer de place pour le tri et la construction de l'index dans le tablespace temporaire.

En mode normal, les données sont progressivement ajoutées dans la table, et donc dans les index. Leur construction ne nécessite pas une importante zone de tri.

En mode `DIRECT=TRUE`, les index sont construits en une fois, à la fin du chargement de la table. Ils ont besoin de plus de place dans la zone de tri.

De manière générale, il est préférable de commencer par charger les données, puis de construire les index. Cela améliore le temps global des deux opérations.

Comment forcer SQL*Loader à effectuer un commit à la fin du chargement ?

Vous ne pouvez pas réaliser une telle opération. Néanmoins, en attribuant au paramètre `ROWS=valeur` une forte valeur, vous augmentez le nombre d'enregistrements entre deux commits.

Comment sont utilisés les rollback segments ?

Lors d'un chargement normal des données, les rollback segments doivent être suffisamment volumineux pour contenir les données entre deux commits.

Le mode `DIRECT` court-circuite les rollback segments.

Que contient le fichier log, résultat d'un chargement ?

Le fichier log, résultat d'un chargement, est essentiel pour valider le bon chargement des données. Le fonctionnement standard de SQL*Loader interrompt le chargement d'un fichier au bout de 50 erreurs rencontrées. Il peut donc s'arrêter avant terme.

Création de la table de test sous SQL*Plus :

```
SQL> connect scott/tiger
Connecté.
SQL> CREATE TABLE TEST (
  2  COLONNE1 NUMBER(5) PRIMARY KEY,
  3  COLONNE2 VARCHAR2(10),
  4  COLONNE3 VARCHAR2(10),
  5  COLONNE4 VARCHAR2(10)
  6  );
```

Table créée.

Fichier de contrôle :

```
LOAD DATA
  INFILE *
  INTO TABLE test
  FIELDS TERMINATED BY "," OPTIONALLY ENCLOSED BY '"'
  TRAILING NULLCOLS
  (colonne1,
   colonne2,
   colonne3,
```

```

        colonne4
    )
    BEGINDATA
    11111,AAAAAAAAAA,BBB,CCCCCCC
    22222,AAAAAAAAAA,BBBBBBBBBB,CCC
    333,AAAAAAAAAA,BBBBBB,CCCCC

```

Exécution du chargement :

```
C:\> sqlldr userid=scott/tiger control=test.ctl log=test.log
```

Voici comment se présente le fichier log, résultat du chargement de la table TEST, piloté par le fichier de contrôle précédent :

```
SQL*Loader: Release 9.0.1 - Production on Di Aug 26 15:43:18 2001
```

```
(c) Copyright 1997 Oracle Corporation. All rights reserved.
```

```

Fichier de contrôle : test.ctl
Fichier de données : test.ctl
Fichier défectueux : test.bad
Fichier de rebut : aucune spécification

```

```
(Allouer tous les rebuts)
```

```

Nombre à charger      : ALL
Nombre à sauter       : 0
Erreurs permises      : 50
Tableau de liens      : 64 lignes, maximum de 65536 octets
Continuation          : aucune spécification
Chemin utilisé        : Conventionnel

```

```

Table TEST, chargée à partir de chaque enregistrement physique.
Option d'insertion en vigueur pour cette table : INSERT
option TRAILING NULLCOLS effective

```

Nom de colonne	Position	Long.	Séparat.	Encadrem.	Type de données
COLONNE1	FIRST	*	,	0(")	CHARACTER
COLONNE2	NEXT	*	,	0(")	CHARACTER
COLONNE3	NEXT	*	,	0(")	CHARACTER
COLONNE4	NEXT	*	,	0(")	CHARACTER

Table TEST:

```

Chargement réussi de 3 Lignes.
0 Lignes chargement impossible dû à des erreurs de données.
0 Lignes chargement imposs. car échec de toutes les clauses WHEN.
0 Lignes chargement imposs. car tous les champs étaient non renseignés.

```

```

Espace affecté au tableau de liens:      18944 octets(64 lignes)
Espace alloué à la mémoire en plus du tableau de liens: 0 octets

```



```
Nombre total d'enregistrements logiques ignorés :      0
Nombre total d'enregistrements logiques lus :          3
Nombre total d'enregistrements logiques rejetés :      0
Nombre total d'enregistrements logiques mis au rebut : 0
```

```
Le début de l'exécution a été effectué Di Jan 25 15:43:18 2005
La fin de l'exécution a été effectuée Di Jan 25 15:43:25 2005
```

```
Temps écoulé (ELAPSED):      00:00:06.61
Temps processeur (CPU):      00:00:00.18
```

En effectuant une lecture attentive, vous constaterez que tout est mentionné dans ce fichier log : le contenu du fichier de contrôle, les paramètres lancés lors de l'exécution de SQL*Loader et le résultat de l'exécution incorporant la durée totale du chargement.

Oracle SQL*Plus

SQL*Plus est un outil en mode caractère permettant l'accès à une base de données Oracle. Il peut être utilisé pour exécuter de façon interactive des ordres SQL unitaires, lancer des fichiers contenant un ensemble d'ordres SQL, exécuter des programmes écrits en PL/SQL, afficher le résultat de requêtes SQL, formater le résultat d'une requête et en améliorer la présentation, accéder à plusieurs bases de données simultanément, etc.

SQL*Plus remplace maintenant l'ancien Server Manager pour créer, démarrer, arrêter, administrer et effectuer des opérations complexes de restauration sur les bases Oracle.

Server Manager n'est plus livré avec les nouvelles versions depuis Oracle9. SQL*Plus devient donc l'interface privilégiée d'Oracle par laquelle la totalité des ordres SQL concernant l'administration et l'utilisation d'Oracle peuvent être exécutés.

SQL*Plus est un outil généraliste, livré depuis des années avec toutes les versions d'Oracle. Il a l'avantage d'exister sur toutes les plates-formes où Oracle est porté et dispense de l'achat d'un outil complémentaire pour accéder à une base Oracle 10g. Il présente l'inconvénient d'une ergonomie en mode caractère qui peut faire préférer pour certains usages des outils graphiques parfois moins performants mais plus agréables d'utilisation.

Ce paragraphe a comme but de vous faire découvrir cet outil puissant et d'en cerner les principales caractéristiques. Nous vous présenterons de nombreux exemples dont l'utilisation de SQL*Plus pour créer des interrogations dynamiques.

Pourquoi administrer Oracle avec SQL*Plus ?

Pourquoi se limiter à l'utilisation d'un outil dont l'interface est en mode caractère ? OEM (*Oracle Enterprise Manager*) est un outil graphique qui reprend une partie des possibilités de SQL*Plus. Bien que l'approche graphique proposée par OEM soit très utile, elle ne nous dispense pas d'en connaître et d'en comprendre les concepts et ordres SQL

induits. De plus, dans certaines situations critiques, seul SQL*Plus peut vous permettre de redémarrer votre base. Par ailleurs, un outil mode caractère et des scripts de commande sont indispensables à l'automatisation de tâches comme le démarrage, l'arrêt et les sauvegardes de vos bases. Pour toutes ces raisons, SQL*Plus est un outil que tout administrateur Oracle se doit de connaître.

Les deux modes de connexion à SQL*Plus

Nous avons vu que SQL*Plus peut être utilisé pour deux types d'ordres SQL :

- des ordres SQL de création, démarrage, arrêt, restauration de la base ;
- des ordres SQL d'utilisation de la base.

Le mode de connexion est différent suivant le type d'utilisation car la création ou le démarrage d'une base nécessitent plus de pouvoirs qu'une simple utilisation.

Connexion à SQL*Plus en mode « administration »

Dans chaque outil Oracle, le lancement est immédiatement suivi d'une demande de connexion à une base de données existante. Mais comment opérer sur une base qui n'est ni créée ni démarrée ? À cet instant, la base de données ne peut intervenir dans le processus d'identification et d'autorisation d'accès. À cet effet, l'utilisateur Windows qui exécute SQL*Plus doit bénéficier de droits supplémentaires provenant de privilèges issus du système d'exploitation ou vérifiés dans un fichier mot de passe situé à l'extérieur à la base. Ces points sont détaillés au chapitre 24, *Stratégie de sécurité sous Windows*.

Les privilèges SYSDBA et SYSOPER permettent de se connecter à Oracle avec plus de privilèges. Ils sont décrits au chapitre évoqué précédemment.

L'utilisation de SQL*Plus pour créer, démarrer, arrêter ou restaurer une base Oracle nécessite obligatoirement l'une (ou les deux) conditions suivantes :

- être connecté sous un utilisateur Windows possédant des privilèges OS supplémentaires,
- être un utilisateur Oracle ayant reçu les privilèges SYSDBA ou SYSOPER qui sont vérifiés dans un « fichier mot de passe » conservé à l'extérieur de la base.

Pour un utilisateur, les privilèges Oracle SYSOPER et SYSDBA permettent les actions suivantes :

- CREATE DATABASE ;
- STARTUP ;
- SHUTDOWN ;
- ALTER DATABASE OPEN / MOUNT ;
- ALTER DATABASE BACKUP CONTROLFILE ;
- ALTER TABLESPACE BEGIN / END BACKUP ;

- ARCHIVE LOG ;
- RECOVER.

Certaines de ces commandes sont étudiées aux chapitres 14 et 10, *Création d'une base Oracle 10g* et *Démarrer et arrêter une base Oracle 10g*.

Pour se connecter en mode administration, les syntaxes possibles sont les suivantes. Elles sont décrites au chapitre 24, *Stratégie de sécurité sous Windows*.

```
C:\> sqlplus /nolog

# connexion avec un utilisateur ayant reçu le privilège SYSDBA
connect utilisateur/passwd as SYSDBA

# connexion avec un utilisateur ayant reçu le privilège SYSOPER
connect utilisateur/passwd as SYSOPER

# connexion sous le privilège SYSDBA ou SYSOPER
# pour cette syntaxe, l'utilisateur doit avoir reçu un pouvoir
# supplémentaire hérité du système d'exploitation
connect / as SYSDBA
connect / as SYSOPER

# idem mais en client-serveur
# (nécessite l'utilisation d'un « fichier mot de passe »)
# sur le serveur cible.
connect utilisateur/passwd@alias as SYSDBA
connect utilisateur/passwd@alias as SYSOPER
```

L'usage de l'ancienne syntaxe CONNECT INTERNAL est interdite. Il faut maintenant utiliser l'une des syntaxes précédentes.

Connexion à SQL*Plus en mode « utilisation »

Pour exécuter des ordres SQL autres que ceux mentionnés au paragraphe précédent, la syntaxe de connexion est plus simple. Utilisez la commande suivante pour lancer SQL*Plus :

```
■ sqlplus
```

Un utilisateur Oracle et son mot de passe vous sont alors demandés. Vous pouvez aussi saisir des options lors du lancement de SQL*Plus :

```
■ C:\> sqlplus -S utilisateur/mot_de_passe@base_cible @fichier_sql
```

- -S est un mode « silencieux » ;
- l'utilisateur et son mot de passe peuvent être suivis de l'alias Oracle Net de la base cible à laquelle vous souhaitez vous connecter. Dans ce cas, le signe @ est accolé au mot de passe ;

- un fichier de commande contenant des ordres SQL, des commandes SQL*Plus et PL/SQL peut être automatiquement lancé, un espace doit alors précéder le signe @ ;
- Vous pouvez aussi lancer SQL*Plus en indiquant qu'il doit se comporter comme une version précédente, en indiquant la version souhaitée. Vous aurez ainsi une garantie supplémentaire pour vos scripts existants.

Notez la différence entre les deux exemples précédents :

sqlplus utilisateur/pwd@base_cible se connecte à une base de donnée particulière ;

sqlplus utilisateur/pwd@fichier.sql exécute automatiquement le fichier de commande cité sur la base indiquée par la variable d'environnement ORACLE_SID.

Vous pouvez aussi utiliser les syntaxes :

```
sqlplus /nolog
connect utilisateur/mot_de_passe
connect utilisateur/mot_de_passe@alias_oracle_net
connect utilisateur/mot_de_passe@fichier_a_executer
```

Pour accéder à l'aide décrivant l'ensemble des syntaxes accessibles lors du lancement de SQL*Plus, il faut appeler :

```
sqlplus -
```

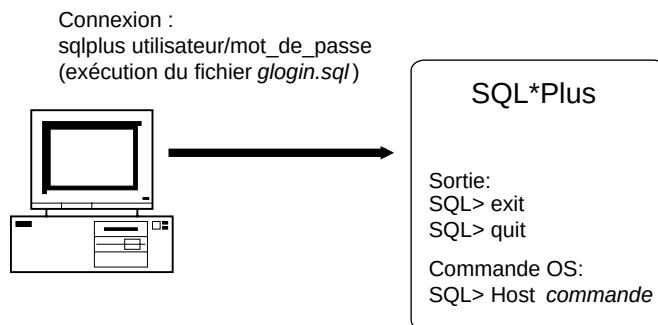
La commande EXIT vous permet de sortir de SQL*Plus.

Interactions entre SQL*Plus et son environnement

La figure suivante décrit les liens entre SQL*Plus et son environnement.

Figure 18-4

*Interactions entre SQL*Plus
et son environnement*



Les grandes caractéristiques sont les suivantes :

- lors du lancement de SQL*Plus, un nom d'utilisateur, son mot de passe et la base de donnée cible vous sont demandés. Sous certaines conditions, le mot de passe et la base cible sont optionnels ;

- lors de la connexion à la base Oracle 10g, le fichier `glogin.sql` est exécuté. Ce fichier, situé sur la machine qui héberge le code exécutable de SQL*Plus (le serveur si vous êtes sur le serveur, un poste Windows si vous êtes en client-serveur), peut contenir toutes sortes d'ordres SQL ;
- par défaut, l'invite d'une session SQL*Plus est `SQL>` ;
- à partir de la session SQL*Plus, vous pouvez lancer des commandes OS par la commande `HOST`. Celle-ci ne termine pas votre session SQL*Plus qui peut être bloquée ou, au choix, active en attendant la fin de l'exécution de la commande OS ;
- vous pouvez vous déconnecter de la base Oracle cible tout en restant dans SQL*Plus au moyen de la commande `DISC` (`DISCONNECT`) ;
- pour se connecter à un autre compte Oracle ou à une autre base de données, utilisez la commande `CONNECT` ;
- pour se déconnecter et terminer une session SQL*Plus, utilisez `EXIT` ou `QUIT`.

Les différentes commandes de SQL*Plus

Différentes commandes peuvent être lancées depuis l'interface SQL*Plus. Par leur diversité, interrogation, débogage, éditeur de texte..., il est souvent facile de les confondre. Les commandes SQL*Plus peuvent être regroupées par famille :

- des ordres SQL ;
- des ordres de création, démarrage, arrêt d'une base ;
- des commandes de type « éditeur de texte » ;
- des commandes de contrôle d'environnement :
- contrôle d'entrée/sortie ;
- dialogue avec l'utilisateur ;
- formatage et présentation d'éditations ;
- définition de l'environnement ;
- contrôle de l'exécution.

Installation d'un jeu d'essai

Si la base de données a été créée par défaut, l'utilisateur SCOTT et son jeu d'essai ont été installés.

Si vous souhaitez placer ce jeu d'essai sur une nouvelle base de données (ou sous un autre compte utilisateur), créez cet utilisateur et exécutez le script sous le compte SYSTEM, en ayant préalablement créé l'utilisateur SCOTT :

```
#sous l'utilisateur SYSTEM
create user SCOTT identified by TIGER;
```

```
# se connecter sous SCOTT
connect SCOTT/TIGER
# Lancer le script
ORACLE_HOME\RDBMS\ADMIN\SCOTT.SQL
```

Chargement de l'aide en ligne SQL*Plus

Sous SQL*Plus, la commande *Help* vous permet de disposer en ligne des principales syntaxes SQL. Pour installer cette aide, il faut définir la variable locale `SYSTEM_PASS` contenant l'utilisateur et le mot de passe administrateur et lancer un script d'installation :

```
C:\> cd ORACLE_HOME\bin
C:\> set SYSTEM_PASS=system/manager
C:\> helpins
```

Que signifie le message « *Product User Profile is not loaded* » ?

Ce message apparaît lors du lancement de SQL*Plus dans certaines circonstances.

SQL*Plus est un outil très puissant. Vous pouvez en limiter l'usage grâce à la table `PRODUCT_USER_PROFILE`, dans laquelle vous précisez les utilisateurs qui peuvent se connecter à SQL*Plus et leurs droits d'utilisation du produit.

Si la base de données a été créée par défaut, ou si vous avez sélectionné l'option *Create Database Object* lors de l'installation de SQL*Plus, cette table a été établie automatiquement sous le compte de l'administrateur `SYSTEM`.

Si une table d'aide n'a pas été créée en même temps que la base, il faut la créer pour supprimer le message « *Product User Profile is not loaded* » qui apparaît à chaque lancement de SQL*Plus. Pour cela, la variable locale `SYSTEM_PASS` doit contenir l'utilisateur et le mot de passe administrateur :

```
C:\> cd ORACLE_HOME\bin
C:\> set SYSTEM_PASS=system/manager
C:\> pupbld
```

Le fichier `ORACLE_HOME\sqlplus\admin\pupbld.sql` est alors lancé.

Les ordres SQL et les commandes SQL*Plus

Sous SQL*Plus, il faut différencier ces types de commande :

- les ordres SQL qui sont adressés à la base de données ;
- les commandes SQL*Plus qui servent à gérer l'affichage, la présentation, en un mot à « habiller » les ordres SQL dans un environnement de travail plus convivial.

Pour chacune de ces commandes, la syntaxe de lancement est différente :

- un ordre SQL est toujours terminé par le signe « ; » ou le signe « / » qui déclenche l'exécution de l'ordre ;
- une commande SQL*Plus est exécutée sans caractère de fin particulier.

Par exemple, l'ordre SQL suivant (qui peut être saisi sur plusieurs lignes) est exécuté par le signe « ; » :

```
SQL> select * from scott.dept
      2  where deptno = 10
      3  ;
```

DEPTNO	DNAME	LOC
10	ACCOUNTING	NEW YORK

Si aucun caractère d'exécution ne vient terminer l'ordre SQL, celui-ci est présent dans le buffer de commande de SQL*Plus, mais n'est pas exécuté.

Dans l'exemple précédent, si l'ordre SQL s'étend sur plusieurs lignes, les numéros des lignes sont indiqués. C'est la fameuse « erreur 2 » que de nombreux utilisateurs de SQL*Plus signalaient au support Oracle, en voyant s'afficher « 2 » à la fin de la saisie d'un ordre SQL. Il suffit alors d'ajouter « ; » pour lancer l'exécution de l'ordre SQL courant.

Dans le cas d'une commande SQL*Plus, aucun caractère particulier n'est nécessaire :

```
SQL> show user
USER is "SYSTEM"
```

Si vous lancez successivement plusieurs commandes SQL et PL/SQL, seul le dernier ordre SQL reste dans le buffer SQL de SQL*Plus. La commande SQL*Plus `list` ou `l` permet de visualiser la dernière commande SQL saisie et la commande `run` ou `r` exécute l'ordre SQL présent dans le buffer :

```
SQL> l
SQL> select * from scott.dept
      2  where deptno = 10
      3  *
SQL> r
```

DEPTNO	DNAME	LOC
10	ACCOUNTING	NEW YORK

Utilisation de SQL*Plus avec PL/SQL

Dans le chapitre 17, *Programmer avec PL/SQL*, nous présentons l'utilisation de SQL*Plus pour réaliser des procédures, des fonctions, des triggers et des packages en langage PL/SQL.

Utiliser un éditeur de texte pour modifier vos ordres SQL

Comment faire pour modifier un ordre SQL comportant des erreurs de syntaxe ? SQL*Plus possède un éditeur de texte en mode ligne très rudimentaire, qui se rapproche de l'antique « edlin », sous DOS ! Nous vous déconseillons fortement de l'utiliser...

À la place, la commande `ed` pour edit permet d'appeler l'éditeur de texte de votre choix et y place l'ordre SQL présent dans le buffer courant. Lors de la sortie de l'éditeur, l'ordre SQL modifié sera rétabli dans le buffer courant.

Pour préciser l'éditeur de texte de votre choix (ici notepad), utilisez la commande SQL*Plus :

```
SQL> def _editor = notepad
```

Attention, l'éditeur choisi doit être un éditeur en mode ascii, à l'exclusion des logiciels de traitement de texte qui incorporent leurs caractères de contrôle dans le texte ascii.

Les commandes SQL*Plus étant conservées jusqu'à la sortie de l'outil, nous vous conseillons de positionner la commande précédente dans le fichier `glogin.sql`. L'éditeur appelé par `ed` sera alors déterminé pour chaque lancement de SQL*Plus. Ce fichier se trouve dans `ORACLE_HOME\sqlplus\admin\glogin.sql`.

Créer un fichier de commandes SQL

L'utilisation de `ed` permet aussi de travailler avec des fichiers texte contenant à la fois des ordres SQL et des commandes SQL*Plus.

```
SQL> ed mon_fichier.sql
```

La saisie de l'extension `.sql` n'est pas obligatoire.

Par exemple, ce fichier peut contenir une suite d'ordres tels que :

```
--  
-- Fichier contenant des ordres SQL et SQL*Plus  
--  
show user  
  
select * from scott.dept  
where deptno = 10  
;  
select ename, job, sal from scott.emp  
where deptno = 10  
/
```

Exécuter un fichier de commande à partir de SQL*Plus

Exécutez le fichier de commande précédent par la commande suivante :

```
start mon_fichier.sql
```

Pour notre exemple, le résultat est alors :

```
SQL> start mon_fichier.sql  
  
USER is "SYSTEM"
```


DEPTNO	DNAME	LOC
10	ACCOUNTING	NEW YORK

ENAME	JOB	SAL
CLARK	MANAGER	2450
KING	PRESIDENT	5000
MILLER	CLERK	1300

SQL*Plus vous propose tout un ensemble de commandes pour améliorer la présentation des données obtenues. Nous vous les présentons ci-après.

Mesurer la durée de vos ordres SQL

L'option SQL*Plus SET TIMING ON|OFF permet de mesurer la durée d'exécution de vos ordres SQL (en millisecondes). Attention à son interprétation, car le temps affiché mesure la durée de l'ordre SQL ainsi que le transport et l'affichage des enregistrements retournés par un ordre SELECT.

```
set timing on

select ename, job, sal from scott.emp
where deptno = 10
/

ENAME      JOB      SAL
-----
CLARK      MANAGER    2450
KING       PRESIDENT  5000
MILLER     CLERK      1300

real: 520
set timing off
```

Exécuter une commande OS

@nom_de_fichier	Exécute le fichier spécifié.
@@nom_de_fichier	Si un fichier exécuté par @nom_de_fichier doit lui aussi lancer ses propres fichiers, ils seront appelés par @@nom_de_fichier.
START nom_de_fichier	Exécute le fichier spécifié.

La commande HOST permet de lancer tout type de commande ou de programme présent sur la machine où s'exécute SQL*Plus. Suivant les versions de SQL*Plus, il est possible grâce à HOST d'exécuter en totalité une commande OS, puis de revenir dans SQL*Plus, ou de lancer cette commande et de revenir dans SQL*Plus sans attendre la fin de l'exécution.

```
host commande_à_lancer paramètre1 paramètre2 ...
```

Interrompre un ordre SQL

Pour interrompre un ordre SQL en cours (par exemple un ordre SELECT qui rapatrie trop d'enregistrements), on peut mettre un terme à l'affichage écran en appuyant sur les touches [Ctrl]+[c] pour les systèmes BSD, ou [Delete] pour les System V.

Inclure des commentaires

-- texte	Utilisé pour inclure des commentaires dans vos ordres SQL. Le texte qui suit -- est alors ignoré lors du traitement de l'ordre SQL. Le commentaire se poursuit jusqu'à la fin de la ligne courante.
/* texte */	Le texte encadré est ignoré même s'il s'étend sur plusieurs lignes.
REM texte	Remarque. Utilisé pour inclure des commentaires dans vos ordres SQL. Doit obligatoirement être situé en début de ligne et ne concerne que la ligne courante.

Écrire des commandes interactives avec SQL*Plus

■ ACCEPT variable PROMPT texte

Cette commande permet de définir une variable et d'afficher un texte avant sa saisie. La variable saisie peut être utilisée dans vos ordres SQL, préfixée du signe &.

```
SQL> ACCEPT v_nom PROMPT 'Recherche utilisateur : '
```

```
Recherche utilisateur : MILLER
```

```
old 2: where upper(ename) = '&v_nom'
```

```
new 2: where upper(ename) = 'MILLER'
```

```
SQL>SELECT ename, job, sal FROM emp
```

```
2 WHERE UPPER(ename) = UPPER( '&v_nom')
```

```
3 /
```

```

ENAME      JOB          SAL
-----
MILLER     CLERK          1300

```

La variable avant et après substitution de la valeur est affichée. Vous pouvez bloquer cet affichage :

```
SQL> SET VERIFY OFF
```

```
SQL> ACCEPT v_nom PROMPT 'Recherche utilisateur : '
```

```
Recherche utilisateur : MILLER
```

```
SQL>SELECT ename, job, sal FROM emp
```

```
2 WHERE UPPER(ename) = UPPER( '&v_nom')
```

```
3 /
```

```

ENAME      JOB          SAL
-----
MILLER     CLERK          1300

```

La commande suivante définit une variable sous SQL*Plus sans qu'il faille en saisir manuellement le contenu :

```
SQL> DEFINE v_nom MILLER

SQL>SELECT ename, job, sal FROM emp
2  WHERE UPPER(ename) = UPPER( '&v_nom')
3  /
```

ENAME	JOB	SAL
MILLER	CLERK	1300

L'utilisation de variables dans vos programmes SQL*Plus permet une meilleure lisibilité ainsi qu'une réutilisation du code.

Décrire la structure des tables et des vues

La possibilité de décrire les tables et les vues contenues dans la base de données est une option très intéressante de SQL*Plus. Son utilisation est simple : seules les tables et les vues auxquelles vous avez accès en lecture sont accessibles par DESC. Par exemple, pour connaître la structure de la table EMP, propriété de l'utilisateur SCOTT :

```
SQL> DESC scott.emp

Name                               Null?    Type
-----
EMPNO                               NOT NULL NUMBER(4)
ENAME                               VARCHAR2(10)
JOB                                 VARCHAR2(9)
MGR                                 NUMBER(4)
HIREDATE                           DATE
SAL                                 NUMBER(7,2)
COMM                                NUMBER(7,2)
DEPTNO                             NUMBER(2)
```

Améliorer la présentation de vos résultats

SQL*Plus n'est pas un outil destiné à produire des éditions perfectionnées comme peuvent le faire des outils spécialisés. Pourtant, un certain nombre de commandes rend l'environnement de travail plus convivial. Il est possible de réaliser des éditions minimalistes avec cet outil.

Effectuer des additions sur des colonnes

Pour effectuer une addition sur une colonne, il faut définir le niveau de rupture souhaité et la colonne de la table sur laquelle s'effectue l'addition.

```
SQL>BREAK ON REPORT
SQL>COMPUTE SUM OF sal ON REPORT
SQL>
SQL>select ename, sal from scott.emp
  2  where deptno = 10
  3  /
```

ENAME	SAL
CLARK	2450
KING	5000
MILLER	1300
sum	8750

```
SQL>CLEAR BREAK
SQL>CLEAR COMPUTE
```

Afficher un titre avant vos ordres SQL

TTITLE est utilisé pour afficher un titre avant chaque ordre SQL. Il permet de centrer le titre (gauche, droite, milieu) et de jouer avec les polices de caractère (gras, italique...).

```
SQL> -- Le titre est placé à gauche (LEFT)
SQL> -- SKIP 1 saute une ligne
SQL> -- le caractère « - » en fin de ligne indique que TTITLE continue
SQL>
SQL>TTITLE LEFT '#####' skip 1 -
> LEFT 'Employés et salaire' skip 1 -
> LEFT '#####'
SQL>
SQL> -- L'exécution de l'ordre SQL provoque l'affichage du titre
SQL>
SQL>select ename, sal from scott.emp
  2  where deptno = 10
  3  /
```

```
#####
Employés et salaire
#####
```

ENAME	SAL
CLARK	2450
KING	5000
MILLER	1300

```
SQL> -- annule l'affichage d'un titre pour les prochains ordres SQL
SQL> TTITLE ''
```

Enregistrer dans un fichier texte

Il est possible d'enregistrer dans un fichier tout ce qui se passe à l'écran sous SQL*Plus.

```
SQL> SPOOL fichier_texte.lis
.....
SQL> SPOOL OFF -- force l'écriture dans le fichier spool.
```

L'éditeur *ED fichier_texte.lis* vous permet d'éditer le résultat obtenu.

Afficher un commentaire

La commande **PROMPT** permet d'afficher un texte lors de l'exécution de vos ordres SQL. Elle sert notamment à commenter le résultat lors de l'exécution d'un script SQL. Ainsi :

```
--
-- Fichier script SQL
--
PROMPT Début du script ...
...
```

donnera comme résultat :

```
Début du script ...
```

Notez la différence entre les commentaires destinés à améliorer la lisibilité du script et *PROMPT* qui sert à en commenter le résultat.

Afficher / cacher les ordres SQL avant leur exécution

Vous pouvez afficher/cacher le texte des ordres SQL avant leur exécution. Ainsi :

```
--
-- Fichier script SQL
--
SET ECHO ON

select ename, sal from scott.emp
where deptno = 10
/
...
```

affichera comme résultat :

```
SET ECHO ON
SQL>select ename, sal from scott.emp
  2  where deptno = 10
  3  /

ENAME                SAL
```

```

-----
CLARK          2450
KING           5000
MILLER         1300

```

L'utilisation de SET ECHO OFF cachera le texte de l'ordre SQL. Ainsi :

```

--
-- Fichier script SQL
--
SET ECHO OFF

select ename, sal from scott.emp
where deptno = 10
/

```

aura comme résultat :

```

SET ECHO OFF

ENAME          SAL
-----
CLARK          2450
KING           5000
MILLER         1300

```

Ne pas afficher le nombre d'enregistrements renvoyés

Par défaut, SQL*Plus affiche le nombre d'enregistrements renvoyés dès que ceux-ci sont supérieurs à cinq. Pour que cette information ne s'affiche plus, utilisez SET FEEDBACK OFF.

```

SQL> SET FEEDBACK ON
SQL> select ename from emp;

ENAME
-----
SMITH
MARTIN
CLARK
SCOTT
KING
TURNER
ADAMS
JAMES
FORD
MILLER

10 rows selected.

SQL> SET FEEDBACK OFF
SQL> select ename from emp;

```

```
ENAME
-----
SMITH
MARTIN
CLARK
SCOTT
KING
TURNER
ADAMS
JAMES
FORD
MILLER
SQL>
```

Les commandes de présentation SET

Un grand nombre de commandes *SET variable paramètres* est proposé par SQL*Plus. Nous en avons déjà rencontré quelques-unes :

```
SET ECHO [ON | OFF]
SET FEEDBACK [ON | OFF]
SET TIMING [ON | OFF]
...
```

Reportez-vous à la documentation SQL*Plus pour disposer de l'ensemble des commandes SET.

Création de requêtes dynamiques

Vous pouvez utiliser une option très puissante de SQL*Plus pour créer automatiquement des ordres SQL avec des données issues de la base Oracle. Vous souhaitez, par exemple, obtenir la description de l'ensemble des tables et des vues qui sont votre propriété.

La première étape consiste à créer un ordre SQL qui recherche la liste des vues et des tables. Ces données sont précédées et suivies de textes, qui serviront à créer une commande SQL complète encadrant les données issues de la base.

La seconde étape exécutera ces ordres SQL, préalablement conservés dans un fichier de commandes.

Première étape : fichier de commandes

Ce fichier déclenche automatiquement les autres étapes.

```
--
-- Création d'une requête SQL dynamique
--

-- Gestion de la présentation
SET ECHO OFF
SET FEEDBACK OFF
```

```

SET HEADING OFF

-- Création du fichier SQL qui sera exécuté
SPOOL fichier_temporaire.sql

-- Bannière destinée à figurer dans le fichier de commandes
PROMPT SET ECHO ON
PROMPT REM #####
PROMPT REM Fichier créé automatiquement
PROMPT REM #####
-- Fichier qui contiendra le resultat
PROMPT SPOOL resultat.lis
select 'DESC '||object_name from user_objects
where object_type in ('TABLE','VIEW')
/
-- arrêt de l'enregistrement du fichier résultat
PROMPT SPOOL OFF

spool off
START fichier_temporaire.sql

```

Deuxième étape : le contenu du fichier temporaire

Le fichier de commande *fichier_temporaire.sql* est créé automatiquement lors de l'exécution du fichier précédent. La liste des tables et des vues provient directement de la base Oracle 10g.

```

SET ECHO ON
REM #####
REM Fichier créé automatiquement
REM #####
SPOOL resultat.lis

DESC DEPT
DESC EMP
DESC TOTO
DESC TRACE_SALAIRE

SPOOL OFF

```

À la place de DESC NOM_TABLE, vous pouvez placer n'importe quel ordre SQL valide en plaçant à la fin le signe « ; ».

Troisième étape : exécuter le fichier de commandes temporaire

Cette étape est effectuée à la fin du script de commandes principal par :

```

START fichier_temporaire.sql

```


Le résultat du descriptif des tables est enregistré dans le fichier *resultat.lis* dont le contenu est le suivant :

```
SQL>
SQL> DESC BONUS
      Name                                Null?    Type
-----
      ENAME                                VARCHAR2(10)
      JOB                                  VARCHAR2(9)
      SAL                                  NUMBER
      COMM                                NUMBER

SQL> DESC DEPT
      Name                                Null?    Type
-----
      DEPTNO                               NOT NULL NUMBER(2)
      DNAME                                VARCHAR2(14)
      LOC                                  VARCHAR2(13)

SQL> DESC EMP
      Name                                Null?    Type
-----
      EMPNO                               NOT NULL NUMBER(4)
      ENAME                                VARCHAR2(10)
      JOB                                  VARCHAR2(9)
      MGR                                  NUMBER(4)
      HIREDATE                             DATE
      SAL                                  NUMBER(7,2)
      COMM                                NUMBER(7,2)
      DEPTNO                               NUMBER(2)
      ...

SQL> SPPOOL OFF
```

Autre requête dynamique

Les exemples d'utilisation des requêtes dynamiques sont très nombreux. Ici, nous utilisons les requêtes dynamiques pour accomplir une tâche d'administration. Oracle 10g utilise des statistiques pour déterminer la manière dont son moteur SQL traite les requêtes qui lui sont soumises. Les statistiques doivent être mises à jour régulièrement. La commande :

```
ANALYZE [TABLE|INDEX] nom COMPUTE STATISTICS...
```

analyse tables et index et met à jour le dictionnaire interne d'Oracle. Rechercher et saisir l'ensemble des tables et index de la base de données, est une opération longue et fastidieuse. Or, le dictionnaire de données d'Oracle9 nous fournit la liste des tables et index contenus. D'où le programme SQL*Plus :

```
--
-- Création d'une requête SQL dynamique:
-- Analyse des tables et index
--

-- Modifications de l'environnement SQL*Plus
SET ECHO OFF
SET FEEDBACK OFF
SET HEADING OFF
SET PAGESIZE 100

-- Création du fichier SQL qui sera exécuté
SPOOL analyze.SQL

select 'analyze '||object_type||' '||object_name
      ||' estimate statistics sample 20 percent;'
from user_objects
where object_type in ('TABLE','INDEX')
order by object_type
/
spool off
-- Modifications de l'environnement SQL*Plus
SET ECHO ON
SET ECHO OFF
SET FEEDBACK ON
SET HEADING ON
SET PAGESIZE 20
-- Exécution de l'analyse de tables et des index
PROMPT #####
PROMPT Début de l'analyse des tables et index
PROMPT #####
START analyze.sql
PROMPT #####
PROMPT Fin d'analyse
PROMPT #####
-- On peut supprimer le fichier temporaire (ici en commentaire)
-- HOST rm analyze.sql
```

Le fichier analyze.sql contient :

```
analyze INDEX SYS_C001227 estimate statistics sample 20 percent;
analyze INDEX SYS_C0012271 estimate statistics sample 20 percent;
analyze INDEX SYS_C001229 estimate statistics sample 20 percent;
analyze TABLE BONUS estimate statistics sample 20 percent;
analyze TABLE DEPT estimate statistics sample 20 percent;
analyze TABLE TOTO estimate statistics sample 20 percent;
analyze TABLE TRACE_SALAIRE estimate statistics sample 20 percent;
analyze TABLE EMP estimate statistics sample 20 percent;
analyze TABLE MLOG$_EMP estimate statistics sample 20 percent;
analyze TABLE SALGRADE estimate statistics sample 20 percent;
analyze TABLE SNAP$_EMP_SNAPSHOT estimate statistics sample 20 percent;
```

Ce fichier est lancé automatiquement à l'issue du script principal.

La commande *HOST rm analyse.sql* peut être utilisée à l'issue du script pour supprimer toute trace du fichier temporaire.

Les outils complémentaires

En complément de la présentation détaillée des outils standard fournis par Oracle, nous avons choisi de vous présenter d'autres outils qui répondent à des besoins spécifiques, à savoir :

- des besoins non satisfaits par les outils standard Oracle ;
- des offres innovatrices qui comblent des lacunes.

Le principal reproche que l'on puisse adresser aux outils standard d'Oracle est leur manque de convivialité et leur accès complexe. Ce sont des produits en mode caractère destinés à des informaticiens. Ils ont l'avantage d'être présents sur tout type d'environnement, d'être performants, stables et d'offrir des possibilités étendues. Nous sommes loin d'un univers « 100 % graphique », mais ils sont puissants et fiables.

Quelle complémentarité attendre d'Oracle Enterprise Manager ?

Le chapitre 25, *Oracle Enterprise Manager*, est consacré à ce sujet. Souvent, les outils incorporés dans Oracle Enterprise Manager proposent un habillage graphique plus convivial que celui des outils Oracle fournis en standard.

Les modules proposés par Oracle Enterprise Manager proposent toutes sortes de services. Ils administrent vos bases de données, permettent d'ordonnancer et de planifier des tâches sur les systèmes distants, d'arrêter et de démarrer une base, de l'administrer, de créer une table, un utilisateur, etc. C'est à la fois trop et pas assez. Trop, pour un utilisateur néophyte qui est perdu devant un ensemble d'écrans dont il ne comprend pas toujours le sens et l'utilisation (de plus, il est difficile de gérer des profils d'utilisateur). Pas assez, si vous êtes un utilisateur professionnel souhaitant disposer de l'ensemble des possibilités offertes par la base Oracle 10g. Par ailleurs, cédant à la mode du tout graphique, certains modules sont proches du gadget et n'ont pas de finalité précise.

Cependant, Oracle Enterprise Manager est livré gratuitement et il convient de l'étudier soigneusement avant d'avoir recours à l'achat d'autres produits. De même, il est nécessaire de le mettre en concurrence avec des outils spécialisés avant d'acquérir des modules supplémentaires (et facturés) d'Oracle Enterprise Manager.

Les limites d'Oracle Export/Import ?

Dans la majorité des cas, tant en termes de fonctionnalités que de performance, l'Export/Import est très satisfaisant. Si vous manipulez de très gros volumes de données, des éditeurs spécialisés comme ex-PLATINUM (acheté depuis par Computer Associates) proposent des outils complémentaires.

Les limites d'Oracle Data Pump ?

Oracle Data Pump Export et Import apportent de nouvelles possibilités aux traditionnels Export et Import. Ils sont annoncés comme une véritable avancée en terme de performance. Sur de petites configurations (serveur mono processeur, faible volume de données), l'utilisation de Data Pump peut s'avérer plus longue qu'Export ou Import. Sur de gros serveurs, avec une forte volumétrie de données (on parle alors en centaines de Go...) les gains seront certainement probants. Je vous conseille d'effectuer préalablement un test pour comparer les deux possibilités dans votre contexte.

Les limites d'Oracle SQL*Loader ?

De même que pour l'Export/Import, Oracle SQL*Loader est très satisfaisant, laissant peu d'opportunités à d'autres outils payants.

Quelles sont les limites d'Oracle SQL*Plus ?

Il faut dissocier l'usage de SQL*Plus pour créer, démarrer, arrêter une base et celui plus large de la « programmation SQL ». Dans le premier cas, SQL*Plus et Oracle Enterprise Manager sont les seuls outils répondant à cet usage. Pour programmer en SQL, le marché des outils tiers est florissant.

Oracle SQL*Plus est un outil de base indispensable, mais qui montre vite ses limites si vous le destinez à des programmeurs habitués à des environnements de développements graphiques performants. Oracle Enterprise Manager propose SQL*Worksheet, mais ce dernier n'est pas à la hauteur de ses concurrents. Parmi les éditeurs fournissant des outils de substitution, axés sur la gestion et la programmation d'une base Oracle, nous pouvons citer :

- Oracle, qui propose ses propres outils de développement,
- Platinum Technology, <http://www.cai.com>,
- Quests Software, <http://www.quests.com> qui a acheté l'excellent TOAD,
- Tora, <http://www.globecom.se/tora> un très bon outil qui existe sous Windows et Linux,
- DataBee, <http://www.databee.com>, un outil très bien conçu autour des opérations d'export et d'import,
- SQL-Programmer édité par BMC, <http://www.bmc.com>,
- SQL*Object Builder de IDB, <http://www.idb-consulting.fr>.

Quels autres outils pour quels besoins ?

Pour découvrir le marché des outils tiers gravitant autour d'Oracle, la meilleure source d'informations sont les publicités qui paraissent dans *Oracle magazine*, magazine papier bimensuel à abonnement gratuit. Le formulaire d'abonnement est accessible à l'adresse <http://www.oramag.com>.

C'est un marché de « niche », fortement spécialisé, où des éditeurs proposent des produits très techniques à forte valeur ajoutée. La palette proposée est large et comprend notamment :

- des outils de conversion automatique d'applications Oracle*Forms vers Oracle Developer 2000 ;
- des outils complémentaires opérant autour d'Oracle Application ;
- des outils de réorganisation de base de données ;
- des utilitaires pour automatiser la sécurité d'accès.

Résumé

Oracle 10g est fourni avec un ensemble d'utilitaires puissants, trop souvent méconnus. Ce chapitre, volontairement détaillé, a permis de vous en présenter les champs d'application et d'utilisation ainsi que leurs limites :

- Oracle SQL*Loader pour charger toutes sortes de fichiers dans une base Oracle ;
- Oracle Export/Import pour extraire et transférer des données de base Oracle à base Oracle ;
- Oracle SQL*Plus, l'interface privilégiée pour interagir sur la base de données et pour la création, le démarrage et l'arrêt des bases ;
- des outils tiers, complémentaires aux précédents.

19

Pilotes JDBC et SQLJ

Dans ce chapitre :

- le standard JDBC ;
- les pilotes JDBC d'Oracle ;
- SQLJ, simplicité et contraintes ;
- les nouvelles fonctions.

Dans les architectures d'applications distribuées, les middlewares jouent un rôle fondamental. Ils permettent l'échange de données entre des applications et des bases de données. Pour les applications, une base de données est un espace de stockage performant qui sert à la consultation, à la mise à jour et à l'ajout de données. Les standards JDBC et SQLJ ont été créés dans cette optique : ce sont des middlewares destinés aux programmes Java. Ils permettent aux applications Java autonomes ou à des applets Java de communiquer avec des bases de données. Au chapitre 15, *Oracle Net, le middleware Oracle*, nous avons présenté les différents moyens de communication entre une application de l'environnement Windows et une base Oracle 10g. Outre les pilotes présentés dans ce chapitre, Oracle propose des pilotes JDBC pour se connecter à Oracle 10g à partir de n'importe quel programme Java. Par rapport à ODBC, JDBC ajoute la portabilité de Java.

Ce chapitre vous explique quand et comment utiliser les pilotes JDBC. Par ailleurs, nous aborderons SQLJ pour montrer comment il est possible d'incorporer très facilement des instructions SQL à du code Java.

Principe des pilotes JDBC

Il existe deux types de pilotes JDBC Oracle :

- les pilotes JDBC utilisés par les machines virtuelles Java standard ;
- un pilote JDBC spécifique pour les méthodes Java exécutées par la machine virtuelle Java d'Oracle.

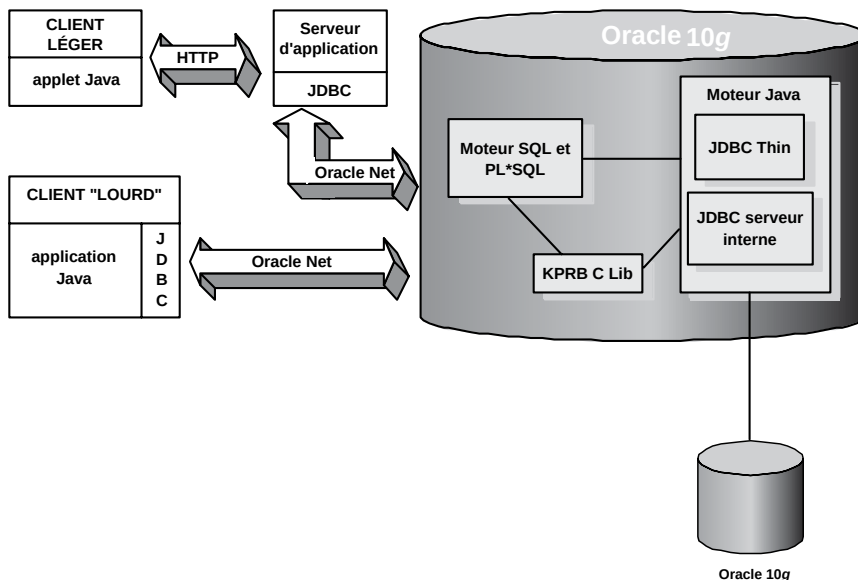


Figure 19-1

Architecture d'une application multinationale

La figure 19-1 montre une architecture dans laquelle JDBC est une interface incontournable. Le modèle présente un déploiement de type client lourd et un déploiement de type client léger. Dans le premier cas, l'application Java accède à la base de données via JDBC, qui repose lui-même sur le protocole Oracle Net. Cette application nécessite un poste de travail dit client lourd car elle réalise des opérations à la fois sur les données et sur l'affichage de ces données. Dans le second cas, le client est, par exemple, un navigateur Web qui communique avec un serveur d'applications selon le protocole HTTP. Le client léger doit uniquement afficher les informations qu'il échange avec le serveur d'applications ; il ne réalise aucun traitement sur ces données. Le serveur d'applications les recueille depuis la base par l'intermédiaire d'une connexion JDBC et exécute les opérations nécessaires sur ces données. Ensuite, il les transmet au client léger qui les présente à son tour à l'utilisateur final. Ces deux catégories de clients, lourd et léger, ont accès à tous les objets stockés dans la base Oracle 10g, via les protocoles HTTP et JDBC : tables, procédures stockées PL/SQL et Java, etc.

JDBC : les pilotes et leur utilisation

JDBC est principalement une bibliothèque de classes Java destinée à faciliter l'accès à des données contenues dans des bases relationnelles. Ce n'est pas une invention d'Oracle, mais une interface standardisée utilisée par de multiples acteurs de l'informatique. Toute la difficulté de cette interface consiste à en saisir le « lego logiciel », les différents composants mis en œuvre par telle ou telle implémentation. Les pilotes JDBC sont en effet multiples et utilisent des « chemins d'accès » très variés.

JDBC a été conçu après ODBC et en reprend bon nombre de caractéristiques : connexion simultanée à plusieurs bases de données, gestion des transactions, interrogations, appel à des procédures stockées, etc. JDBC ne se contente pas d'imiter ODBC : il innove sur des points importants, tels le téléchargement du pilote par le réseau avant son exécution. Le premier pilote JDBC opérationnel, proposé par JavaSoft (la filiale de Sun dédiée aux produits Java), nécessitait l'installation de pilotes ODBC pour fonctionner. Cette situation a heureusement évolué.

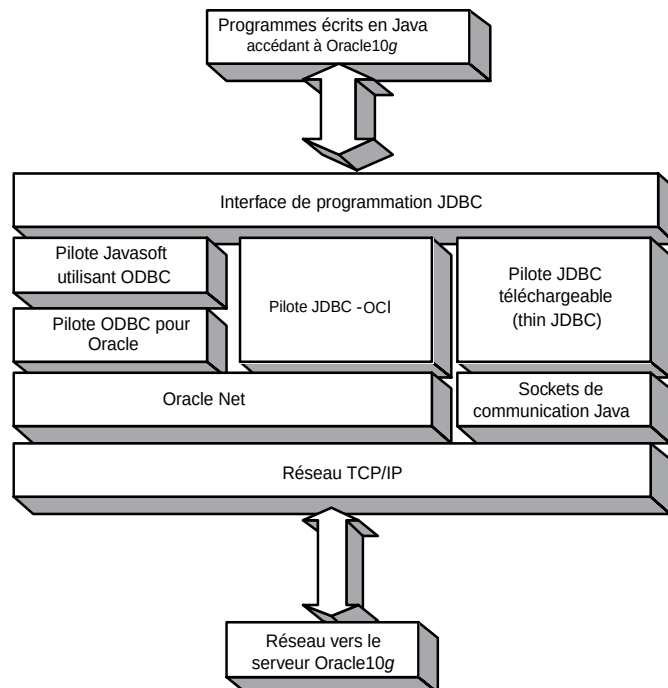
On peut considérer qu'il existe trois types de pilotes JDBC :

- ceux qui réutilisent ODBC ;
- ceux qui se réfèrent à Oracle Net ;
- ceux qui n'ont besoin ni d'ODBC, ni d'Oracle Net.

Ces différents accès sont résumés dans le schéma suivant :

Figure 19-2

*Les différents types
de pilotes JDBC*



La façon dont vous programmez vos applications Java est unique : elle est définie par les spécifications de l'interface de programmation JDBC. À l'inverse, lors de l'écriture de vos programmes, vous pouvez préciser le type de pilote, donc le chemin que vous souhaitez emprunter pour accéder à Oracle 10g.

Comment choisir entre les différents JDBC existants

L'intérêt principal de Java est sa portabilité et les possibilités de téléchargement d'applications qu'il offre. L'avantage majeur que les entreprises recherchent dans ces solutions techniques est avant tout l'absence de déploiement d'applications. Actuellement, il existe des outils de développement très productifs, mais dont l'utilisation est limitée par la difficulté de leur installation sur les postes des utilisateurs. Même si les applications peuvent être téléchargées après modification, les couches techniques de bas niveau comme Oracle Net doivent être installées sur chaque poste utilisateur.

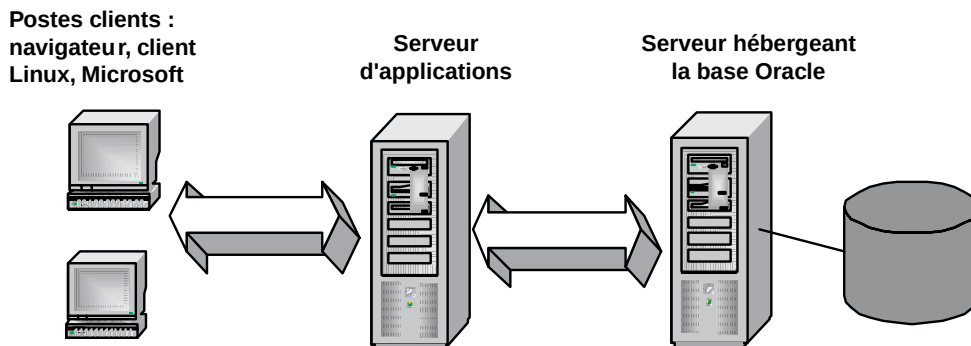


Figure 19-3

Architecture de base

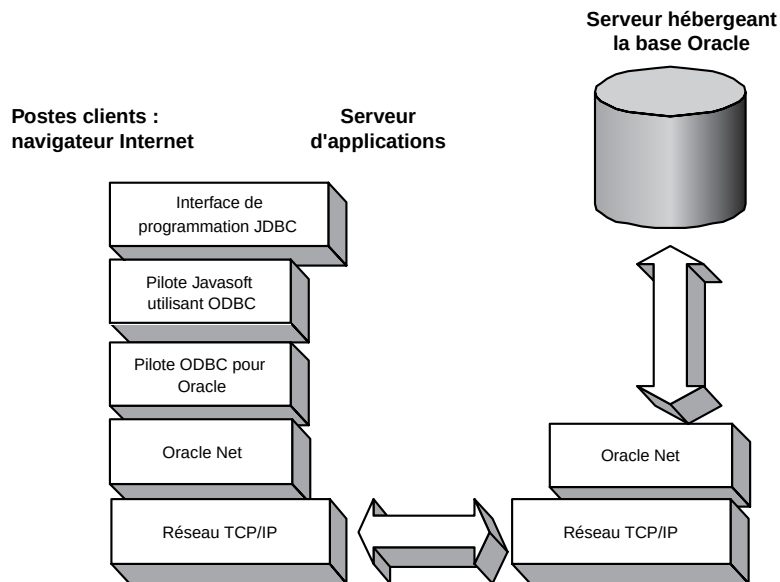
Illustrons nos commentaires au moyen d'une architecture à trois niveaux : un poste client équipé d'un navigateur, un serveur d'applications intermédiaire et un serveur de données hébergeant une base Oracle 10g.

Voyons, selon ces critères, les différentes solutions existantes.

Pilote JavaSoft (ou autre) utilisant ODBC

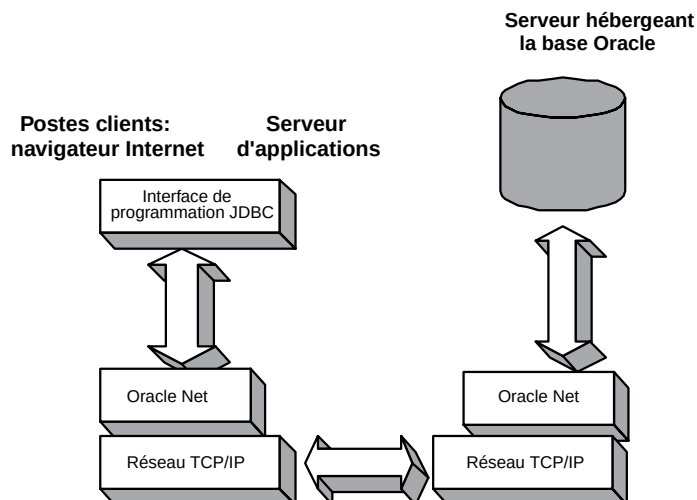
C'est le moins intéressant des trois. Il a comme seul avantage de valider une solution technique novatrice. Pour fonctionner, il faut installer à la fois l'interface JDBC (ajout au langage Java qui interprète les appels JDBC dans vos programmes), un pilote ODBC pour Oracle 10g (plus sa configuration) et enfin le « client » Oracle Net et sa configuration. Cela fait beaucoup !

Ce type de pilote alourdit énormément le poste client. Il convient de le réserver à une validation technique ou à des programmes fonctionnant sur des serveurs d'applications.

**Figure 19-4***Pilote JDBC utilisant ODBC*

Pilote OCI s'appuyant sur Oracle Net

Pour cette architecture, l'installation et la configuration locale de Oracle Net et du pilote JDBC sont nécessaires. Ce pilote se caractérise par une bonne capacité à gérer une charge importante.

**Figure 19-5***Pilote JDBC s'appuyant sur Oracle Net*

La solution technique proposée dans ce cas est fort semblable à celle du client-serveur traditionnel. Deux sortes d'applications peuvent être visées par ce type de pilote. Tout d'abord celles fonctionnant sur un serveur d'applications : le code applicatif mis en œuvre sur le poste client ne gère que l'affichage des données et le code autorisant l'accès aux données fonctionne sur le serveur d'applications. Une seule installation est alors nécessaire, et ce quel que soit le nombre de clients connectés.

L'autre type d'applications concernées sont les applications client ayant un fort besoin d'échanges avec la base Oracle 10g, à l'instar des applications client-serveur actuelles. La lourdeur d'installation d'un parc important de machines client se justifie par les performances.

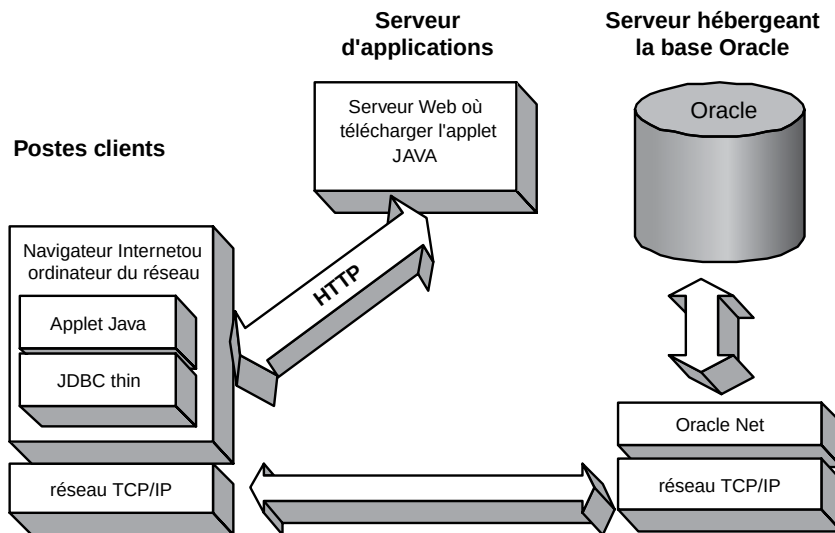
Pilote téléchargeable Thin JDBC

Ce pilote possède un avantage majeur : il ne nécessite pas l'installation d'un logiciel Oracle sur le poste client et peut être téléchargé par le réseau, sous forme d'une applet Java.

Techniquement, le code de ce pilote « émule » Oracle Net et permet une communication directe entre votre poste client et votre base de données Oracle 10g.

Figure 19-6

Téléchargement d'une applet



Les limitations techniques du pilote Thin JDBC concernent son volume de chargement, environ 300 Ko hors application Java, qui le rend difficilement utilisable sur des réseaux à faible débit.

Remarque : pour les « anciens » d'Oracle, le pilote Thin JDBC d'Oracle n'est autre qu'une remise au goût du jour de l'ancien SQL*Net V1 !

Pilote JDBC intégré à la JVM Oracle 10g

Oracle 10g est fourni avec une machine virtuelle Java intégrée au serveur de base de données. Comme toute machine virtuelle Java, la JVM d'Oracle prend en charge le standard JDBC pour offrir une interface de communication entre une base de données et une application Java. Néanmoins, les pilotes JDBC décrits précédemment (JDBC Thin, JDBC OCI) ne sont pas utilisables par les méthodes Java exécutées par la JVM Oracle. Ces méthodes accèdent à la base via le pilote JDBC embarqué qui garantit de meilleures performances que les pilotes externes à Oracle 10g, car il partage les ressources de la base. Il repose sur les bibliothèques natives de la base de données. N'importe quelle méthode Java peut être exécutée dans Oracle 10g : un développeur d'applications trouve un intérêt à déployer dans Oracle 10g les classes Java exploitant les données de la base. Le pilote JDBC embarqué fournit la même API que les pilotes JDBC externes : les instructions JDBC prévues pour s'exécuter en dehors de la base de données sont strictement identiques aux instructions JDBC qui s'exécutent dans Oracle 10g.

Connexion JDBC à l'extérieur d'Oracle 10g

Pour l'ouverture d'une connexion JDBC, les méthodes Java qui s'exécutent à l'extérieur de la base de données doivent spécifier :

- le nom du pilote JDBC ;
- le nom d'utilisateur ;
- le mot de passe de cet utilisateur ;
- les paramètres pour localiser la base de données.

Par exemple, la commande suivante établit une connexion JDBC Thin avec la base ORCL, accessible par le port 1521 du serveur unServeur, sous l'identité SCOTT/TIGER :

```
DriverManager.registerDriver(new oracle.jdbc.driver.OracleDriver());  
connexion conn = DriverManager.getconnexion("<URL>");  
où <URL> = jdbc:oracle:thin:scott/tiger@unServeur:1521:ORCL
```

Connexion JDBC dans la JVM Oracle

Lorsqu'une méthode Java est exécutée par la JVM d'Oracle, son contexte transactionnel est celui de la session courante, la session dans laquelle la méthode est appelée. Au chapitre 20, *Java et Oracle 10g*, nous verrons qu'une méthode Java est accessible depuis des instructions SQL ou PL/SQL.

Imaginons qu'une session SQL*Plus soit ouverte par l'utilisateur SCOTT (mot de passe TIGER). Une fois connecté, SCOTT appelle la procédure stockée *Java unPackage.getNom(3)* qui renvoie le nom de la personne numéro 3. La méthode Java n'a pas besoin de spécifier l'adresse du serveur, le nom de la base, le nom de l'utilisateur, son mot de passe. La méthode Java s'exécute dans la session ouverte par SCOTT.

Une méthode Java côté serveur est donc toujours connectée à la base de données. L'objet connexion est créé à partir de la session en cours avec l'instruction suivante :

```
■ connexion conn = new oracle.jdbc.driver.OracleDriver.defaultconnexion();
```

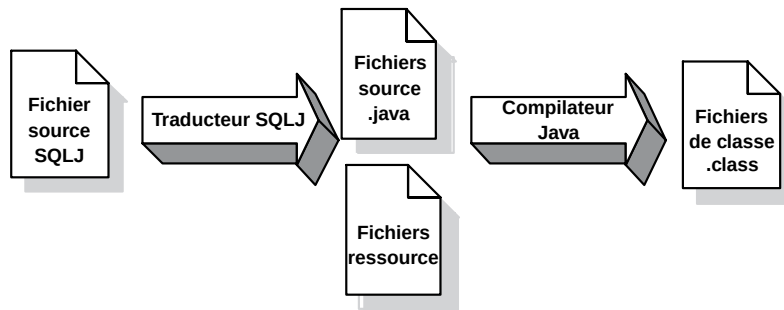
Ensuite, des objets Statement, PreparedStatement ou CallableStatement sont créés à partir de l'objet conn.

SQLJ

Depuis longtemps, il est possible de programmer en langage C, C++, Cobol, etc., pour accéder à Oracle. Dans ce but, il existe des « précompilateurs » adaptés au langage utilisé. La mise en œuvre de SQLJ est simple : une balise particulière, *#sql*, indique l'emplacement des instructions SQL intégrées au code Java. Le fichier source Java est alors doté de l'extension .sqlj, pour indiquer que ce n'est pas un fichier source Java pur. Ce fichier .sqlj doit être traité par un précompilateur, dont la tâche est de traduire les instructions SQL en instructions Java. Il génère des fichiers source Java, qui seront compilés par un compilateur Java standard.

Figure 19-7

Principe de SQLJ



SQLJ est donc un précompilateur Java qui permet d'accéder facilement aux bases de données Oracle 10g.

SQLJ est un standard pour créer des requêtes SQL plus simplement qu'à partir de l'API JDBC. Le développeur n'a pas besoin de maîtriser la complexité de JDBC pour écrire des instructions SQL dans un programme Java. Les lignes de code SQLJ sont constituées par des instructions SQL identiques à celles exécutées par des outils comme SQL*Plus.

La syntaxe est la suivante :

```
#sql { <requête SQL> };
```

Par exemple la commande SQLJ correspondant à :

```
SELECT JOB FROM EMP WHERE EMPNO=3;
```

s'écrit :

```
#sql {SELECT JOB INTO :empFonction FROM EMP WHERE EMPNO=3};
```

SQLJ et JDBC

Comparé à JDBC, SQLJ apporte une grande simplification de la syntaxe :

- le code est plus lisible ;
- le nombre de lignes de code est moindre ;
- la syntaxe SQL des instructions SQLJ est vérifiée pendant la compilation ;
- la correction des erreurs est simplifiée.

Tout programme Java peut exploiter SQLJ, qui peut s'utiliser dans les applications client et dans les procédures stockées Java (triggers, méthodes, EJB, objets CORBA). Une méthode Java peut contenir à la fois des appels JDBC et des instructions SQLJ.

Comme avec JDBC, les paramètres des instructions SQLJ sont des valeurs littérales ou des variables Java. Un avantage de SQLJ consiste en la vérification syntaxique et en la cohérence avec le schéma de la base de données, au moment de la compilation du code. Si une instruction SQLJ appelle par exemple une procédure PL/SQL inexistante, le traducteur SQLJ le signalera.

En revanche, SQLJ nécessite de connaître à l'avance la requête qui sera exécutée, alors que JDBC permet de créer dynamiquement des requêtes puisqu'une requête JDBC est définie par une chaîne de caractères.

L'exemple suivant présente la différence entre un code JDBC et le code SQLJ équivalent. Le code de cet exemple :

- ouvre une connexion dans la base Oracle 10g sous l'identité SCOTT ;
- recherche la fonction de l'employé 7 788 ;
- affiche le résultat de la requête ;
- recherche la fonction exercée par chaque employé ;
- affiche la liste des employés, avec leur fonction.

Exemple utilisant JDBC

Définissons le fichier *SalariesJDBC.java* :

```
import java.sql.*;

public class SalariesJDBC {

    CONNEXION conn;

    public SalariesJDBC() throws SQLException {

        // Enregistrement du pilote JDBC Oracle

        DriverManager.registerDriver(new oracle.jdbc.driver.OracleDriver());
```

```
//Ouverture d'une session dans la base
/*Il faut adapter la chaîne unServeur:1521:orcl
   remplacer unServeur par le nom TCP/IP du serveur de la base
   remplacer 1521 par le port Oracle Net de la base
   remplacer ORCL par le SID de votre base Oracle
*/

try {
    conn = DriverManager.getConnection("jdbc:oracle:thin:@unServeur:1521:orcl",
    ➡"scott","tiger");
}
catch (SQLException exception) {
    System.out.println("Impossible d'ouvrir la session");
    System.err.println(exception) ;
    System.exit(1);
}

}

public void requete() throws SQLException {
    String job;
    String nom;
    int empId = 7788;

    //Recherche du nom et de la fonction du salarié 7 788

    //Préparation de la requête

    PreparedStatement requete = conn.prepareStatement("SELECT ename,job from EMP
    ➡where empno=?");
    requete.setInt(1,empId);

    //Exécution de la requête

    ResultSet resultat = (ResultSet)requete.executeQuery();

    //Affichage des résultats

    while (resultat.next())
    {
        System.out.println("Nom du salarié #" + empId + " : " + resultat.getString(1));
        System.out.println("Fonction du salarié #" + empId + " : " + resultat.getString(2));
    }
    resultat.close();
    requete.close();

    //Liste des fonctions de tous les salariés
    //Préparation de la requête

    requete = conn.prepareStatement("SELECT ename,job from EMP");
```



```

//Exécution de la requête

resultat = (ResultSet)requete.executeQuery();

//Affichage des résultats

System.out.println("Liste des salariés avec leur fonction :");
while (resultat.next()) {
    System.out.println(resultat.getString(1)+" est un "+resultat.getString(2));
}
resultat.close();
requete.close();
}

public static void main (String [] args) throws SQLException {
    SalariesJDBC application= new SalariesJDBC();
    application.requete();
}
}

```

Exemple identique avec SQLJ

Définissons le fichier *Salaries.sqlj* :

```

import sqlj.runtime.ref.*;
import java.sql.*;

public class Salaries {

    /*Définition d'un itérateur qui contient un champ ename et un champ
    job pour recevoir des lignes de la table EMP du schéma SCOTT
    */
    #sql iterator liste(
    String ename,
    String job );

    public Salaries() throws SQLException {

        //Enregistrement du pilote JDBC Oracle

        DriverManager.registerDriver(new oracle.jdbc.driver.OracleDriver());

        //Ouverture d'une session dans la base
        //Définition d'un contexte pour les instructions SQLJ

        try {
            CONNEXION conn = DriverManager.getConnection("jdbc:oracle:thin:@unServeur:1521:orcl",
            ➤"scott","tiger");
            DefaultContext.setDefaultContext(new DefaultContext(conn));
        }
    }
}

```

```
        catch (SQLException exception) {
            System.out.println("Impossible d'ouvrir la session");
            System.err.println(exception) ;
            System.exit(1);
        }
    }

    public void requete() throws SQLException {
        String job;
        String nom;
        int empId = 7788;

        //Recherche du nom et de la fonction du salarié 7 788

        #sql {SELECT ENAME,JOB into :nom,:job from EMP where empno=:empId};
        System.out.println("Nom du salarié #" + empId + " : " + nom);
        System.out.println("Fonction du salarié #" + empId + " : " + job);

        //Liste des fonctions de tous les salariés

        liste uneListe;
        #sql uneListe={SELECT ENAME,JOB from EMP};

        //Affichage des résultats

        System.out.println("Liste des salariés avec leur fonction :");
        while (uneListe.next()) {
            System.out.println(uneListe.ename() + " est un " + uneListe.job());
        }
    }

    public static void main (String [] args) throws SQLException {
        Salaries application= new Salaries();
        application.requete();
    }
}
```

Comparaison du code source JDBC et du code source SQLJ

La différence est nette entre les deux codes source Java. Le premier nécessite :

- la création d'un objet `PreparedStatement` ;
- la création d'un objet `ResultSet` ;
- un appel de méthode `setInt` pour affecter la valeur d'un paramètre de la requête ;
- un appel de méthode `getString` pour récupérer la valeur d'une colonne.

Le second, avec SQLJ, nécessite :

- la création d'une variable INT pour affecter la valeur d'un paramètre de la requête ;
- la création d'une variable STRING pour récupérer la valeur d'une colonne ;
- le codage d'une instruction SQL dans une syntaxe très claire.

En outre, le code SQLJ est identique à l'instruction SQL qui serait exécutée dans SQL*Plus.

Exécution des exemples précédents

Avant d'exécuter les deux applications, il faut adapter la chaîne de connexion à la base de données, c'est-à-dire :

- remplacer unServeur par le nom TCP/IP du serveur de la base ;
- remplacer 1521 par le port Oracle Net de la base ;
- remplacer ORCL par le SID de votre base Oracle.

L'application SalariesJDBC et l'application *Salaries* produisent le même résultat, à savoir :

```
Nom du salarié #7788 : SCOTT
Fonction du salarié #7788 : ANALYST

Liste des salariés avec leur fonction :

SMITH est un CLERK
ALLEN est un SALESMAN
WARD est un SALESMAN
JONES est un MANAGER
MARTIN est un SALESMAN
BLAKE est un MANAGER
CLARK est un MANAGER
SCOTT est un ANALYST
KING est un PRESIDENT
TURNER est un SALESMAN
ADAMS est un CLERK
JAMES est un CLERK
FORD est un ANALYST
MILLER est un CLERK
```

Écrire un fichier SQLJ

Comment différencier un fichier source Java utilisant SQLJ d'un fichier source n'utilisant pas SQLJ ? Tout simplement par l'extension du fichier source :

- un fichier source Java utilisant SQLJ porte l'extension .sqlj ;
- un fichier source Java n'utilisant pas SQLJ est doté de l'extension .java.

Traduire et compiler un fichier .sqlj : l'utilitaire SQLJ

Les compilateurs Java standard (celui de SUN, celui d'Oracle 10g), ne savent pas compiler directement un fichier source .sqlj. Pour compiler un fichier source .sqlj, un traducteur SQLJ traduit le fichier SQLJ en fichiers source Java et en fichiers de ressources. Ensuite, un compilateur Java standard compile ces fichiers source Java en fichiers .class. Le traducteur se charge de vérifier que les noms des tables, des colonnes et des types de données qui apparaissent dans les requêtes SQLJ sont cohérents avec le schéma relationnel de la base.

Oracle 10g est fourni avec l'utilitaire SQLJ, qui réalise les opérations de génération et la compilation des fichiers .java.

Pour que SQLJ fonctionne, la configuration suivante est nécessaire :

■ CLASSPATH doit contenir %ORACLE_HOME%\sqlj\lib\translator.zip

La syntaxe de SQLJ est simple :

```
sqlj <nom fichier sqlj>
ou pour indiquer sur quel schéma vérifier les requêtes SQL
sqlj -user=nom/mot de passe@jdbc:oracle:thin:@<URL> nom_fichier.sqlj
<URL> = <serveur>:<port Oracle Net >:<SID> (ex: localhost:1521:ORCL)
Par exemple :
sqlj exemplesqlj.sqlj
```

Pour déployer un fichier .sqlj dans Oracle 10g, Loadjava accepte :

- les fichiers générés avec l'utilitaire SQLJ ;
- le fichier .sqlj.

Dans le deuxième cas, l'utilitaire Loadjava appellera le traducteur SQLJ interne à la base de données pour générer les sources Java et les fichiers de ressource et compiler ces fichiers.

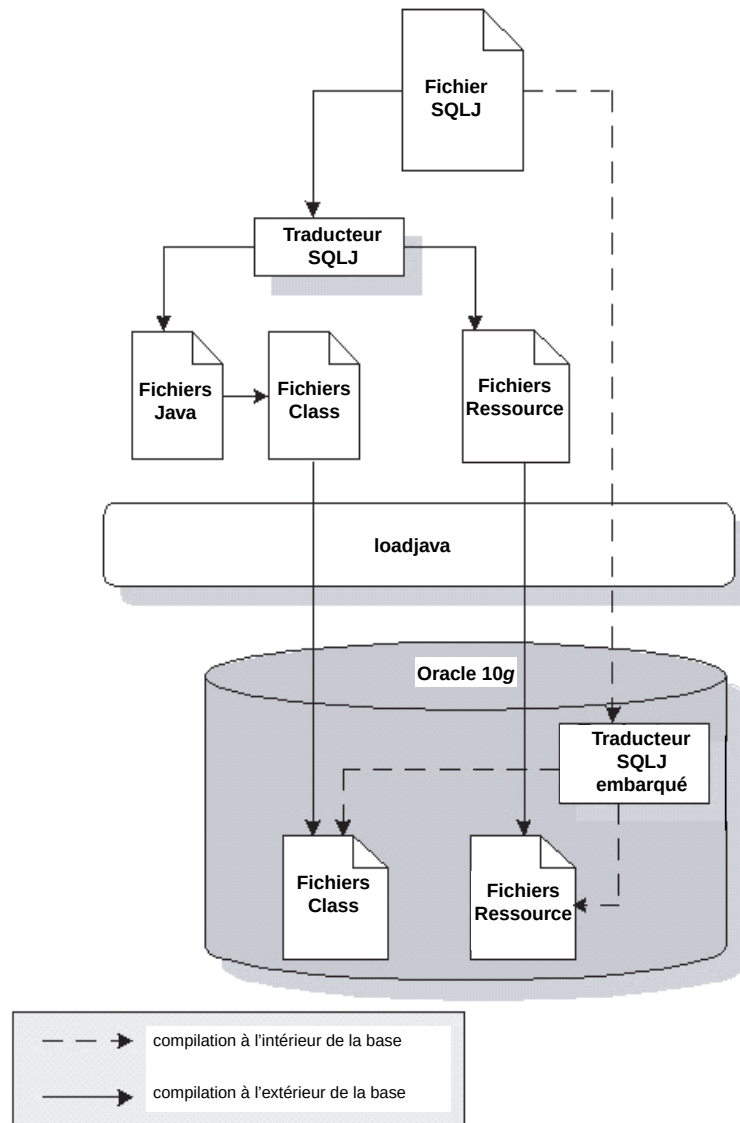
La figure précédente montre les deux mécanismes possibles pour charger dans Oracle 10g des fichiers SQLJ.

SQLJ côté serveur

Aucune modification du code Java n'est nécessaire si le code SQLJ s'exécute dans la base de données : la syntaxe est identique. La seule différence entre une méthode s'exécutant à l'extérieur d'Oracle et une méthode s'exécutant à l'intérieur d'Oracle concerne l'ouverture de session. Lorsque le code SQLJ est traité par la JVM d'Oracle, la session est implicitement la session en cours, comme pour les opérations JDBC. Dans ce cas, il n'est donc pas nécessaire de réaliser une ouverture de session. Côté serveur, SQLJ présente une restriction : les instructions SQLJ ne s'appliquent que sur la base de données où la méthode Java s'exécute ; une base distante n'est pas accessible par des commandes SQLJ.

Figure 19-8

Chargement d'un fichier
SQLJ dans Oracle 10g



Le code du fichier *SalariesKPRB.sqlj* se présente comme suit :

```
import sqlj.runtime.ref.*;
import java.sql.*;
public class SalariesKPRB {
    public SalariesKPRB() throws SQLException {
    }
}
```

```

public static String requete(int empId) throws SQLException {
    String job;
    String nom;

    #sql {SELECT ENAME,JOB into :nom,:job from EMP where empno=:empId};
    System.out.println("Nom du salarié #" + empId + " : " + nom);
    System.out.println("Fonction du salarié #" + empId + " : " + job);
    return new String("#" + empId + " : " + nom + " - " + job);
}

```

La classe SalariesKPRB est prévue pour fonctionner dans Oracle 10g, ce qui explique l'absence d'ouverture de session. Elle comprend la méthode *requete*, qui prend comme argument le numéro d'un employé et renvoie une chaîne de caractères indiquant le nom et la fonction de cet employé. La requête *SELECT* est exécutée sur la table *EMP* du schéma courant. Pour cet exemple, il faudra se connecter sous *SCOTT* afin d'exécuter la méthode Java.

Nous compilons ce fichier avec *SQLJ*.

```

■ sqlj SalariesKPRB.sqlj

```

Trois fichiers sont générés, à savoir :

- SalariesKPRB.class, un fichier de classe Java ;
- SalariesKPRB_SJProfile0.ser, un fichier de ressource Java ;
- SalariesKPRB_SJProfileKeys.class, un fichier de classe Java.

Ensuite, nous déployons ces fichiers dans Oracle 10g pour pouvoir appeler la méthode *requete* comme une procédure stockée Java. Les étapes de mise en œuvre des procédures stockées Java seront présentées au chapitre 20, *Java et Oracle 10g*.

Pour déployer les fichiers dans Oracle 10g, il faut procéder comme suit :

```

■ loadjava -resolve -thin -user scott/tiger@unServeur:1521:ORCL SalariesKPRB.class
  ➔ SalariesKPRB_SJProfile0.ser SalariesKPRB_SJProfileKeys.class

```

La méthode Java *requete* est publiée en tant que fonction PL/SQL sous le nom *Salaries-NomJob* :

```

■ sqlplus scott/tiger
CREATE OR REPLACE FUNCTION SalariesNomJob(p0 NUMBER) RETURN VARCHAR2
AS LANGUAGE JAVA
NAME 'SalariesKPRB.requete(int) return java.lang.String';

```

Pour tester cette méthode, il convient de :

- se connecter à la base en tant que *SCOTT/TIGER* ;
- paramétrer *SQL*PLUS* pour qu'il affiche les sorties envoyées sur *System.out* ;
- appeler la fonction *SalariesNomJob*.

Voici le détail des commandes à lancer dans SQL*PLUS :

```
SQL> set serveroutput on
SQL> call dbms_java.set_output(2000);
SQL> VARIABLE chaine VARCHAR2(80);
SQL> call SalariesNomJob(7839) into :chaine;
Nom du salarié #7839 : KINGFonction du salarié #7839 : PRESIDENT
Appel terminé.

SQL> print chaine;
CHAINE
-----
#7839 : KING - PRESIDENT
```

JDBC 2.0

Depuis la version 8.1.6, les drivers JDBC d'Oracle supportaient partiellement JDBC 2.0. Cette lacune est désormais comblée avec un support total de cette version, et même un support partiel de JDBC 3.0 (cf. la fin de ce chapitre). Ils peuvent être utilisés avec les versions du JDK allant de 1.1 à 1.4.

Les fonctionnalités JDBC 2.0 implémentées sont :

- mise à jour en batch ;
- ResultSet parcourable ;
- JNDI pour spécifier et obtenir une connexion ;
- Connexion Pooling ;
- Connexion Caching ;
- transactions distribuées XA ;
- support des types SQL 3 (CLOB, BLOB, Types Structurés, array).

Mise à jour en batch

La mise à jour en batch permet de grouper plusieurs UPDATE, DELETE ou INSERT dans un unique batch qui est envoyé et traité en une seule fois. Ceci permet de limiter les allers et retours avec la base et donc d'améliorer les performances des applications.

On peut voir deux modèles pour la mise à jour en batch :

- Le modèle d'Oracle permet de gérer implicitement la mise à jour en batch. Il faut juste définir la valeur du batch qui correspond au nombre d'opérations à accumuler avant que le batch soit envoyé à la base.

La mise à jour en batch d'Oracle ne supporte que les OraclePreparedStatement.

```
import java.sql.*;
import oracle.jdbc.*;
```

```

class SetExecuteBatch
{
    public static void main (String args [])
        throws SQLException
    {
        DriverManager.registerDriver(new oracle.jdbc.OracleDriver());
        // Connexion à la base de données
        Connection conn = DriverManager.getConnection ("jdbc:oracle:oci8:@",
            ➤ "scott", "tiger");
        conn.setAutoCommit(false);
        // La valeur du batch est placée à 2 pour tous les PreparedStatement
        ➤ de cette connexion
        ((OracleConnection)conn).setDefaultExecuteBatch (2);
        PreparedStatement ps = conn.prepareStatement ("insert into dept values (?, ?, ?)");
        ps.setInt (1, 12);
        ps.setString (2, "Oracle");
        ps.setString (3, "USA");
        // Aucune donnée n'est envoyée par ce executeUpdate
        System.out.println ("Nombre de rangées mises à jour : "+ ps.executeUpdate ());
        ps.setInt (1, 11);
        ps.setString (2, "Applications");
        ps.setString (3, "Indonesia");
        // Le nombre d'opérations a été atteint
        // donc envoi de données à la base et insertion
        int rows = ps.executeUpdate ();
        System.out.println ("Nombre de rangées mises à jour : " + rows);
        conn.commit();
        ps.close ();
        conn.close();
    }
}

```

- Le modèle standard permet d'effectuer une mise à jour en batch de manière manuelle c'est-à-dire de façon explicite. Dans ce cas, il n'y a pas de valeur de batch, il faut ajouter manuellement les opérations au batch et ensuite choisir explicitement quand exécuter le batch.

```

import java.sql.*;
import oracle.jdbc.*;

class StdExecuteBatch
{
    public static void main (String args [])
        throws SQLException
    {
        DriverManager.registerDriver(new oracle.jdbc.OracleDriver());
        // Connexion à la base de donnée
        Connection conn = DriverManager.getConnection ("jdbc:oracle:oci8:@",
            ➤ "scott", "tiger");
        conn.setAutoCommit(false);
        PreparedStatement ps = conn.prepareStatement ("insert into dept values (?, ?, ?)");
    }
}

```



```
        ps.setInt (1, 22);
        ps.setString (2, "Charon");
        ps.setString (3, "Fr");
        // Ajout au batch de l'opération précédente
        ps.addBatch();
        ps.setInt (1, 21);
        ps.setString (2, "Lebeau");
        ps.setString (3, "Fr");
        ps.addBatch();
        // Exécution du batch
        ps.executeBatch ();
        conn.commit();
        ps.close ();
        conn.close();
    }
}
```

Dans les deux cas, il est nécessaire de faire un *commit* pour valider la mise à jour (sauf si l'*AutoCommit* est laissé activé).

Les *ResultSet* parcourables

Les *ResultSet* existaient déjà dans JDBC 1. Ils permettent d'encapsuler les enregistrements retournés par l'exécution d'une requête et, grâce à un pointeur sur l'enregistrement courant, d'obtenir la valeur des colonnes de l'enregistrement courant. Pour cela, on utilise les méthodes *getXXX()* qui convertissent le type SQL dans un type Java correspondant.

JDBC 2 a permis au *ResultSet* d'être parcourable dans les deux sens (SCROLL contrairement à FORWARD_ONLY), de suivre les changement de la base qui influent sur lui (SENSITIVE contrairement à INSENSITIVE), d'être mis à jour (UPDATABLE contrairement à READ_ONLY), et offre aussi le positionnement absolu et relatif. On obtient alors six catégories de *ResultSet* :

- FORWARD_ONLY/READ_ONLY ;
- FORWARD_ONLY/UPDATABLE ;
- SCROLL_SENSITIVE/READ_ONLY ;
- SCROLL_SENSITIVE/UPDATABLE ;
- SCROLL_INSENSITIVE/READ_ONLY ;
- SCROLL_INSENSITIVE/UPDATABLE.

Exemples montrant les nouvelles fonctionnalités des *ResultSet* :

```
// Chargement du driver JDBC d'Oracle
DriverManager.registerDriver(new oracle.jdbc.OracleDriver());
// Connexion à la base
Connection conn =
```

```

DriverManager.getConnection ("jdbc:oracle:oci8:@", "scott", "tiger");
// Crée un Statement
Statement stmt = conn.createStatement (ResultSet.TYPE_SCROLL_INSENSITIVE,
    ➤ResultSet.CONCUR_UPDATABLE);
// Sélection des noms des employés de la table emp
ResultSet rset = stmt.executeQuery ("select ENAME from EMP");

```

À partir de ce `ResultSet`, on peut afficher le résultat en effectuant un parcours en partant de la fin :

```

// Vérifie que le curseur n'est pas après la dernière ligne
if (!rset.isAfterLast())
{
    // Place le curseur après la dernière ligne
    rset.afterLast ();
}
// Parcourt le ResultSet à l'envers en utilisant previous()
while (rset.previous())
    System.out.println (rset.getString ("ENAME"));

```

Exemple d'affichage du résultat en utilisant la navigation absolue :

```

int idx=1;
while (rset.absolute(idx))
{
    System.out.println (rset.getString ("ENAME"));
    idx ++;
}

```

Prenons un nouvel exemple :

```

// Chargement du driver JDBC d'Oracle
DriverManager.registerDriver(new oracle.jdbc.OracleDriver());
// Connexion à la base
Connection conn =
    DriverManager.getConnection ("jdbc:oracle:oci8:@", "scott", "tiger");
// Crée un Statement
Statement stmt = conn.createStatement (ResultSet.TYPE_SCROLL_INSENSITIVE,
    ➤ResultSet.CONCUR_UPDATABLE);
// Sélection des noms des employés de la table emp
ResultSet rset = stmt.executeQuery ("select ENAME,SAL from EMP");

```

Donnons une augmentation de 500 \$ à tous les employés :

```

int salaire = 0;
while (rset.next ())
{
    salaire = rset.getInt ("SAL");
    // Mise à jour de la ligne
    rset.updateInt ("SAL", salaire + 500);
    // Mise à jour dans la base
    rset.updateRow ();
}

```

Insérons un nouvel employé dans le ResultSet :

```
// Préparation du ResultSet à l'insertion d'une ligne
rset.moveToInsertRow();
rset.updateInt("EMPNO",9876);
rset.updateString("EMPNAME","Doulet");
rset.updateInt("SAL",2000);
// Insertion de la ligne
rset.insertRow();
// Retour à la ligne courante
rset.moveToCurrentRow();
```

Toutes ces nouvelles fonctionnalités apportées par JDBC 2 rendent les ResultSet plus souples et plus complets.

DataSource

Les API de JDBC 2 introduisent le concept de DataSource qui est un objet utilisé pour spécifier une base de données ou d'autres sources de données. Ces sources de données peuvent être liées à Java Naming and Directory Interface (JNDI), une interface permettant d'accéder à une base de données par des noms logiques, tout cela dans le souci d'une meilleure portabilité.

Le service de DataSource fournit un remplacement complet du service Driver Manager.

Caractéristique et propriété

Précédemment, on a vu comment utiliser le Driver Manager. Cependant, le problème avec ce modèle est qu'il faut inclure dans le code le nom de la classe spécifique au vendeur, l'URL de la base et d'autres propriétés (nom de la machine, numéro de port ...).

Avec DataSource utilisant JNDI, il n'est plus nécessaire d'enregistrer la classe spécifique au vendeur...

Interface DataSource et l'implémentation d'Oracle

Une source de donnée est une instance de la classe qui implémente l'interface standard `javax.sql.DataSource`.

Oracle implémente aussi cette interface avec la classe `OracleDataSource` du package `oracle.jdbc.pool`. La méthode `getConnection()` prend les mêmes paramètres que celle fournie par Sun Microsystems mais elle a été surchargée et retourne une instance `OracleConnection`.

La classe `OracleDataSource`, comme toutes les classes implémentant l'interface `DataSource`, fournit un ensemble de propriétés permettant de se connecter à une base de données :

- `databaseName` : nom de la base de données sur le serveur (SID) ;

- `dataSourceName` : nom de la classe de la source de donnée ;
- `description` : description de la source de données ;
- `networkProtocol` : protocole de communication avec le serveur (par défaut `tcp`) ;
- `user` : login de l'utilisateur ;
- `password` : mot de passe de l'utilisateur ;
- `portNumber` : numéro de port où le serveur écoute ;
- `serverName` : nom du serveur de base de données.

La classe `OracleDataSource` implémente les méthodes `get_` et `set_` sur toutes les propriétés sauf `getPassword()` pour des raisons de sécurité évidentes.

Ces propriétés sont les propriétés standards définies par Sun Microsystems. Mais Oracle a aussi défini des propriétés spécifiques :

- `driverType` : définit le type de driver JDBC Oracle (`oci`, `Thin` ou `kprb`) ;
- `tnsEntry` : nom de l'entrée TNS (utilisable seulement avec le driver `oci`) ;
- `url` : chaîne de connexion à la base de données ("`jdbc:oracle:<driveType>:@<tnsEntry>`"). Cette propriété peut être utilisée pour remplacer les propriétés standards (`databaseName`, `serverName...`) et donc on aurait :
`"jdbc:oracle:<driveType>:@<serverName>:<portNumber>:<databaseName>"`
- `nativeXA` : autoriser cette propriété `OracleXADataSource` quand les caractéristiques `HeteroRM XA` sont utilisés avec le driver `oci` (faux par défaut) .

Création d'une instance `DataSource` et connexion sans JNDI

Exemple basic d'utilisation d'un `DataSource` pour se connecter à une base de données sans utiliser les fonctionnalités de JNDI. Ceci impose que le code contienne des propriétés de connexion au serveur de données en dur.

1. Création d'une instance `OracleDataSource` :

```
OracleDataSource ods = new OracleDataSource();
```

2. Initialisation des propriétés de connexion :

```
ods.setDriverType("oci8");  
ods.setServerName("Serveur1");  
ods.setNetworkProtocol("tcp");  
ods.setDatabaseName("db");  
ods.setPortNumber(1521);  
ods.setUser("scott");  
ods.setPassword("tiger");
```

3. Récupération d'une connexion :

```
Connection conn = ods.getConnection();
```

on peut aussi forcer une connexion avec un autre utilisateur :

```
...  
Connection conn = ods.getConnection("bill", "PASS");  
...
```

Création d'une instance DataSource, enregistrement avec JNDI et connexion

Ici, l'utilisation de JNDI permet de n'avoir les propriétés codées en dur que dans une seule partie du code où est liée l'instance de la DataSource au nom logique. Ensuite, le reste du code devient totalement portable car, par la suite, on utilise le nom logique de la source de données pour la connexion.

1. Création d'une instance OracleDataSource :

```
OracleDataSource ods = new OracleDataSource();
```

2. Initialisation des propriétés de connexion :

```
ods.setDriverType("oci8");  
ods.setServerName("serveur1");  
ods.setNetworkProtocol("tcp");  
ods.setDatabaseName("db");  
ods.setPortNumber(1521);  
ods.setUser("scott");  
ods.setPassword("tiger");
```

3. Enregistrement de l'instance OracleDataSource :

```
...  
// Récupération du contexte initial de nommage JNDI  
Context ctx = new InitialContext();  
// Liaison du DataSource avec le nom Logique  
ctx.bind("jdbc/ora9i", ods);  
...
```

L'espace de nommage JNDI a une hiérarchisation similaire au système de fichier. Dans cet exemple, le nom JNDI spécifie le nom logique ora9i qui appartient au sous-contexte jdbc.

La spécification JDBC 2 impose que toutes les sources de données JDBC soit enregistrées dans un sous-contexte jdbc.

4. Ouvrir une connexion depuis le nom logique :

```
...  
OracleDataSource ods = (OracleDataSource)ctx.lookup("jdbc/ora9i");  
Connection conn = ods.getConnection();  
...
```

Le pool de connexions

En général, la connexion à une base de données est une opération assez lourde (surtout la création et la fermeture de session). Il serait donc intéressant de pouvoir s'y connecter plus rapidement. La solution proposée par JDBC 2 est de réutiliser les connexions physiques à la base (qui prennent beaucoup de temps) en les réaffectant pour des nouvelles demandes de connexions de clients. On distingue alors deux types de connexions : les connexions physiques à la base de données et les connexions logiques.

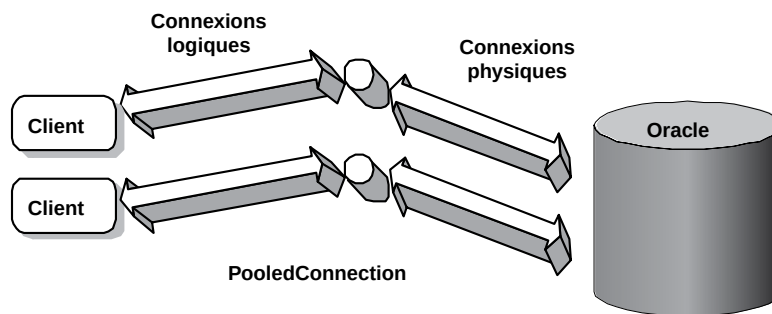
Les principaux concepts sont :

- Connection PoolDataSource qui ont des fonctionnalités et des concepts similaires au DataSource mais les méthodes retournent des instances de Pooled Connection au lieu d'instances de connexion normale.
- Pooled Connection qui représente une connexion physique à la base de données, maintenue ouverte par des connexions logiques.

Avec l'utilisation des pools de connexions, la fermeture d'une connexion logique par le client n'impose pas la fermeture de la connexion physique :

Figure 19-9

Pool de connexion



L'interface ConnectionPoolDataSource et l'implémentation d'Oracle

L'interface `javax.sql.ConnectionPoolDataSource` fournit les fonctionnalités standard des `PooledConnectionDataSource`.

Oracle fournit aussi son implémentation de cette interface avec la classe `oracle.jdbc.pool.OracleConnectionPoolDataSource` qui hérite de la classe `OracleDataSource` et donc inclut toutes les propriétés de connexion ainsi que toutes les méthodes `get_` et `set_` décrites précédemment.

Pour obtenir une instance de `PooledConnection`, on utilise la méthode `getPooledConnection()` qui peut prendre comme paramètres le nom de l'utilisateur et son mot de passe.

L'interface PooledConnection et l'implémentation d'Oracle

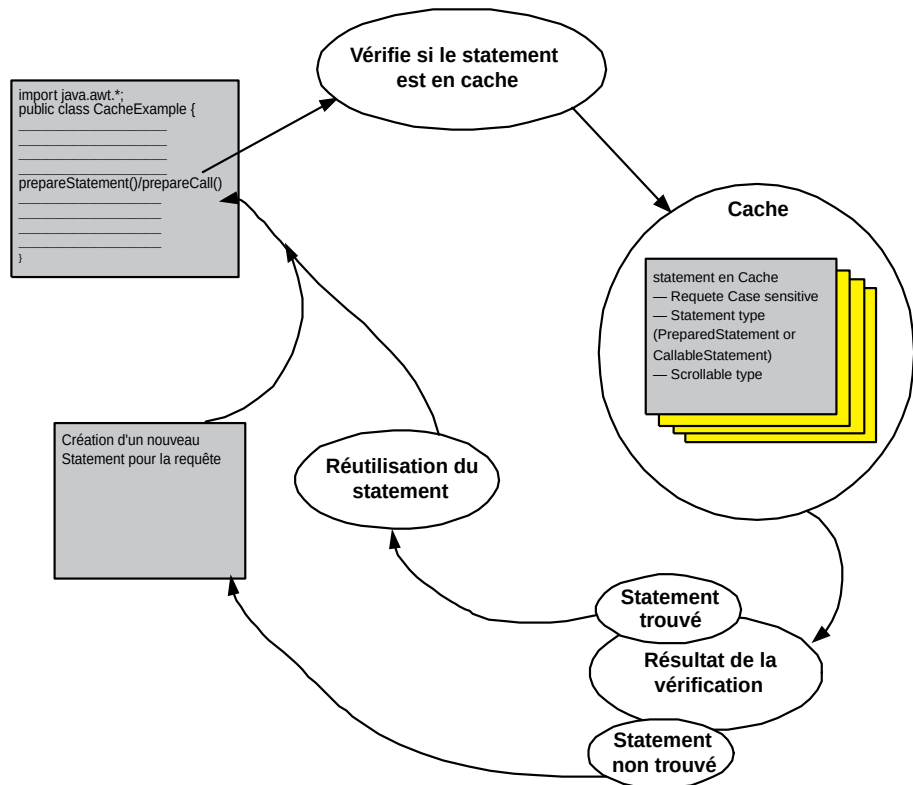
Une PooledConnection est une instance de la classe qui implémente l'interface standard `javax.sql.PooledConnection`. La méthode `getConnection()` spécifiée par cette interface retourne une instance de connexion logique.

Oracle fournit son implémentation de cette interface avec la classe `oracle.jdbc.pool.OraclePooledConnection`. Une PooledConnection sera amenée à produire un certain nombre d'instances de connexions, mais seulement une à la fois. Chaque fois que la méthode `getConnection()` est appelée, l'instance de PooledConnection retourne une nouvelle instance de connexion et ferme toute connexion précédemment ouverte.

La classe `OraclePooledConnection` permet d'utiliser le statement caching pour les PooledConnection. Ce cache de déclaration de requête SQL (*statement*) permet à un statement créé par une connexion logique d'être réutilisé par une autre connexion logique. Le *Statement caching* peut être utilisé de façon implicite (sans dénomination spéciale du statement pour sa réutilisation) ou de façon explicite (nomination du statement).

Figure 19-10

Statement Caching



Création d'une `ConnectionPoolDataSource` et Connexion

Exemple basic d'utilisation d'une `ConnectionPooledDataSource` pour se connecter à une base de données sans utiliser les fonctionnalités de JNDI. Cependant, comme pour les `DataSources`, il est possible d'utiliser JNDI pour lier une instance de `ConnectionPoolDataSource` à un nom logique.

1. Importation du package Oracle pour le Connection Pooling :

```
import oracle.jdbc.pool.*;
```

2. Création d'une instance `OracleConnectionPoolDataSource` :

```
OracleConnectionPoolDataSource ocpds = new OracleConnectionPoolDataSource();
```

3. Initialisation des propriétés de connexions :

```
ocpds.setDriverType("oci8");  
ocpds.setServerName("serveur1");  
ocpds.setNetworkProtocol("tcp");  
ocpds.setDatabaseName("db");  
ocpds.setPortNumber(1521);  
ocpds.setUser("scott");  
ocpds.setPassword("tiger");
```

4. Récupération d'une `PooledConnection` :

```
PooledConnection pc = ocpds.getPooledConnection();
```

5. Récupération d'une connexion :

```
Connection conn = pc.getConnection();
```

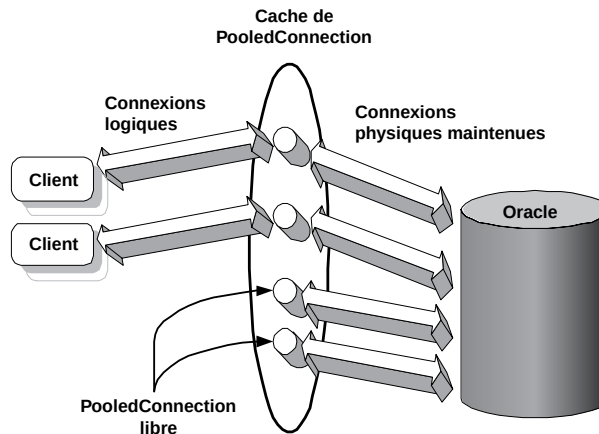
Connection Caching

Le Connection Caching permet de garder et d'utiliser un cache de connexions physiques à une base de données. Il utilise les `PooledConnection` et est généralement implanté sur un serveur intermédiaire.

Les spécifications JDBC 2 ne comportent pas d'implémentation de Connection Caching mais Oracle en fournit une en tant qu'exemple avec les fonctionnalités minimum :

- l'interface `oracle.jdbc.pool.OracleConnectionCache` qui hérite des méthodes de l'interface `DataSource` mais spécifie aussi les méthodes `reusePooledConnection()` pour affecter une instance de `PooledConnection` au cache de `PooledConnection`, `closePooledConnection()` pour fermer une instance de `PooledConnection` du cache et `close()` pour fermer l'instance de `OracleConnectionCache` ;
- la classe `oracle.jdbc.pool.OracleConnectionCacheImpl` qui implémente cette interface.

Figure 19-11
Cache de connexions



Création d'un cache de PooledConnection

Exemple de création d'un cache de PooledConnection en utilisant l'interface d'Oracle.

1. Création de l'instance de OracleConnectionCacheImpl :

```
OracleConnectionCacheImpl ocacheimpl = new OracleConnectionCacheImpl();
```

2. Initialisation des propriétés des instances de PooledConnection du cache :

```
ocacheimpl.setDriverType("oci8");
ocacheimpl.setServerName("serveur1");
ocacheimpl.setNetworkProtocol("tcp");

ocacheimpl.setDatabaseName("db");
ocacheimpl.setPortNumber(1521);
ocacheimpl.setUser("scott");
ocacheimpl.setPassword("tiger");
```

3. Paramétrage du nombre d'instances de PooledConnection :

```
// Définition du maximum d'instance de PooledConnection du cache
ocacheimpl.setMaxLimit(10);
// Définition du minimum d'instance de PooledConnection du cache
ocacheimpl.setMinLimit(3);
```

4. Définition du mode d'utilisation des PooledConnection :

Il existe trois modes d'utilisation : un mode dynamique (DYNAMIC_SCHEME) qui alloue de nouvelles connexions au-delà de la limite définie et les libère dès qu'elles sont inutilisées, un mode fixe sans attente (FIXED_RETURN_NULL_SCHEME) qui ne permet pas de dépassement du nombre maximum de PooledConnection et au-delà de la limite, chaque nouvelle requête de connexion retourne *null*. Le dernier mode est un mode fixe avec attente (FIXED_WAIT_SCHEME) qui ne permet pas de dépassement de la limite du nombre d'instances de connexion et, au-delà de la limite, une nouvelle requête attend qu'un autre client libère une connexion.

Le mode par défaut est le mode dynamique.

```
ocacheimpl.setCacheScheme(OracleConnectionCacheImpl.FIXED_RETURN_NULL_SCHEME);
```

5. Récupération des caractéristiques du cache :

Il est possible de récupérer le nombre d'instances de `PooledConnection` actives dans le cache avec `getActiveSize()`. Il est aussi possible d'obtenir le nombre total de `PooledConnection` du cache (les connexions active et inactive).

6. Récupération d'une connexion :

```
Connection conn = ocacheimpl.getConnection();
```

Il est aussi possible d'utiliser JNDI pour référencer le cache de connexion et pour récupérer la connexion à partir du nom logique.

Les transactions distribuées

Une transaction distribuée est un ensemble de deux transactions ou plus qui doivent être coordonnées. Prenons l'exemple d'un transfert d'argent entre deux banques. Dans ce cas, il ne faudrait pas que la transaction soit validée sans avoir l'assurance que tout s'est bien passé pour chaque banque.

Dans les API de JDBC 2, les interfaces de transaction distribuée `javax.sql.XADataSource`, `javax.sql.XAConnection` sont très proches des interfaces de `ConnectionPoolDataSource` et `PooledConnection`. Ces transactions distribuées suivent le standard XA (eXtended Architecture).

Composants d'une transaction distribuée

Un système de transaction distribuée est généralement basé sur un gestionnaire de transaction externe (Transaction Manager) pour coordonner chaque transaction. Le gestionnaire de transaction peut être un composant logiciel qui implémente JTA (Java Transaction API).

Pour chaque base de données utilisée dans la transaction distribuée est créée une instance de `XADataSource`. Une `XAConnection` est obtenue en demandant une connexion à cette `XADataSource`.

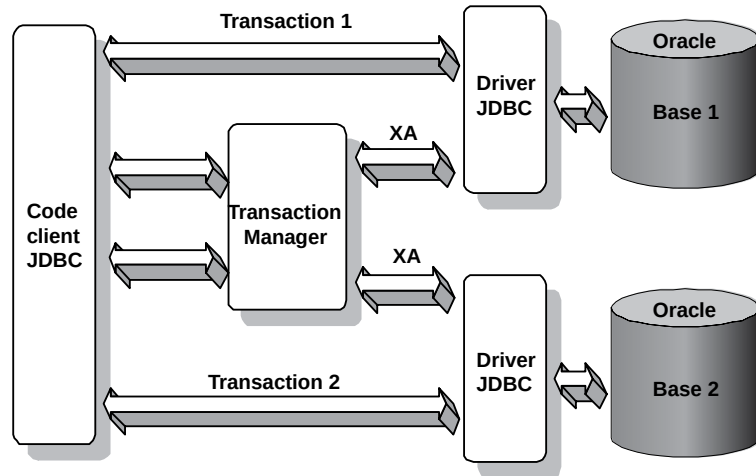
Une `XAConnection` produit une `XAResource` et une instance de connexion JDBC. La `XAResource` est utilisée par le Transaction Manager pour coordonner la transaction de cette branche de la transaction distribuée.

Une instance de `XAResource` a la possibilité de démarrer, finir, préparer, valider (commit) ou annuler (rollback) les opérations de transaction sur la branche associée avec l'instance de `XAResource`. Mais, comme l'issue de la transaction est déterminée par le gestionnaire de transaction, l'application ne doit jamais utiliser les méthodes `Connection.commit()` et `Connection.rollback()`.

Chaque instance de XAResource est associée à un identifiant de transaction (Transaction ID) qui inclut le même identifiant de transaction distribuée.

Figure 19-12

Exemple de transaction distribuée



Implémentation d'une transaction distribuée

Le code suivant est un exemple de mise en œuvre d'une validation à deux phases utilisant XA pour une transaction distribuée.

1. Création de la première instance de XADataSource :

```

OracleXADataSource oxds1 = new OracleXADataSource();
oxds1.setURL("jdbc:oracle:oci8:@");
oxds1.setUser("scott");
oxds1.setPassword("tiger");

```

2. Création de la deuxième instance de XADataSource :

```

OracleXADataSource oxds2 = new OracleXADataSource();
oxds2.setURL("jdbc:oracle:thin:@(description=(address=(host=serveur1)(protocol=tcp)
➔(port=1521))(connect_data=(sid=db)))");
oxds2.setUser("scott");
oxds2.setPassword("tiger");

```

3. Récupération des XAConnection :

```

XAConnection pc1 = oxds1.getXAConnection();
XAConnection pc2 = oxds2.getXAConnection();

```

4. Récupération des connexions physiques :

```

Connection conn1 = pc1.getConnection();
Connection conn2 = pc2.getConnection();

```

5. Récupération des XAResource :

```
XAResource oxar1 = pc1.getXAResource();  
XAResource oxar2 = pc2.getXAResource();
```

6. Création des identifiants avec le même identifiant global :

```
Xid xid1 = createXid(1);  
Xid xid2 = createXid(2);
```

7. Association de la nouvelle branche de transaction avec la transaction distribuée :

```
oxar1.start (xid1, XAResource.TMNOFLAGS);  
oxar2.start (xid2, XAResource.TMNOFLAGS);
```

8. Quelques opérations effectuées sur les ressources :

```
doSomeWork1 (conn1);  
doSomeWork2 (conn2);
```

9. Dissociation de la branche de sa transaction distribuée avec succès :

```
oxar1.end(xid1, XAResource.TMSUCCESS);  
oxar2.end(xid2, XAResource.TMSUCCESS);
```

On pourrait aussi avoir d'autres états : TMFAILED ou TMSUSPEND.

10. Préparation des ressources (teste si les bases de données sont accessibles...) :

```
int prp1 = oxar1.prepare (xid1);  
int prp2 = oxar2.prepare (xid2);
```

11. Test quant au commit à deux phases sur les deux ressources.

C'est ici que se décide si la transaction est validée ou non. Dans un premier temps, on vérifie que les deux XAResources sont prêtes et qu'elles ne sont pas en lecture seule. Si toutes ces conditions sont remplies (do_commit = true), et seulement dans ce cas, les deux transactions sont validées (commit) sinon elles sont annulées toutes les deux (rollback).

```
boolean do_commit = true;  
if (!(prp1 == XAResource.XA_OK || (prp1 == XAResource.XA_RDONLY)))  
    do_commit = false;  
if (!(prp2 == XAResource.XA_OK || (prp2 == XAResource.XA_RDONLY)))  
    do_commit = false;  
if (prp1 == XAResource.XA_OK)  
    if (do_commit)  
        oxar1.commit (xid1, false);  
    else  
        oxar1.rollback (xid1);  
if (prp2 == XAResource.XA_OK)  
    if (do_commit)  
        oxar2.commit (xid2, false);  
    else  
        oxar2.rollback (xid2);
```

12. Fermeture des connexions ouvertes :

```
conn1.close();
conn1 = null;
conn2.close();
conn2 = null;
pc1.close();
pc1 = null;
pc2.close();
pc2 = null;
```

On utilise la fonction `CreateXid` pour générer l'identifiant de transaction :

```
static Xid createXid(int bids)
    throws XAException
{
    byte[] gid = new byte[1]; gid[0]= (byte) 9;
    byte[] bid = new byte[1]; bid[0]= (byte) bids;
    byte[] gtrid = new byte[64];
    byte[] bqual = new byte[64];
    System.arraycopy (gid, 0, gtrid, 0, 1);
    System.arraycopy (bid, 0, bqual, 0, 1);
    Xid xid = new OracleXid(0x1234, gtrid, bqual);
    return xid;
}
```

JDBC 3.0

Oracle 10g supporte JDBC 3.0 avec les fonctionnalités suivantes :

- Transaction Savepoints ;
- Passage entre transactions locales et globales ;
- Statement caching ;
- Support du JDK 1.4.

Transaction Savepoints

Il est désormais possible de gérer des Savepoints pendant les transactions, ce qui permet d'en faire une gestion plus fine. Les applications peuvent placer des Savepoints durant les transactions, puis effectuer un Rollback (mais pas de commit !) à partir de ce point. Cette fonctionnalité n'est supportée que dans une transaction locale. Spécifier un Savepoint dans une transaction globale renvoie une `SQLException`.

Pour vérifier si les Savepoints sont supportés, la méthode `oracle.jdbc.OracleDatabase-MetaData.supportsSavepoints()` renvoie `true` si le support est effectif.

Création d'un Savepoint

La création d'un Savepoint se fait à l'aide des commandes suivantes qui renvoient des instances de Savepoint : `Connection.setSavepoint();` ou `OracleConnection.oracleSetSavepoint();`.

Il est possible de leur donner un nom : `Connection.setSavepoint(String nom);` ou `OracleConnection.oracleSetSavepoint(String nom);`.

Effectuer un Rollback vers un Savepoint

Le Rollback se fait avec les commandes suivantes : `Connection.rollback(Savepoint svpt);` ou `OracleConnection.oracleRollback(OracleSavepoint svpt);`.

Suppression d'un Savepoint

La suppression d'un Savepoint se fait avec les commandes suivantes : `Connection.releaseSavepoint(Savepoint svpt)` ou `OracleConnection.oracleReleaseSavepoint(OracleSavepoint svpt)`.

Il est à noter qu'après un ordre Commit ou Rollback, tous les Savepoints sont automatiquement supprimés.

Passage entre transactions locales et globales

Avec JDBC 3.0, les applications peuvent changer de connexion entre les transactions locales et globales. Une connexion est toujours dans l'un des trois modes suivant :

- **NO_TXN** : la connexion n'utilise pas de transaction ;
- **LOCAL_TXN** : la connexion utilise une transaction en auto-commit désactivé ;
- **GLOBAL_TXN** : la connexion utilise une transaction globale.

Chaque connexion est instanciée en **NO_TXN** et passe d'un mode à l'autre en fonction des opérations qui sont effectuées :

- une transaction passe de **NO_TXN** à **LOCAL_TXN** lorsque l'auto-commit est désactivé et un ordre DML (Select, Insert, Update) est exécuté ;
- une transaction passe de **NO_TXN** à **GLOBAL_TXN** lorsque la méthode `start()` est invoquée sur une ressource **XA** ;
- une transaction passe de **LOCAL_TXN** à **NO_TXN** lorsqu'un ordre DDL est exécuté (Create, Drop, Alter, Rename) ou lorsqu'un ordre commit/rollback est exécuté ;
- une transaction ne passe **jamais** de **LOCAL_TXN** à **GLOBAL_TXN** ;
- une transaction passe de **GLOBAL_TXN** à **NO_TXN** lorsque la méthode `end()` est invoquée sur une ressource **XA** ;
- une transaction ne passe **jamais** de **GLOBAL_TXN** à **LOCAL_TXN**.

Statement caching

La mise en cache de Statements JDBC permet d'augmenter sensiblement les performances lorsqu'un même Statement est utilisé plusieurs fois, dans une boucle par exemple. Cette fonctionnalité permet d'économiser les ressources nécessaires à la création de plusieurs curseurs et de plusieurs Statements.

Le cache est associé à une connexion physique, donc à un objet `OracleConnection` ou `OraclePooledConnection`. Dans le premier cas, chaque connexion possède son propre cache, dans le second toutes les connexions du pool utilisent le même cache.

Il y a deux types de mise en cache : implicite et explicite, qui peuvent fonctionner en même temps, indépendamment l'une de l'autre ou pas du tout.

Statement caching implicite

Avec la mise en cache implicite, JDBC met automatiquement en cache tous les `PreparedStatement` et `CallableStatements` lors de l'appel de leur méthode `close()`. Ces Statements sont identifiés dans le cache par une clé qui correspond à leur requête SQL, c'est pourquoi les Statements « classiques » ne sont pas mis en cache : ils sont construits sans requête SQL.

La recherche dans le cache JDBC s'effectue lors de l'appel de `PreparedStatement()` ou `PrepareCall()` et se base sur les critères suivant :

- la requête SQL doit être identique ;
- le type de Statement doit être identique (`Callable` ou `Prepared`) ;
- le type de `ResultSet` de retour doit être le même (classique ou parcourable).

Si aucun Statement correspondant aux critères n'est trouvé dans le cache, un nouveau est créé et exécuté automatiquement.

Statement caching explicite

La mise en cache explicite permet de mettre en cache tous les types de Statements. Alors qu'avec la mise en cache implicite un simple appel à `PreparedStatement()` ou à `CallableStatement` déclenchait une recherche dans le cache, avec l'option explicite, il va falloir utiliser des méthodes « `WithKey` » pour mettre en cache et récupérer des Statements. Ces méthodes fonctionnent avec une chaîne de caractères Java qui va servir d'identifiant à chaque Statement mis en cache. Si ces méthodes ne trouvent pas de Statement correspondant aux critères requis dans le cache, elles renvoient « null », contrairement à la mise en cache implicite qui créait automatiquement un nouveau Statement.

La mise en cache s'effectue à l'aide de la méthode `CloseWithKey()`, et la recherche s'effectue avec `getStatementWithKey()` et `getCallWithKey()`.

Activation/Désactivation de la mise en cache implicite

L'activation de la mise en cache implicite se fait par l'appel de `setImplicitStatementCaching(true)` sur la connexion.

La désactivation de la mise en cache explicite se fait par l'appel de `setImplicitStatementCaching(false)` sur la connexion.

Activation/Désactivation de la mise en cache explicite

Pour mettre en place la mise en cache explicite, il faut tout d'abord configurer la taille du cache en invoquant les méthodes `OracleConnection.setStatementCacheSize()` sur la connexion ou `OracleDataSource.setMaxStatements()`. Dans les deux cas, l'argument à envoyer est le nombre de Statements qui pourront être mis en cache : 0 signifie pas de cache.

Exemple de configuration du cache pour 10 Statements :

```
■ ((OracleConnection)conn).setStatementCacheSize(10);
```

Pour désactiver la mise en cache explicite, il faut utiliser la méthode `setExplicitStatementCaching(false)`.

Support du JDK 1.4

Le JDK 1.4 implémente des classes qui étaient auparavant développées par Oracle pour combler un manque fonctionnel. Voici la liste des classes mises à jour :

- `oracle.jdbc.OracleSavepoint` devient `java.sql.Savepoint` ;
- `oracle.jdbc.OracleConnexion` devient `java.sql.connexion` ;
- `oracle.jdbc.OracleParameterMetaData` devient `java.sql.ParameterMetaData`.

Résumé

JDBC est le standard indispensable pour réaliser la communication entre un programme Java (applet ou application) et une base de données. Avec sa version 2, cette API permet de mettre en œuvre des pools de connexions, des transactions distribuées qui la rendent facilement utilisable dans des architectures multiniveaux. Contrairement à ODBC, l'API JDBC est portable sur toutes les plates-formes prenant en charge Java.

Pour sa part, le standard SQLJ fournit une interface moins complexe pour accéder à une base de données.

20

Java et Oracle 10g

Dans ce chapitre :

- l'environnement Java d'Oracle 10g ;
- les objets Java stockés dans Oracle 10g ;
- les procédures Java stockées.

Ce chapitre aborde les fonctionnalités Java intégrées dans le serveur de base de données Oracle 10g. Il ne s'agit en aucun cas d'un guide d'apprentissage du langage Java dont la connaissance est cependant nécessaire pour une bonne compréhension.

Java a d'abord été conçu pour développer des interfaces graphiques de petite taille facilement téléchargeables sur Internet. Il offre en effet des possibilités intéressantes pour améliorer l'ergonomie et les fonctionnalités des sites Internet, notamment l'interactivité avec les utilisateurs. Néanmoins, Java n'est pas uniquement destiné au développement d'applications client : il a rendu possible la conception d'applications distribuées complexes et multi-plates-formes. Grâce à sa portabilité et à sa nature orientée objet, Java est maintenant à la base de très nombreuses applications métier déployées sur des serveurs.

Oracle 10g fournit une plate-forme de mise en œuvre destinée aux applications réseau (intranet ou Internet), qui intègre une machine virtuelle Java au sein du serveur de base de données. Elle constitue un complément du moteur PL/SQL. Avec Java et Oracle 10g, il devient possible de réaliser des procédures stockées écrites en Java.

Après avoir décrit comment fonctionne la machine virtuelle Java Oracle et la configuration de l'environnement Java, nous décrirons le déploiement d'objets Java et de procédures stockées Java dans Oracle 10g. À la fin de ce chapitre, la machine virtuelle Java d'Oracle 10g et ses nombreuses fonctionnalités n'auront plus de secret pour vous.

L'environnement Java d'Oracle 10g

Oracle a développé sa propre machine virtuelle Java, compatible avec les spécifications de SUN Microsystems. Dans la version Oracle 10g, l'environnement Java est conforme aux spécifications Java 2. Par rapport aux machines virtuelles destinées aux postes client, la JVM d'Oracle 10g est capable de hautes performances dans le domaine transactionnel ; elle peut s'adapter automatiquement à un nombre grandissant d'utilisateurs et communiquer facilement avec la base de données Oracle, notamment avec le moteur PL/SQL d'Oracle 10g.

L'environnement Java d'Oracle 10g se caractérise par :

- une machine virtuelle Java performante ;
- une intégration étroite à la base de données.

Les nombreuses applications développées en PL/SQL n'ont pas besoin d'être réécrites en Java. Grâce à l'intégration de la JVM dans la base de données, une véritable interopérabilité est possible entre le moteur PL/SQL, le moteur SQL et la JVM d'Oracle 10g. La base de données peut exécuter des déclencheurs (triggers), des fonctions, des procédures et des méthodes de type objet écrits en Java, tout comme elle peut exécuter du code PL/SQL.

Évolutions d'Oracle 10g

La machine virtuelle Java interne d'Oracle 10g est en version 1.4.

Les technologies Java J2EE ne sont plus supportées dans la version Oracle 10g.

Ces technologies regroupent :

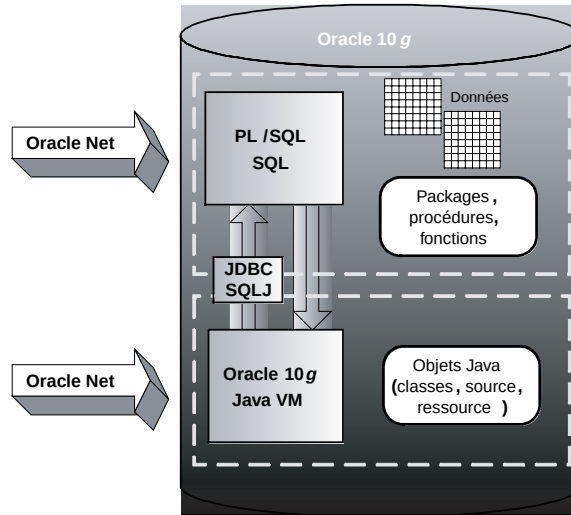
- le conteneur pour Enterprise Java Beans (EJB) ;
- le moteur Java Server Pages (JSP) ;
- le moteur Oracle Servlet Engine (OSE) ;
- l'environnement CORBA intégré dans la base Oracle.

Dans la gamme des produits Oracle, les standards J2EE sont désormais disponibles avec Oracle 10g Application Server (10g AS), le serveur d'application Oracle. Le module OC4J de 10g AS constitue le moteur de déploiement des applications J2EE.

Nous venons de présenter l'environnement Java intégré dans Oracle 10g ; apprenons maintenant à l'exploiter.

Figure 20-1

Java dans Oracle 10g



Configuration Java dans Oracle 10g

Le fonctionnement de la JVM d'Oracle dans une instance Oracle 10g est déterminé par les paramètres de configuration suivants :

- SHARED_POOL_SIZE;
- JAVA_POOL_SIZE ;
- JAVA_SOFT_SESSIONSPACE_LIMIT ;
- JAVA_MAX_SESSIONSPACE_SIZE.

Les paramètres SHARED_POOL_SIZE et JAVA_POOL_SIZE agissent directement sur les performances de la JVM d'Oracle 10g.

Les deux autres paramètres fixent des seuils de consommation mémoire Java : le premier est un seuil d'avertissement, le second un seuil d'erreur.

Ces quatre paramètres sont spécifiés dans le fichier d'initialisation de l'instance (local ou serveur).

Paramètre SHARED_POOL_SIZE

La zone mémoire dont la taille est spécifiée par le paramètre SHARED_POOL_SIZE sert, dans le cadre de la JVM, pour le chargement des classes Java par l'utilitaire *loadjava* par exemple. En effet le chargement d'objets Java dans la base de données

nécessite des traitements lourds en terme de ressource mémoire. Le mécanisme de chargement et de résolution des dépendances entre classes Java est décrit plus loin dans ce chapitre.

La valeur minimum conseillée pour ce paramètre est environ 50 Mo.

Il faut noter que la zone mémoire du pool partagé n'est pas dédiée uniquement à la JVM d'Oracle 10g.

Paramètre JAVA_POOL_SIZE

Le paramètre JAVA_POOL_SIZE spécifie la taille de la zone mémoire utilisée pour le stockage des objets Java pendant leur utilisation.

La taille minimum conseillée est 20 Mo.

Depuis la version Oracle 10g (release 1 et 2), le paramètre JAVA_POOL_SIZE est ajusté automatiquement par les processus de la base.

Paramètre JAVA_SOFT_SESSIONSPACE_LIMIT

Ce paramètre indique le seuil de mémoire Java consommée par une session Oracle au-delà duquel un message d'avertissement est enregistré dans un fichier de trace Oracle.

Paramètre JAVA_SOFT_SESSIONSPACE_SIZE

Ce paramètre indique la quantité de mémoire Java maximale que peut consommer une session Oracle. Si la mémoire Java utilisée par la session dépasse cette valeur, Oracle renvoie au programme appelant une erreur de dépassement de mémoire (« out of memory »).

Par défaut, la valeur de ce paramètre est fixée à 4 Go. S'il existe un risque que la consommation mémoire des classes Java n'est pas maîtrisée, il est conseillé de fixer une valeur moins élevée à ce paramètre.

Sécurité Java dans Oracle 10g

Oracle 10g a des fonctionnalités de gestion de la sécurité pour tous les objets de la base de données. Elle gère la sécurité sur les objets des schémas par type d'accès (lecture, exécution, modification, suppression, etc.). La sécurité de l'environnement Java 2 repose initialement sur l'octroi de permissions sur les ressources système.

Dans ce livre, nous n'expliquerons pas en détail les permissions Java 2.

Cette section décrit l'implémentation et la gestion de la sécurité Java 2 dans Oracle 10g.

Gérer la sécurité Java 2 dans Oracle 10g

Dans les JVM externes, les règles de sécurité Java 2 sont décrites dans un fichier (Security policy file).

Dans Oracle 10g, les règles de sécurité Java 2 sont enregistrées dans une table (Java policy table).

La consultation de la table des permissions Java 2 se fait à travers des vues du dictionnaire Oracle 10g. Ces vues décrivent la configuration des permissions Java 2 paramétrées dans la base de données :

- DBA_JAVA_POLICY ;
- USER_JAVA_POLICY.

Dans ces vues on trouve les informations suivantes :

Colonne	Information
Kind	GRANT ou RESTRICT (octroi ou restriction)
Grantee	schéma ou rôle auquel la permission est accordée
Type_schema	schéma propriétaire de la classe de la permission
Type_name	nom de la classe de la permission
Name	nom de la permission
Action	actions associées à la permission (read, write par exemple)
Enabled	permission active ou inactive
Seq	clé identifiant la permission

Un enregistrement dans la table des permissions Java 2 ne signifie pas forcément un octroi de privilèges.
Le type de la permission est GRANT ou RESTRICT.
Un enregistrement de la table peut donc indiquer une interdiction.

Comme pour les privilèges standard Oracle (système et objets), les permissions Java 2 peuvent être accordées, soit à des schémas, soit à des rôles.

Sécurité Oracle 10g préconfigurée

Dans Oracle 10g, l'octroi de permissions Java 2 est facilité par l'existence de rôles Oracle standard déjà configurés à la création de la base de données :

- JAVA_ADMIN ;
- JAVA_DEPLOY ;
- JAVASYSPRIV ;
- JAVAUSERPRIV ;

- JAVADEBUGPRIV.

Le rôle JAVA_ADMIN accorde des droits privilégiés sur la gestion de la sécurité Java (PolicyTablePermission). Par défaut, il est accordé au rôle DBA.

Le rôle JAVA_DEPLOY accorde des droits pour le déploiement des bibliothèques de classes Java compilées nativement.

Le rôle JAVASYSPRIV accorde des droits privilégiés comme :

- l'administration des classes de la JVM d'Oracle 10g ;
- l'utilisation de sockets (client et serveur) ;
- l'accès complet aux fichiers du système d'exploitation.

Le schéma PUBLIC reçoit par défaut les droits de chargement et d'exécution des classes. Il ne peut en revanche pas modifier les classes des packages Java système telles que oracle.aurora, java et jdbc.

Le rôle JAVAUSERPRIV accorde des droits d'exécution supplémentaires (par rapport à PUBLIC), comme :

- l'utilisation de sockets (client) ;
- l'accès aux fichiers en lecture seule ;
- la gestion des threads.

Le rôle JAVADEBUGPRIV accorde des droits pour utiliser les fonctions de débogage de la JVM d'Oracle 10g.

Gérer les permissions Java 2 dans Oracle 10g

La gestion des permissions Java 2 est effectuée via une API PL/SQL ou Java.

Le package PL/SQL s'appelle DBMS_JAVA.

Il comprend les méthodes pour accorder ou supprimer des privilèges Java 2 dans la JVM Oracle 10g.

Types de permissions

Aux types de permissions Java 2 communs, Oracle 10g ajoute deux types propres à la JVM interne de la base de données :

- oracle.aurora.rdbms.security.PolicyTablePermission;
- oracle.aurora.security.JserverPermission.

Le premier permet d'accorder les droits sur la table des permissions Java 2.

Le second permet d'accorder des droits sur l'utilisation de la JVM interne d'Oracle 10g.

Types de permission dans Oracle 10g

java.util.PropertyPermission
java.io.SerializablePermission
java.io.FilePermission
java.net.NetPermission
java.net.SocketPermission
java.lang.RuntimePermission
java.lang.reflect.ReflectPermission
java.security.SecurityPermission
oracle.aurora.rdbms.security.PolicyTablePermission
oracle.aurora.security.JServerPermission

Accorder des permissions

Pour accorder des permissions, la méthode PL/SQL de l'API est `grant_permission`.

Les informations à fournir à cette méthode sont :

Paramètre	Information
Grantee	schéma ou rôle auquel la permission est accordée
Permission_type	classe correspondant à la permission
Permission_name	attribut de la permission (dépend de la permission)
Permission_action	actions associées à cette permission
Key	(optionnel) identifiant de la permission dans la table

À partir d'un programme Java, il faut appeler la méthode `Grant` de la classe `oracle.aurora.rdbms.security.PolicyTableManager`.

Restreindre des permissions

Pour restreindre des permissions, la méthode PL/SQL de l'API est `restrict_permission`.

Les informations à fournir à cette méthode sont :

Paramètre	Information
Grantee	schéma ou rôle auquel la permission est accordée
Permission_type	classe correspondant à la permission
Permission_name	attribut de la permission (dépend de la permission)
Permission_action	actions associées à cette permission
Key	(optionnel) identifiant de la permission dans la table

À partir d'un programme Java, il faut appeler la méthode `Restrict` de la classe `oracle.aurora.rdbms.security.PolicyTableManager`.

Accorder des permissions du type `PolicyTablePermission`

Précédemment, nous avons décrit `JAVA_ADMIN` comme un rôle ayant les privilèges nécessaires à la modification de la table des permissions Java d'Oracle 10g. Un utilisateur ayant reçu ce rôle peut modifier n'importe quelle permission.

Oracle 10g permet d'accorder des droits de modifications sur la table des permissions Java à d'autres utilisateurs que ceux possédant le rôle `JAVA_ADMIN`. Chaque utilisateur peut recevoir le droit de modifier un ou plusieurs types de permissions dans la table des permissions.

Par exemple, on pourra autoriser le schéma `SCOTT` à mettre à jour uniquement la sécurité sur les accès aux fichiers (`java.io.FilePermission`).

La méthode PL/SQL s'appelle `grant_policy_permission`.

Les informations à fournir à cette méthode sont :

Paramètre	Information
Grantee	schéma ou rôle auquel la permission est accordée
Permission_schema	schéma propriétaire de la classe de la permission
Permission_type	classe du type de permission sur laquelle on accorde la permission
Permission_name	attribut de la permission (dépend de la permission)
Key	(optionnel) identifiant de la permission dans la table

À partir d'un programme Java, il faut appeler la méthode `grantPolicyPermission` de la classe `oracle.aurora.rdbms.security.PolicyTableManager`.

Les objets Java dans Oracle 10g

La version d'Oracle 10g est fournie avec une machine virtuelle Java pour utiliser et exécuter des classes Java depuis le moteur de la base de données. À l'image des autres types d'objets pouvant être créés et stockés dans la base de données (tables, index, séquences, procédures, packages, etc.), Oracle 10g permet d'y stocker les objets Java. La JVM interne d'Oracle 10g n'exécute pas des classes rangées parmi les fichiers du système d'exploitation : elle accède aux classes chargées et stockées dans des schémas de la base de données.

Objets Java stockés dans la base de données

Les différents types d'objets Java sont :

- des fichiers source Java (.java, .sqlj) ;
- des fichiers de classe Java (.class) ;

- des fichiers de ressource Java (.ser, .properties, etc.) ;
- des archives compressées de fichiers Java (.jar, .zip).

Dans la base de données, ces objets sont rangés dans trois catégories de fichiers :

- les sources Java (fichiers source Java) ;
- les classes Java (fichiers de classe Java) ;
- les ressources Java.

Un schéma est propriétaire de chaque objet Java. Ainsi, la même classe peut être chargée plusieurs fois dans des schémas séparés mais dans un schéma ne peut exister qu'un exemplaire de l'objet Java.

De plus, il est possible de créer des synonymes privés ou publics sur les objets Java stockés dans la base.

Pendant le chargement des fichiers d'archives, comprimés, il est possible de les conserver en tant que fichiers d'archives ou de charger séparément les fichiers qu'ils contiennent. Dans le deuxième cas, les fichiers contenus dans les archives sont extraits et triés en source, classe et ressource selon leur extension.

Lancer la suppression d'une archive compressée dans la base de données revient à supprimer les objets Java de la base de données correspondant aux fichiers présents dans cette archive. La suppression concerne uniquement les objets inclus dans le schéma sous lequel la suppression est déclenchée.

Déployer des programmes Java dans la base Oracle

Une différence majeure, par rapport aux programmes Java qui s'exécutent en dehors de la base de données, réside dans la manière de les déployer et de les lancer.

Pour déployer des classes Java dans un système d'exploitation, il faut compiler puis copier des fichiers d'archives, compressées, ou des répertoires référencés dans la variable d'environnement CLASSPATH. Pour lancer une application Java, sa classe principale doit comporter une méthode statique Main.

Dans la base de données Oracle 10g, la notion de programme Java n'existe pas. L'accès aux classes Java se résume à l'appel des méthodes statiques et publiques des classes. Il n'y a pas d'équivalent au programme java (ou java.exe) des machines virtuelles Java externes.

Développer, déployer et exécuter une procédure stockée Java comprend quatre étapes, à savoir :

- écrire le code Java ;
- charger le code Java dans la base de données ;
- publier les procédures Java dans le dictionnaires de données Oracle ;
- utiliser les procédures stockées Java.

Gestion des références entre objets Java

Autre différence significative entre une machine virtuelle Java externe et la JVM interne de la base : la localisation des packages, des classes et des fichiers de ressource.

Dans les JVM externes, la portée de la recherche est déterminée par le paramètre CLASSPATH. Dans la JVM de la base de données, c'est un module de résolution (resolver) qui prend en charge les références entre classes, packages et fichiers de ressources.

Pour une JVM externe, la variable d'environnement CLASSPATH indique les chemins d'accès disques des classes, des packages ou des archives, compressées, contenant les classes et packages. Quand une classe est appelée, la JVM externe recherche cette classe dans les emplacements spécifiés par la variable CLASSPATH.

La notion de répertoire (conteneur de fichiers) ou de fichier n'existe pas dans la base Oracle 10g. À la place, on trouve des schémas (propriétaires d'objets) et des objets. De même, Oracle 10g ne permet pas de spécifier des variables d'environnement internes dans la base. Le paramètre CLASSPATH est donc remplacé par le resolver. La JVM d'Oracle 10g utilise son resolver pour trouver les classes Java référencées par les autres classes.

Si les classes références appartiennent à des schémas Oracle différents du schéma dans lequel les classes vont être chargées, il faut explicitement indiquer le nom des autres schémas au resolver.

La base de données maintient automatiquement à jour dans son dictionnaire les dépendances entre les classes Java chargées dans chaque schéma.

Ainsi, comme une JVM externe sait manipuler des packages ou des classes isolées, stockées dans des arborescences différentes, la base de données peut manipuler sans erreur des packages ou des classes isolées dont les schémas propriétaires sont différents.

Contexte d'exécution d'une classe Java

À l'instar des applications clientes Oracle exécutant des ordres SQL ou appelant du code PL/SQL, une session Oracle est nécessaire pour utiliser les classes Java déployées dans la base de données Oracle.

Implicitement, toutes les manipulations de données sur les objets de la base de données sont effectuées dans le cadre de la session Oracle ouverte pour l'exécution du code Java.

Chaque session Oracle possède son propre contexte d'exécution en termes de gestion de la mémoire, de gestion des instances de classes et des variables statiques. Par exemple, le mécanisme de nettoyage (garbage collector) de la machine virtuelle Java traite les instances de classes par session.

Par analogie avec une machine virtuelle Java externe à la base, le contexte d'exécution d'une classe Oracle est comparable à un process de JVM. La base de données gère autant de contextes Java qu'il existe de sessions faisant appel à des programmes Java.

Néanmoins, pour ne pas pénaliser les performances du système hébergeant la base de données, la base de données Oracle 10g n'alloue pas à chaque session toutes les ressources nécessaires à un process d'une JVM traditionnelle. La JVM d'Oracle 10g est développée pour que chaque session ne consomme qu'un faible surcoût de mémoire. La première session invoquant une classe Java provoque l'octroi des ressources pour démarrer la machine virtuelle Java de la base de données. Puis cet espace mémoire commun est partagé pour chaque session. Seule la mémoire nécessaire à l'exécution des méthodes des classes Java est allouée à chaque session.

Compiler une classe Java

Avant d'utiliser une classe Java déployée ou non dans la base de données Oracle 10g, le code source de cette classe doit être transformé en bytecodes : ce traitement est réalisé par la compilation Java.

Oracle 10g offre plusieurs alternatives pour la compilation des classes Java stockées dans la base :

- compilation préalable au chargement dans la base ;
- compilation au moment du chargement ;
- compilation au moment de l'exécution.

Pour compiler une classe Java avant son chargement dans la base Oracle 10g, l'outil standard javac peut être invoqué dans le système d'exploitation. Ensuite, au lieu d'indiquer le nom du fichier source Java, il faut charger directement la classe compilée dans la base de données.

La compilation au moment du chargement est réalisée par l'outil loadjava décrit dans la section suivante.

L'outil loadjava peut également charger un fichier source Java sans le compiler.

Dans la suite du chapitre, nous présenterons la compilation native des classes Java qui a pour but d'accélérer les temps d'exécution des classes Java déployées dans la base Oracle 10g.

Charger des objets Java avec l'utilitaire loadjava

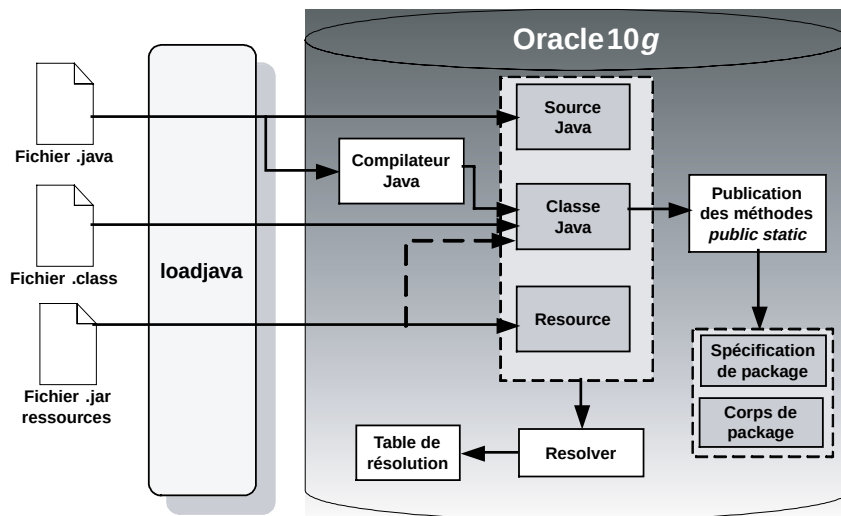
Pour que la base de données Oracle 10g soit capable d'exécuter une méthode Java, il faut charger la classe ou les classes correspondantes dans la base de données.

À cet effet, il convient de recourir à l'utilitaire loadjava fourni avec Oracle 10g et qui réalise les opérations suivantes :

- chargement des fichiers source Java (ou des classes Java compilées) et des fichiers de ressources ;
- chargement effectif des classes Java dans la base de données en tant qu'objets de schémas.

Figure 20-2

Principe
de l'utilitaire
loadjava



L'utilitaire loadjava charge chacun des fichiers qui lui sont spécifiés et exécute les traitements opportuns. Les fichiers source Java sont compilés par le compilateur Java d'Oracle 10g, puis enregistrés comme objets compilés Java. Les fichiers de classes Java sont enregistrés directement comme objets Java. Les fichiers de ressources (comme les fichiers de propriétés) sont stockés comme objets ressources Java. Une table d'options contient des informations supplémentaires sur les fichiers. Si le fichier à charger est une archive JAR ou ZIP, alors loadjava charge séparément chaque fichier de l'archive.

Au moment de l'exécution d'une application Java, les classes sont chargées dynamiquement. Pour remplacer la variable d'environnement CLASSPATH, la machine virtuelle Java d'Oracle 10g utilise un resolver (l'outil resolver). La résolution d'une classe consiste à définir la liste des objets nécessaires pour l'exécuter. Le resolver enregistre les dépendances pour chaque classe chargée dans la JVM d'Oracle 10g. Le resolver marque ensuite une classe comme valide si toutes les classes dont elle dépend sont valides. Dès qu'une classe devient invalide (rechargement par exemple), les classes qui appellent cette classe deviennent invalides. Les classes invalides ne sont plus exécutables ou appelables jusqu'à ce que le resolver les qualifie de valides. L'action du resolver a lieu, soit au moment du chargement de la classe, soit ultérieurement au moment de l'exécution d'une classe. Par défaut, le resolver recherche les classes dans le schéma courant de l'utilisateur connecté et dans le schéma PUBLIC. Le schéma SYS contient tous les packages Java standard comme java.lang et les packages système Java d'Oracle 10g comme oracle.aurora. Des synonymes publics sont automatiquement créés pour les classes standard Java et les classes système Java d'Oracle 10g. Si les classes, ou packages, appelées ont été chargées dans un autre schéma Oracle, il faut spécifier le paramètre resolver de la commande loadjava.

Oracle 10g inclut la génération automatique de classes Java « vides » à l'utilitaire loadjava. Les classes Java générées automatiquement remplacent les classes inexistantes

dans la base de données mais obligatoires pour la compilation des classes Java chargées. L'analyse des classes manquantes n'est appliquée que pour les fichiers Java chargés sous forme de classes. L'analyse n'est pas effectuée pour les fichiers source Java car la détection des références manquantes dans les fichiers sources Java est compliquée.

L'option associée à cette fonctionnalité de l'utilitaire loadjava est `genmissing`.

La syntaxe de loadjava est la suivante :

```
loadjava <options> <nom des classes>
<nom des classes> : liste des classes à charger
```

Voici quelques options de l'outil loadjava :

Option	Description
user ou u	chaîne de connexion à la base de données JDBC OCI : user/password[@tns] JDBC Thin : user/password@serveur:port:SID
Thin	utilisation du pilote JDBC Thin
oci ou oci8	utilisation du pilote JDBC OCI
grant userA, userB	accorde le privilège EXECUTE sur les classes en cours de chargement aux comptes Oracle userA et userB ; équivalent à <i>grant execute</i> pour les procédures PL/SQL
resolve	demande la résolution pendant le chargement de la classe
resolver [(« package ou classe » « schema ») (« package ou classe » « schema »)...]	paramètre de la résolution de classes
synonym	crée un synonyme public pour les classes à charger ; la résolution de la classe sera possible depuis n'importe quel schéma
genmissing	génération des classes manquantes (l'implémentation des méthodes est vide)
genmissingjar	En plus des actions de genmissing, création d'un jar qui contient les classes générées
publish « package »	crée ou remplace le package « package » pour écrire des méthodes wrapper pour les méthodes Java (des classes à charger) éligibles à la publication sous forme de procédures stockées, utilisable seulement avec les fiches '.class'. La méthode Java doit appartenir à une classe <i>public</i> , doit être déclarée <i>public</i> et <i>static</i> , et ne doit comporter que des types de paramètres convertibles directement dans des types de valeurs PL/SQL.
synonym	crée des synonymes publics pour les classes chargées (nécessite le privilège CREATE PUBLIC SYNONYM)
schema	indique dans quel schéma les objets Java sont à charger (nécessite les privilèges CREATE ANY PROCEDURE, CREATE ANY TABLE)

Option	Description
tableschema	Schéma dans lequel seront stockées les tables utilisées par loadjava. Par défaut, elles sont stockées dans le schema où sont chargées les classes.
verbose	Affiche des informations au chargement des fichiers. Recommandé pour trouver les classes manquantes plus tôt !
order	Résout les classes dans l'ordre croissant
noverify	Ne vérifie pas le bytecode
force	Force le chargement de toutes les classes. Normalement, les classes identiques aux classes déjà chargées ne sont pas rechargées.
definer	Spécifie que les méthodes des classes chargées seront exécutées avec les privilèges de créateur, et non de simple utilisateur. Par défaut, les méthodes s'exécutent avec les privilèges de leur utilisateur. Des créateurs différents peuvent avoir des privilèges différents, et une application peut avoir de nombreuses classes, donc assurez-vous que les méthodes d'une classe donnée s'exécutent avec seulement les privilèges requis.

Ci-dessous un exemple pour spécifier de prendre en compte toutes les classes et packages Java du schéma USER1 et toutes les classes et packages du schéma PUBLIC :

```
-resolver '(((* USER1) (* PUBLIC)))'
```

Chargeons le fichier Salarie.class dans le schéma SCOTT, avec la commande shell suivante :

```
loadjava -oci8 -user scott/tiger Salarie.class
```

La classe Salarie possède une méthode public static augmenteSalaire qui prend comme paramètre le nom d'un employé et un taux d'augmentation de salaire. Elle augmente le salaire de l'employé du taux indiqué.

Nous allons charger la classe Salarie dans la base de données et publier automatiquement la méthode augmenteSalaire en tant que procédure stockée. Le principe des procédures stockées Java est présenté plus loin dans ce chapitre.

Le paramètre publish de loadjava permet de spécifier le nom du package PL/SQL à créer pour contenir une procédure d'appel de la méthode Java augmenteSalaire.

La signature de la méthode augmenteSalaire est la suivante :

```
public static void augmenteSalaire(String p_salarie, float p_taux)
```

Il n'est pas nécessaire d'indiquer à loadjava quelle méthode publier en tant que procédure stockée : l'utilitaire détecte automatiquement les méthodes qui satisfont aux critères pour être publiées en tant que procédures stockées. La méthode Java doit appartenir à une classe public, doit être déclarée public et static et ne doit comporter que des types de paramètres convertibles directement dans des types de valeurs PL/SQL.

```
loadjava -publish pSalarie -oci8 -user scott/tiger Salarie.class
```

Cette commande va charger la classe Java Salarie en tant que classe dans la base de données puis créer le package pSalarie avec une méthode augmenteSalaire.

Elle est équivalente à :

- charger la classe Java dans la base de données :

```
loadjava -oci8 -user scott/tiger Salarie.class
```

- créer le package pSalarie avec SQL*Plus :

```
>sqlplus scott/tiger
SQL> CREATE OR REPLACE PACKAGE P_SALARIE AUTHID CURRENT_USER AS
  PROCEDURE AUGMENTE_SALAIRE(arg0 VARCHAR, arg1 NUMBER) AS LANGUAGE JAVA NAME
    ➔ 'entreprise.Salarie.augmenteSalaire(java.lang.String ,float )';
END;
/
SQL>
```

Supprimer des objets Java avec l'utilitaire dropjava

L'utilitaire dropjava est utilisé pour supprimer une classe, des packages ou des fichiers de ressources Java chargés dans la base de données Oracle 10g. Si la classe a été chargée à partir d'une archive JAR, les fichiers contenus dans cette archive ne sont pas forcément supprimés : seules les classes qui dépendent de la classe supprimée sont signalées comme invalides. Les classes qui héritent de la classe supprimée le sont également.

La syntaxe de dropjava est similaire à celle de loadjava :

```
dropjava -user <utilisateur>[/<mot de passe>[@<connexion>]] <options> <nom des classes>
<utilisateur>[/<mot de passe>] : nom et mot passe pour se connecter à la base de données;
<connexion> : chaîne de connexion à la base; voir la syntaxe de loadjava
<nom des classes> : liste des classes à supprimer dans le schéma de l'utilisateur
➔ qui ouvre la connexion
```

Par exemple, pour supprimer les classes contenues dans le fichier archive unearchive.jar qui ont été chargées dans le schéma SCOTT :

```
dropjava -verbose -oci8 -user scott/tiger unearchive.jar
```

Charger des objets Java en PL/SQL

La méthode PL/SQL loadjava du package dbms_java joue un rôle identique à l'utilitaire loadjava qui est lancé à l'extérieur de la base.

Elle charge des objets Java dans la base de données, à la différence près qu'on peut l'exécuter depuis un bloc PL/SQL.

Par exemple :

```
begin
  dbms_java.loadjava('e:\digora\deploy\unearchive.jar', '-resolve');
end;
```

Supprimer des objets Java en PL/SQL

La méthode PL/SQL `dropjava` du package `dbms_java` joue un rôle identique à l'utilitaire `dropjava` qui est lancé à l'extérieur de la base.

Elle supprime des objets Java dans la base de données, à la différence près qu'on peut l'exécuter depuis un bloc PL/SQL.

```
begin
  dbms_java.dropjava('e:\digora\deploy\unearhive.jar');
end;
```

Options de compilation par défaut

Si aucune option de compilation n'est indiquée pendant le chargement d'une classe, la JVM prend des valeurs par défaut. Les valeurs par défaut du compilateur sont enregistrées dans la table `JAVA$OPTIONS` du schéma où sont chargées les classes. Si la table `JAVA$OPTIONS` n'existe pas, elle est automatiquement créée.

Le package PL/SQL `DBMS_JAVA` fournit trois méthodes pour modifier les valeurs des options de compilation par défaut pour chaque package.

Méthode	Paramètres	Description
<code>set_compiler_option</code>	name, option, value	affecte la valeur d'une option
<code>get_compiler_option</code>	name, option	récupère la valeur d'une option sous forme d'une chaîne de caractères
<code>reset_compiler_option</code>	name, option	réaffecte la valeur initiale d'une option

Le paramètre `name` spécifie pour quels packages s'applique l'option par défaut. Si le paramètre `name` est une chaîne vide, alors l'option s'applique pour tous les packages.

Le paramètre `option` indique le nom de l'option de configuration. C'est une valeur parmi les trois possibles : `online`, `debug` ou `encoding`.

Le paramètre `value` contient la valeur de l'option de configuration.

Consulter les objets Java stockés dans Oracle 10g

Comme pour tous les objets créés et stockés dans Oracle 10g, des vues d'administration sont fournies en standard pour consulter les objets Java stockés dans la base.

Les informations sur les objets Java (fichiers source, classes Java, fichiers de ressource) sont accessibles dans les vues `DBA_OBJECTS` et `USER_OBJECTS`.

La vue `USER_OBJECTS` contient les informations sur les objets Java mis en œuvre dans la base de données, dans le schéma de l'utilisateur courant. La vue `DBA_OBJECTS` élargit la visibilité des objets à tous les schémas de la base de données. La vue `DBA_OBJECTS` n'est accessible que par des utilisateurs Oracle DBA.

Par défaut, ces vues affichent le nom des objets Java sous une forme codifiée. Pour rendre lisible le nom des objets, il faut utiliser la fonction `dbms_java.longname` qui transforme

le nom d'objet codifié en un nom d'objet textuel lisible. La fonction réciproque de `dbms_java.longname` est `dbms_java.shortname` qui renvoie le nom d'une classe sous forme codifiée.

Les types d'objets renvoyés dans ces vues pour les objets Java sont :

- `JAVA SOURCE` pour les codes source Java ;
- `JAVA CLASS` pour les classes Java ;
- `JAVA RESOURCE` pour les autres objets Java.

Le statut des objets Java est, soit `VALID`, soit `INVALID`. Si le statut d'un objet Java est `INVALID`, cela signifie qu'il n'a pas pu être compilé, soit pour des erreurs de syntaxe, soit pour des erreurs de dépendances non trouvées.

La commande suivante extrait de cette vue le nom des objets Java, leur type (classe Java, fichiers de ressource Java) et leur statut (valide ou invalide).

```
select replace(dbms_java.longname(object_name), '/', '.') as
  ↳ " OBJET JAVA ", object_type, status from user_objects where object_type like 'JAVA%';
```

Par exemple, la requête précédente produit un affichage de type :

OBJET JAVA	OBJECT_TYPE	STATUS
Entreprise	JAVA CLASS	VALID
META-INF.MANIFEST.MF	JAVA RESOURCE	VALID
connections.properties	JAVA RESOURCE	VALID

3 rows selected.

Cet exemple montre que le schéma de la base possède :

- une classe Java intitulée `ENTREPRISE` ;
- deux fichiers de ressources, *meta-inf.manifest.mf* et *connections.properties*.

L'interface shell ojvmjava

La base de données Oracle 10g n'est plus livrée avec cet utilitaire pour accéder à l'espace des objets Java d'un schéma sous forme d'un environnement shell.

Le script `ojvmjava.bat` offre notamment la possibilité d'exécuter des applications Java, c'est-à-dire d'invoquer la méthode static `main` des classes Java.

Le script `ojvmjava.bat` était livré avec la version Oracle9i.

Pour le recréer en environnement MS Windows :

- 1) recopier le fichier `loadjava.bat` sous le nom `ojvmjava.bat` ;
- 2) ouvrir le fichier `ojvmjava.bat` ;
- 3) remplacer la chaîne `oracle.aurora.server.tools.loadjava.LoadJavaMain` par la chaîne :
`oracle.aurora.server.tools.shell.ShellClient`.

La liste des commandes disponibles est limitée :

Commandes	Description
Echo	affiche un message sur la sortie standard
Java	programme Java similaire au programme Java livré dans le JDK par exemple : Java Uneclasse invoque la méthode main de la classe Uneclasse
Exit	quitte l'environnement ojvmjava
Help	Aide
Version	version de l'utilitaire
Whoami	nom de l'utilisateur connecté

Compilation native des classes Java dans Oracle 10g

La portabilité des applications Java repose sur le principe suivant. À la différence des applications écrites en C par exemple, les compilateurs Java ne créent pas des programmes directement exécutables sur une plate-forme particulière. Le code source Java est transformé en bytecodes qui sont indépendants de toute plate-forme d'exécution. Ces bytecodes sont traduits par une machine virtuelle Java au moment de leur exécution. La machine virtuelle Java est propre à chaque environnement d'exécution (système d'exploitation et matériel). En terme de performance, la version interprétée d'une application Java est plus lente que la version compilée de la même application.

Pour pallier cet inconvénient, les machines virtuelles Java peuvent intégrer des compilateurs natifs. Ces compilateurs natifs traduisent les bytecodes Java dans du code natif à la façon d'un compilateur C par exemple. Les performances s'en trouvent améliorées car le mécanisme d'interprétation des bytecodes est supprimé.

Deux stratégies sont possibles pour la compilation native des bytecodes Java :

- la compilation dite JIT (Just-In-Time) ;
- la compilation statique.

Avec la compilation JIT, la machine virtuelle Java détecte les blocs d'instructions exécutés le plus fréquemment et transforme ces instructions en code natif. Cette compilation a lieu pendant l'exécution de l'application Java et aucun programme exécutable n'est créé.

La compilation statique consiste à traduire les bytecodes Java en code C puis à compiler ce code C avant l'exécution de l'application. L'application Java ainsi compilée peut exploiter au mieux les capacités de la plate-forme d'exécution.

À l'installation des fichiers binaires de la base de données Oracle 10g, les bibliothèques Java standard fournies avec Oracle 10g sont compilées nativement.

Tous les programmes Java peuvent profiter de cette fonctionnalité de la machine virtuelle d'Oracle 10g.

Configuration de la compilation native Oracle 10g

Plusieurs étapes de configuration sont nécessaires avant de pouvoir utiliser la compilation native d'Oracle 10g sur le serveur hébergeant la base de données.

La configuration concerne, d'une part l'environnement logiciel du système d'exploitation, d'autre part l'octroi de privilèges particuliers au sein de la base de données.

Outil de génération des bibliothèques dynamiques DLL

Pendant la compilation native Oracle 10g, l'outil `ncomp` génère des bibliothèques dynamiques Windows (DLL). Oracle 10g n'est pas livré avec un outil permettant la génération de bibliothèques dynamiques DLL, donc un produit tiers doit être installé. Le logiciel Visual C++ de Microsoft est un exemple de produit qui peut être utilisé pour la compilation native d'Oracle 10g.

Une partie des paramètres de la compilation native d'Oracle 10g est spécifiée dans le fichier `Settings_os.properties`. Ce fichier se trouve dans le répertoire du module Java Accelerator d'Oracle 10g : `%ORACLE_HOME%\javavm\jahome`.

Le fichier des paramètres de compilation est à modifier en fonction du produit choisi pour compiler des fichiers source C et générer des DLL. Le contenu du fichier `Settings_os.properties` est préconfiguré pour s'exécuter avec Microsoft Visual C++.

Un paramètre reste cependant à adapter en fonction du répertoire d'installation de Visual C++ sur le serveur : `visual.c.home`. La valeur de ce paramètre est le chemin complet du produit Visual C++ sur le serveur.

Par exemple :

```
visual.c.home = c:/Program Files/MSVisualStudio60/VC
```

Environnement Java du système d'exploitation

Pour utiliser les utilitaires Oracle 10g de compilation native (`ncomp`, `deploync` et `statusnc`), un environnement JDK 1.4 doit être présent sur le serveur.

La base de données Oracle 10g est installée avec un environnement JDK 1.4 dans le répertoire `%ORACLE_HOME%\jdk`.

Le chemin d'accès complet au JDK 1.4 doit être affecté à la variable d'environnement `JAVA_HOME`.

Par exemple :

```
JAVA_HOME = c:\oracle\ora10g\jdk
```

La variable d'environnement `JAVA_HOME` peut être positionnée :

- directement dans les paramètres système de Windows ;
- dans les scripts `deploync.bat`, `ncomp.bat` et `statusnc.bat` ;
- dans le script appelant les utilitaires Oracle 10g.

Configuration dans la base de données

Avant de pouvoir utiliser la compilation native des classes Java, l'utilisateur Oracle 10g propriétaire des classes Java doit recevoir des privilèges spécifiques. En effet, pour la compilation native d'Oracle 10g, il est nécessaire d'accéder avec tous les droits aux fichiers du système d'exploitation (génération de fichiers temporaires, des DLL, déplacement de fichiers, etc.) et de lancer des programmes externes (compilation, génération des DLL).

Ces privilèges sont accordés via des rôles standard Oracle :

- JAVA_DEPLOY ;
- JAVASYSPRIV.

L'octroi est réalisé avec un client SQL, par exemple SQL*Plus, en étant connecté DBA.

```
c:\digora\ncmp> sqlplus system/manager
SQL>grant java_deploy,javasypriv to scott ;
Autorisation de privilèges (GRANT) acceptée.
```

Utilitaires Oracle 10g pour la compilation native

Oracle 10g est livré avec trois utilitaires pour utiliser la compilation native des classes Java chargées dans la base de données :

- ncomp, pour la compilation et le déploiement des classes Java compilées nativement dans la base de données ;
- deploync, pour le déploiement des classes Java compilées nativement dans la base de données ;
- statusnc, pour la vérification du statut des classes Java compilées nativement déployées dans la base de données.

Les utilitaires sont lancés par des scripts shell Windows (ncmp.bat, deploync.bat, statusnc.bat) stockés dans le répertoire %ORACLE_HOME%\bin.

Lancer la compilation native avec l'utilitaire Oracle ncomp

La compilation native d'Oracle 10g est réalisée par l'utilitaire ncomp. Cet utilitaire opère de la façon suivante :

- traduction des bytecodes en fichiers source C génériques ;
- compilation des fichiers source C ;
- génération de bibliothèques dynamiques DLL sur le serveur hébergeant la base de données ;
- installation des bibliothèques dynamiques DLL dans l'arborescence Oracle et mise à jour du dictionnaire de la base de données.

L'utilitaire `ncomp` ne peut compiler que des classes déjà chargées dans la base de données.

Si les classes à compiler nativement ne sont pas déjà chargées dans la base de données, l'utilitaire `ncomp` peut charger (option `load`) ces classes Java en invoquant implicitement `loadjava`. En effet, la ou les classes à compiler nativement sont indiquées à `ncomp` sous la forme d'un fichier de classe ou d'un fichier archive JAR contenant des classes. L'utilitaire `loadjava` est présenté en détail dans la partie traitant le chargement des objets Java.

Syntaxe de l'utilitaire `ncomp` :

```
ncomp <options> <fichier de classes>
```

```
<options>
```

```
<fichier de classes> : soit une archive compressée JAR ou ZIP contenant les classes
```

```
➔ à compiler ou soit une classe Java à compiler.
```

Options	Signification
user ou u	chaîne de connexion à la base de données JDBC OCI : user/password[[@tns]] JDBC Thin : user/password@serveur:port:SID
force	force la compilation native de toutes les classes
load	charge les classes spécifiées dans la base
Thin	utilisation du pilote JDBC Thin
oci8	utilisation du pilote JDBC OCI (par défaut)
noDeploy	uniquement génération d'un fichier archive JAR avec les classes compilées
outputJarFile	nom du fichier archive JAR (avec chemin absolu) généré par <code>ncomp</code> et qui contient les fichiers pour le déploiement des classes compilées nativement
lightweightDeploy- ment	déploiement séparé des librairies de la compilation native et des informations de la compilation. Les librairies ne sont pas stockées dans le fichier archive JAR avec les informations de compilation.
projectDir ou d	chemin absolu du projet pour la génération des fichiers source, la compilation et la génération des librairies dynamiques
update	met à jour les librairies et le fichier de déploiement avec les nouvelles classes non encore compilées nativement
verbose	affiche tous les messages de la compilation native

Les noms des options de `ncomp` sont sensibles à la casse.

Si vous spécifiez une valeur au paramètre `projectDir`, le répertoire de travail de `ncomp` est la valeur du paramètre, sinon le répertoire de travail est le répertoire courant d'où est lancé `ncomp`. Tous les fichiers temporaires de `ncomp` sont stockés dans le répertoire de travail (fichiers d'include, fichiers source C, fichiers make, fichiers de log, librairies, etc.). Si le paramètre `ProjectDir` n'est pas spécifié, alors tous les fichiers temporaires de `ncomp` sont enregistrés dans le répertoire courant.

Si le mode verbose n'est pas explicitement activé avec l'option verbose, les messages de ncomp sont enregistrés dans le fichier de log ncomp.log du répertoire de travail (répertoire courant ou valeur de projectDir).

Avec l'option noDeploy, les fichiers de déploiement et les bibliothèques dynamiques sont générés, mais ces dernières ne sont pas déployées sur le serveur et dans la base de données Oracle 10g. Il est conseillé de l'utiliser avec l'option outputJarFile qui spécifie le nom de l'archive JAR contenant tous les fichiers nécessaires au déploiement futur des classes compilées.

Prenons l'exemple d'un fichier JAR monjar.jar contenant une classe monpackage/Ma classe.class.

L'utilitaire ncomp va charger la classe monpackage/Ma classe.class dans Oracle 10g puis créer une bibliothèque dynamique DLL sur le serveur hébergeant la base de données.

```
c:\digora\ncomp> ncomp -u scott/tiger -load monjar.jar

#
# this list is produced by query
#   select status, class_name  from jaccelerator$status;
#

NEED_NCOMPING monpackage/Ma classe
NEED_NCOMPING monpackage/
# Deployment History, produced by query:
# select timestamp, status, dll_name  from jaccelerator$dlls order by dll_name
Sat Sep 01 16:29:47 GMT 2005 installed /orajox9_XXXXXX_scott_monpackage.dll
```

Les erreurs de compilation sont accessibles, soit en interrogeant la table JACCELERATOR\$DLL_ERRORS, soit avec l'utilitaire Oracle statusnc. L'utilitaire statusnc est présenté plus loin dans ce chapitre.

La bibliothèque dynamique qui vient d'être générée par l'utilitaire ncomp se trouve dans le répertoire %ORACLE_HOME%\javavm\admin.

```
c:\digora\ncomp>cd %ORACLE_HOME%\javavm\admin
c:\digora\ncomp>dir orajox10_*_scott_monpackage.dll
orajox10_XXXXXXX_scott_monpackage.dll
```

Vous remarquerez que les deux dernières parties du nom de la bibliothèque sont le nom du schéma où est chargée la classe Java suivi du nom du package de cette classe Java. Si la classe Java n'appartient à aucun package, la bibliothèque est nommée orajox10_XXXXXXX_scott_UnnamedPackage.dll.

Pour générer uniquement un fichier archive JAR de déploiement (depmonjar.jar) :

```
c:\digora\ncomp> ncomp -oci8 -user scott/tiger -noDeploy -outputJarFile
➡c:\digora\depmonjar.jar -load monjar.jar

#
```

```
# this list is produced by query
#  select status, class_name  from jaccelerator$status;
#

NEED_NCOMPING monpackage/MaClass
NEED_NCOMPING monpackage/
```

Le fichier depmonjar.jar est créé dans le répertoire c:\digora. Les librairies dynamiques des classes Java ne sont pas déployées dans la base de données.

Déployer des librairies dynamiques de classes Java avec l'utilitaire deploync

Précédemment, nous avons décrit l'utilitaire ncomp qui compile nativement des classes Java. Cet utilitaire offre la possibilité de déployer automatiquement ou non les librairies des classes compilées nativement. Si la seconde alternative (option noDeploy) a été choisie, seul un fichier archive JAR est construit : cette archive JAR contient les classes compilées nativement prêtes à être déployées.

Le déploiement des classes compilées nativement doit alors être réalisé par un autre utilitaire d'Oracle 10g : deploync.

L'utilitaire deploync prend le fichier archive JAR des classes compilées nativement comme source de données.

```
deploync <options> <fichier archive>
```

<fichier archive> : une archive compressée JAR contenant les classes compilées nativement

Options	Signification
user ou u	chaîne de connexion à la base de données JDBC OCI : user/password[@tns] JDBC Thin : user/password@serveur:port:SID
Thin	utilisation du pilote JDBC Thin
oci ou oci8	utilisation du pilote JDBC OCI

Pour déployer les classes compilées du package monpackage :

```
c:\digora\ncomp> deploync -oci8 -user scott/tiger -noDeploy -outputJarFile
➡c:\digora\depmonjar.jar -load monjar.jar
```

```
# Deployment History, produced by query:
# select timestamp, status, dll_name from jaccelerator$dlls order by dll_name
Sat Sep 01 16:29:47 GMT 2005 installed /orajox9_XXXXXX_scott_monpackage.dll
```

La librairie dynamique DLL qui vient d'être déployée est copiée dans le répertoire %ORACLE_HOME%\javavm\admin.

Vérifier la compilation native avec l'outil statusnc

L'outil statusnc permet de valider la compilation native d'une ou plusieurs classes Java.

Syntaxe de l'utilitaire statusnc :

```
statusnc -u <utilisateur>/<mot de passe>[@<alias TNS>] <classes>

<utilisateur>/<mot de passe> : connexion à la base de données.
<alias TNS> : alias TNS pour accéder à la base de données.
<fichier de classes> : soit une archive compressée JAR ou ZIP contenant les classes
➔ à compiler ou soit une classe Java à compiler.
```

Options	Signification
user ou u	chaîne de connexion à la base de données JDBC OCI : user/password[@tns] JDBC Thin : user/password@serveur:port:SID
Thin	utilisation du pilote JDBC Thin
oci ou oci8	utilisation du pilote JDBC OCI (par défaut)
output	fichier où diriger le flux de sortie de l'utilitaire
projectDir ou d	chemin absolu du projet pour la génération des fichiers source, la compilation et la génération des bibliothèques dynamiques

Si vous spécifiez une valeur au paramètre projectDir, le répertoire de travail de statusnc est la valeur du paramètre, sinon le répertoire de travail est le répertoire courant d'où est lancé statusnc. Tous les fichiers temporaires de statusnc sont stockés dans le répertoire de travail. Si le paramètre projectDir n'est pas spécifié, alors tous les fichiers temporaires de statusnc sont enregistrés dans le répertoire courant.

Si le mode verbose n'est pas explicitement activé avec l'option verbose, les messages de statusnc sont enregistrés dans le fichier de log ncomp.log du répertoire de travail (répertoire courant ou valeur de projectDir).

L'utilitaire statusnc indique le statut des classes Java compilées nativement :

- ALREADY_NCOMPED si la classe a été compilée nativement avec succès ;
- NEED_NCOMPING si la classe est chargée dans la base de données mais n'a pas encore été compilée nativement ;
- INVALID si la classe est invalide.

Reprenons l'exemple de la classe Java monpackage/Maclasse.class.

```
c:\digora\ncomp> statusnc -user scott/tiger monjar.jar

#
# this list is produced by query
# select status, class_name from jaccelerator$status;
#
```



```
ALREADY_NCOMPED monpackage/MaClass  
ALREADY_NCOMPED monpackage/
```

Le fichier `exact_class_list.txt` créé dans le répertoire de travail liste un état à la fois des classes Java et des bibliothèques des classes Java compilées nativement. Par exemple :

```
# Windows NT x86  
  
# Classes that are already ncomped  
  
# ALREADY_NCOMPED monpackage/MaClass  
  
# Ncomp libraries are ncomped  
  
# ALREADY_NCOMPED monpackage/
```

Les procédures stockées Java dans Oracle 10g

Dans les versions antérieures à Oracle8i du serveur de base de données Oracle, on recourait uniquement au langage PL/SQL pour développer des applications s'exécutant dans la base Oracle. Le code créé se présentait sous forme de sous-programmes que la base stockait en tant qu'objets : les procédures stockées. Ces objets pouvaient être des procédures, des fonctions ou des déclencheurs de base de données. Avec Oracle 10g, les développeurs peuvent écrire la logique métier d'une application en langage PL/SQL ou en langage Java. Comme pour le code PL/SQL, le code Java déployé dans Oracle 10g peut être composé de procédures, de fonctions et de déclencheurs. Dans Oracle 10g, les procédures stockées Java sont exécutées comme les procédures stockées PL/SQL. Une procédure PL/SQL peut appeler une procédure stockée Java et réciproquement.

Méthodes Java et procédures stockées Java

Déclarer une procédure stockée Java revient à écrire une classe comprenant une méthode static et à déclarer une méthode wrapper PL/SQL pour appeler cette méthode. En effet, le moteur PL/SQL d'Oracle 10g ne fournit pas d'API pour instancier directement des classes. Les seuls composants Java que le moteur PL/SQL peut utiliser sont les méthodes public static de classes Java déclarées public. En revanche, rien n'empêche la méthode static appelée en tant que procédure stockée d'instancier elle-même des classes Java.

Pour qu'une méthode Java puisse être publiée en tant que procédure stockée, elle doit respecter une autre règle : les types de tous les paramètres de la méthode Java, y compris le type de retour pour une fonction, doivent être convertibles en type de paramètre PL/SQL.

Écrire une procédure stockée Java

Nous allons créer une procédure stockée qui renvoie le nom d'un employé et met à jour son salaire. Comme nous l'avons expliqué précédemment, une procédure stockée Java est en réalité une méthode static membre d'une classe Java.

Nous définissons donc une classe Java Salarie, avec une méthode getEmpGrade qui renvoie le nom d'un employé, recherché à partir de son identifiant. Cet exemple utilise l'API JDBC pour exécuter les requêtes SQL sur la table EMP du schéma SCOTT.

Le code source montre un exemple de chaîne de connexion pour ouvrir une connexion JDBC depuis une classe Java stockée dans la base de données. Cet exemple utilise le pilote JDBC Oracle disponible dans la machine virtuelle Java d'Oracle 10g.

Comme les ordres SQL sont exécutés dans le contexte de la session Oracle ouverte pour lancer la classe, il n'est pas nécessaire de spécifier des paramètres de connexion.

```
import java.sql.*;
import oracle.jdbc.driver.*;
import oracle.*;

class Salarie
{

    public static String getEmpGrade (int empid) throws SQLException
    {
        String nom = null;
        Connection conn = null;
        Statement stmt = null;

        try
        {
            DriverManager.registerDriver
            (new oracle.jdbc.driver.OracleDriver ());
            /* la chaîne de connexion utilise le pilote JDBC kprb intégré dans Oracle 10g
            il n'est pas nécessaire de spécifier d'utilisateur et de mot de passe car la requête
            est exécutée dans la session courante */
            conn = DriverManager.getConnection ("jdbc:oracle:kprb:");
            stmt = conn.createStatement();
            ResultSet rset = stmt.executeQuery(
                "SELECT JOB FROM EMP WHERE EMPNO = " + empid);
            while (rset.next())
            {
                nom = rset.getString (1);
            }
        }
        finally
        {
            stmt.close();
            return nom;
        }
    }
}
```

Déployer la procédure stockée Java

La première étape de déploiement d'une procédure stockée Java consiste à charger la classe Java dans Oracle 10g.

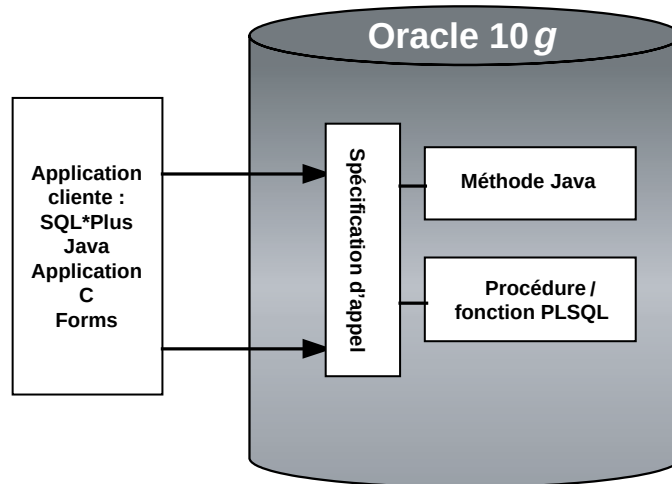
Cette opération peut être réalisée par l'utilitaire loadjava.

Publier manuellement une procédure stockée Java

Avant d'appeler des méthodes Java à partir d'instructions SQL, il faut publier ces méthodes dans le dictionnaire de données Oracle. Le schéma suivant montre comment une application cliente accède à une méthode Java dans Oracle 10g. Pour appeler la méthode Java, l'application utilise la spécification d'appel associée à cette méthode. Le dictionnaire de données d'Oracle 10g recherche alors la méthode Java associée à la spécification d'appel et exécute cette méthode Java.

Figure 20-3

Principe de publication
d'une méthode



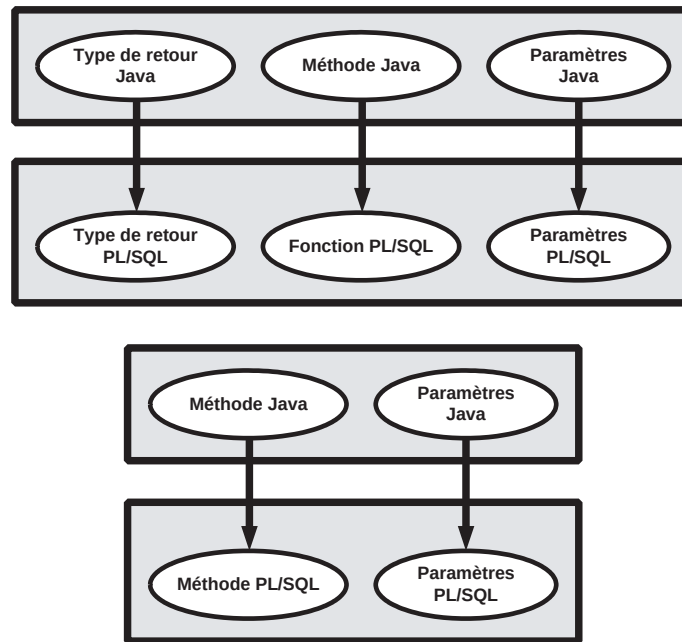
Au moment du chargement d'une classe Java à l'intérieur de la base, les classes ne sont pas publiées, parce qu'Oracle ne connaît pas les méthodes qui seront accessibles à partir des instructions SQL. La publication d'une méthode Java consiste à définir des spécifications d'appel (*call specifications*). Elles indiquent la correspondance entre :

- les noms des méthodes SQL et les noms des méthodes Java ;
- les types des arguments SQL et Java ;
- les types des valeurs de retour SQL et Java.

En résumé, publier une méthode revient à écrire une signature de sous-programme PL/SQL, analogue à la signature de la méthode écrite en Java.

Figure 20-4

Publication des méthodes
Java en procédures stockées



Une méthode Java sans type de retour est publiée en tant que procédure PL/SQL. Une méthode Java qui retourne une valeur est publiée en tant que fonction PL/SQL. Le nom des sous-programmes PL/SQL peut être différent de celui des méthodes Java. Concernant le type des paramètres, la classe `STRING` de Java devient une variable `VARCHAR2` dans PL/SQL ; de même, un argument `float` en Java devient un `NUMBER` dans PL/SQL. Cette transposition de type de données entre Java et PL/SQL montre la perte de précision des valeurs numériques quand elles sont converties plusieurs fois d'un système Java vers un système PL/SQL. Il ne faut pas négliger cet aspect quand une application travaille à la fois avec des procédures stockées PL/SQL et des méthodes Java.

Définir les spécifications d'appel d'une procédure stockée Java

Une spécification d'appel doit appartenir au même schéma que la procédure stockée équivalente.

Pour écrire une spécification d'appel, il faut créer :

- soit une fonction ou une procédure PL/SQL, dans un package ou non ;
- soit une méthode membre d'un type objet SQL.

Une méthode Java ne pourra être publiée que si elle est déclarée `public` et `static` (méthodes de classe). Les méthodes d'instance peuvent être publiées en tant que méthodes membres de type objet SQL.

Définir des spécifications d'appel SQL

SQL*Plus permet de définir manuellement des spécifications d'appel. Pour définir une spécification d'appel d'une procédure il faut utiliser l'instruction SQL standard CREATE OR REPLACE PROCEDURE.

```
CREATE OR REPLACE PROCEDURE <nom procedure>
AUTHID {DEFINER | CURRENT_USER}
AS LANGUAGE JAVA
NAME '<nom de la méthode>(<types des arguments>) <type de retour>' ;
Pour définir une spécification d'appel d'une fonction :
CREATE OR REPLACE FUNCTION <nom procedure>
AUTHID {DEFINER | CURRENT_USER}
AS LANGUAGE JAVA
NAME '<nom de la méthode>(<types des arguments>) <type de retour>' ;
```

La clause AUTHID détermine si une procédure stockée s'exécute avec les droits de son propriétaire ou les droits de l'appelant.

Par exemple, pour appeler la méthode getEmpGrade depuis du code SQL ou PL/SQL, il faut d'abord la publier.

Pour publier la méthode getEmpGrade, nous créons une fonction PL/SQL getEmpGrade qui renvoie une valeur VARCHAR2. La méthode Java getEmpGrade renvoie un objet String qui correspond à une valeur VARCHAR2 dans SQL. L'argument Java dans le type int devient un argument de type NUMBER dans SQL.

```
create or replace function getEmpGrade (empid IN number) return varchar2 is
    language java name 'Salarie.getEmpGrade(int)
    return java.lang.String';
```

Définir des spécifications d'appel dans des packages PL/SQL

Un package PL/SQL est un objet qui regroupe des types, des éléments et des sous-programmes. Il comprend deux parties : la spécification et le corps. La spécification correspond à l'interface pour les applications ; le corps implémente la spécification du package.

Dans l'exemple suivant, la classe Java Deptmanager est constituée de deux méthodes pour ajouter ou supprimer un employé dans la table EMP du schéma SCOTT.

Impossible de trouver les sources pour les classes ci dessous :

```
public class modifEmp {
    public static void nvEmp(String empNom) throws SQLException {
        ...
    }

    public static void supEmp(int empID) throws SQLException {
        ...
    }
}
```

Les deux méthodes sont liées car elles agissent sur la même table et font partie d'une même classe Java. Il est donc souhaitable de regrouper leurs spécifications d'appel dans un package PL/SQL.

Nous allons créer le package PL/SQL référençant les deux méthodes Java. Deux nouvelles méthodes PL/SQL sont nécessaires pour appeler ces méthodes depuis PL/SQL : chaque méthode PL/SQL servira à appeler la méthode homologue Java.

Pour définir le package PL/SQL il faut déterminer la spécification et le corps du package.

Spécification du package

La spécification du package déclare les méthodes qu'une application ou une procédure peut appeler à l'extérieur du package. Ici, nous déclarons deux méthodes.

```
CREATE OR REPLACE PACKAGE modifemp AS
  PROCEDURE nvemp(empnom VARCHAR2) ;
  PROCEDURE supemp(empid NUMBER) ;
END modifemp;
```

Définition du corps du package : les spécifications d'appel des méthodes Java

Le corps du package implémente les méthodes PL/SQL. Dans notre exemple, les méthodes mises en œuvre sont des spécifications d'appel de méthodes Java.

```
CREATE OR REPLACE PACKAGE BODY modifemp AS
  PROCEDURE nvemp(empnom VARCHAR2);
  AS LANGUAGE JAVA
  NAME 'modifEmp.nvEmp(java.lang.String)';
  PROCEDURE supemp(empid NUMBER) ;
  AS LANGUAGE JAVA
  NAME 'modifEmp.supemp(int)';
END modifemp;
```

Finalement, pour tester les procédures Java stockées, l'instruction CALL suffit. L'instruction ci-dessous appelle la procédure supemp du package PL/SQL modifemp que nous venons de créer.

```
CALL modifemp.supemp(1);
```

En réalité, l'appel de la procédure PL/SQL modifemp.supemp se transforme en un appel de la méthode de classe modifEmp.supemp écrite en Java. Les spécifications d'appel cachent l'implémentation Java de la méthode. L'utilisateur de la méthode PL/SQL supemp n'a pas besoin de connaître l'implémentation de la méthode. Il demande juste l'exécution du service Supprimer un employé. Cet exemple révèle la simplicité de remplacement des méthodes PL/SQL par des méthodes Java, et inversement.

Définir des spécifications d'appel dans des types objet

Le langage SQL est compatible avec le modèle objet en proposant les types objet. Un type objet (object type) est un type de données défini par l'utilisateur. Il est composé

d'attributs (les données) et de méthodes (fonctions et procédures) permettant de manipuler ces données.

Un type objet est défini par deux éléments : sa spécification et son corps. La spécification correspond à l'interface pour les applications : elle déclare les attributs et les méthodes de type objet. Le corps implémente la spécification en définissant les traitements des sous-programmes PL/SQL et les spécifications d'appel des méthodes Java. À chaque méthode précisée dans la spécification correspond un sous-programme PL/SQL ou une spécification d'appel dans le corps de type objet. Prenons l'exemple d'une entreprise. Un objet entreprise possède les attributs suivants :

- un nom ;
- un numéro ;
- une adresse.

Définissons une classe Java appelée Entreprise et comportant une méthode `getNomEntr` qui renvoie le nom d'une entreprise. Cette classe Java possède trois attributs : `entrNom`, `entrId`, `entrAdresse` qui représentent respectivement le nom, le numéro et l'adresse d'une entreprise.

```
import java.sql.*;
import java.io.*;
import oracle.sql.*;
import oracle.jdbc2.*;
import java.math.BigDecimal;

public class Entreprise implements SQLData{
    private String entrNom;
    private BigDecimal entrId;
    private String entrAdresse;

    public String getNomEntr() {
        return entrNom;
    }

    String typeSQL;
    public String getSQLTypeName() throws SQLException {
        return typeSQL;
    };

    public void readSQL(SQLInput flux, String nomType) throws SQLException {
        typeSQL = nomType;
        entrNom = flux.readString();
        entrId = flux.readBigDecimal();
        entrAdresse = flux.readString();
    }
}
```

```
public void writeSQL(SQLOutput flux) throws SQLException {  
    flux.writeString(entrNom);  
    flux.writeBigDecimal(entrId);  
    flux.writeString(entrAdresse);  
}  
}
```

Dans la base, nous créons le type objet Entreprise : il possède les mêmes attributs que la classe Java Entreprise et une méthode `getNom` qui renvoie le nom de la société. Cette méthode est une spécification d'appel pour la méthode Java `Entreprise.getNomEntr`.

```
CREATE TYPE Entreprise AS OBJECT (  
    nom VARCHAR2(50),  
    entId NUMBER(2),  
    adresse VARCHAR2(50),  
    MEMBER FUNCTION getNom RETURN VARCHAR2  
    AS LANGUAGE JAVA  
    NAME 'Entreprise.getNomEntr() return java.lang.String'  
);
```

Pour stocker les instances de type Entreprise, nous créons une table entreprises :

```
CREATE TABLE entreprises OF Entreprise;
```

Nous alimentons la table entreprises :

```
INSERT INTO entreprises VALUES('Cie des Ordinateurs',1, 'PARIS');  
INSERT INTO entreprises VALUES('Ets AZERTY',2, 'STRASBOURG');  
INSERT INTO entreprises VALUES('La boutique du CDROM',3, 'LYON');  
INSERT INTO entreprises VALUES('SuperStore',4, 'TOULOUSE');
```

Il suffit d'écrire un bloc PL/SQL anonyme pour tester le type objet Entreprise. Ce bloc recherche une instance d'Entreprise et appelle sa méthode `getNom`.

```
declare  
    entr Entreprise;  
    id NUMBER(2);  
begin  
    -- affecter à id le numéro de l'entreprise  
    id := 2;  
    -- entr pointe vers l'instance d'objet Entreprise  
    select value(e) into entr from entreprises e where entid=id;  
    -- affichage du nom de l'entreprise  
    dbms_output.put_line('Nom de la société : '||entr.getNom);  
end;
```

Le bloc PL/SQL précédent doit produire un résultat de type :

```
Nom de la société : Ets AZERTY
```


Appeler une procédure stockée Java

Pour utiliser les procédures stockées Java publiées dans Oracle 10g, plusieurs solutions sont possibles :

- appeler des méthodes stockées directement depuis SQL*Plus ;
- appeler des méthodes Java depuis un déclencheur de base de données ;
- appeler des méthodes Java depuis des sous-programmes PL/SQL ;
- appeler des méthodes PL/SQL depuis des méthodes Java ;
- appeler des méthodes Java comme membres d'un type objet.

Appeler une méthode Java depuis SQL*Plus

L'instruction SQL CALL permet d'appeler des méthodes Java publiées dans des packages PL/SQL ou dans des types objet SQL.

Avec SQL*Plus, la syntaxe de la méthode CALL est la suivante :

```
CALL <schéma>.<package>.<procédure>(arguments);  
CALL <schéma>.<package>.<fonction>(arguments) INTO :<variable>;
```

Par exemple :

```
CALL modifemp.nv_emp(:empNom);
```

Si le code Java envoie des informations sur la sortie standard, on peut rediriger cette sortie vers le buffer SQL*Plus avec la commande suivante :

```
SET SERVEROUTPUT ON  
CALL dbms_java.set_output(2000) ;
```

Appeler une méthode Java depuis un déclencheur (trigger) de base de données

Un déclencheur de base de données est un sous-programme lié à une table ou une vue. Oracle 10g déclenche le trigger automatiquement quand une certaine opération SQL de manipulation de données est exécutée sur la table ou la vue. Les triggers PL/SQL sont décrits au chapitre 17, *Programmer avec PL/SQL*.

Exemple : nous allons créer un déclencheur qui permet de savoir si le salaire d'un employé dépasse 10 000 après augmentation.

Tout d'abord, nous définissons une classe Java Triggerjava. Le fichier source trigger-Java.sqlj est le suivant :

```
import sqlj.runtime.ref.*;  
import java.sql.*;  
  
public class triggerJava {
```

```
public static void testSalaire (int empID, float nvData) throws SQLException {  
    try {  
        #sql {INSERT INTO controle VALUES (:empID,:nvData)};  
    }  
    catch (SQLException e) {  
        System.err.println(e.getMessage());  
    }  
}
```

Cette classe comporte la méthode `testSalaire` qui va enregistrer dans la table `Controle` les employés dont le salaire dépasse 10 000 après augmentation.

Une requête SQL ne peut pas être exécutée si aucune session n'est ouverte. Comme la procédure s'exécute dans la base de données, la connexion à la base est liée à la session qui a permis de lancer la procédure stockée.

La classe Java est désormais créée. Il reste à définir la spécification d'appel que nous appelons `test_salaire`.

```
CREATE OR REPLACE PROCEDURE test_salaire (  
    empid NUMBER, nvsalaire NUMBER)  
AS LANGUAGE JAVA  
NAME 'triggerJava.testSalaire(int,float)';
```

`test_salaire` est une procédure, car elle ne renvoie aucune valeur. La table `Controle` est créée par la commande SQL suivante :

```
CREATE TABLE controle (  
    empid NUMBER,  
    salaire NUMBER);
```

Enfin, il faut définir le déclencheur, que nous appelons `sal_trig`. C'est un déclencheur de mise à jour qui est évalué à chaque mise à jour de la colonne `sal`.

```
CREATE OR REPLACE TRIGGER trigger_salaire  
AFTER UPDATE OF sal ON emp  
FOR EACH ROW  
WHEN (new.sal > 10000)  
CALL test_salaire(:new.empno, :new.sal)
```

L'appel de la procédure stockée `test_salaire` est réalisé par l'instruction `CALL`.

Quand l'instruction `UPDATE` affecte une valeur à la colonne `SAL` d'une ligne de la table `EMP`, le trigger vérifie la clause `WHERE`. Si la clause `WHERE` est valide, le trigger appelle la méthode `test_salaire` en lui fournissant les arguments nécessaires :

- le numéro de l'employé ;
- le nouveau salaire de cet employé.

`test_salaire` ajoute alors une nouvelle ligne dans la table `Controle`. Le trigger est exécuté pour chaque ligne modifiée par l'instruction `UPDATE`.

Vérifions maintenant que le trigger `trigger_salaire` et la procédure stockée `test_salaire` remplissent leur rôle. Le script suivant vide la table `Controle` puis augmente de 20 % le salaire de chaque employé. La table `Controle` contient alors tous les employés dont le salaire est supérieur à 10 000 après augmentation.

```
SQL> DELETE * FROM controle;
SQL> UPDATE emp SET sal = sal*1.2;
SQL> SELECT * FROM controle;
EMPID      SALAIRE
-----
7499      10200
7566      11850
7698      11700
7782      11220
7788      11880
7839      14280
7844      10080
7876      22680
7902      11880
9 rows selected.
```

Appeler des fonctions Java depuis des instructions de manipulation de données SQL

Les instructions `SELECT`, `INSERT`, `UPDATE` et `DELETE` peuvent appeler les méthodes Java publiées en tant que fonctions.

L'exemple suivant montre comment formater la valeur d'une colonne `SELECT` avec une fonction de formatage écrite en Java.

Nous définissons la classe `AFFEMP`, qui comprend la méthode `modifEmpNom`. La méthode `modifEmpNom` reçoit le nom de l'employé et de sa fonction et renvoie une chaîne concaténant ces deux éléments. Voici le code source du fichier `affEmp.java` :

```
public class affEmp {
    public static String modifEmpNom (String empNom, String empFonction) {
        return (new String(empNom+" est un "+empFonction));
    }
}
```

Il faut publier la méthode `formatEmp`. Nous créons une spécification d'appel de type `FUNCTION`, car `formatEmp` renvoie une chaîne de caractères.

```
CREATE OR REPLACE FUNCTION aff_emp (nom VARCHAR2, job VARCHAR2)
RETURN VARCHAR2
AS LANGUAGE JAVA
NAME 'affEmp.modifEmpNom (java.lang.String, java.lang.String)
    return java.lang.String';
```

Nous allons tester la méthode Java `affEmp.modifempnom` avec une instruction `SELECT` sur la table `EMP` du schéma `SCOTT`. Pour chaque ligne renvoyée par l'instruction

SELECT, la fonction `aff_emp` est appelée avec indication du nom et de la fonction de l'employé :

- la colonne `ename` contient le nom de l'employé ;
- la colonne `job` contient la fonction de l'employé.

La requête SELECT et son résultat sont indiqués ci-dessous, à titre d'exemple :

```
SQL> SELECT aff_emp(ename,job) AS "Nom_Job" FROM emp;
Nom_Job
-----
SMITH est un CLERK
ALLEN est un SALESMAN
WARD est un SALESMAN
JONES est un MANAGER
MARTIN est un SALESMAN
BLAKE est un MANAGER
CLARK est un MANAGER
SCOTT est un ANALYST
KING est un PRESIDENT
TURNER est un SALESMAN
ADAMS est un CLERK
JAMES est un CLERK
FORD est un ANALYST
MILLER est un CLERK
14 rows selected.
```

Appeler des méthodes Java depuis des blocs PL/SQL

Une procédure stockée Java peut être appelée depuis n'importe quel bloc PL/SQL. Définissons une classe `Salaire` qui comporte une méthode augmentant le salaire d'un employé d'une somme donnée. Cette méthode `augSalaire` utilise un argument qui identifie le salarié et la valeur de l'augmentation de salaire. Voici le code source du fichier `SalaireMaj.sqlj` :

```
import sqlj.runtime.ref.*;
import java.sql.*;

public class SalaireMaj {
    public static void augSalaire (int empid, float valeur) throws SQLException {
        #sql {UPDATE emp SET sal = sal+:valeur WHERE empno=:empid};
    }
}
```

Publions la méthode Java `augSalaire` en tant que procédure PL/SQL sous le nom `augmente_salaire` (la méthode Java `augSalaire` ne renvoie pas de valeur).

```
CREATE OR REPLACE PROCEDURE augmente_salaire (empid NUMBER, valeur NUMBER)
AS LANGUAGE JAVA
NAME 'SalaireMaj.augSalaire(int, float)';
```

On peut appeler la méthode `augmente_salaire` :

- à partir d'un bloc PL/SQL anonyme ;
- à partir d'une procédure PL/SQL ;
- à partir d'une méthode d'un type objet.

Par exemple, le bloc PL/SQL anonyme suivant appelle la procédure stockée `augmente_salaire` en lui indiquant l'identifiant de l'employé (7 788) et la hausse de salaire (1 000).

```
select sal from emp where empno=7788;
DECLARE
empid NUMBER;
valeur NUMBER;
BEGIN
empid:=7788;
valeur:=1000;
augmente_salaire(empid,valeur);
END;
/
select sal from emp where empno=7788;
```

Le résultat suivant est produit par cette séquence de commandes :

```
SAL
-----
      13 900
1 row selected.
```

Le salaire initial du salarié 7788 est 13 900. Nous exécutons le bloc PL/SQL anonyme et nous vérifions le nouveau salaire du salarié.

```
SQL> DECLARE
2> empid NUMBER;
3> valeur NUMBER;
4> BEGIN
5> empid:=7788;
6> valeur:=1000;
7> augmente_salaire(empid,valeur);
8> END;
9> /
Statement processed.
SQL> select sal from emp where empno=7788;
SAL
-----
      14 900
1 row selected.
```

Le salaire final du salarié 7 788 s'élève maintenant à 14 900.

Comment a été exécuté le bloc PL/SQL ?

L'appel de méthode `augmente_salaire(empid,valeur)` a réalisé les tâches suivantes :

- passage des valeurs `empid` et `valeur` à la méthode Java `Salaire.augSalaire` ;
- exécution de la méthode Java `Salaire.augSalaire`.

Appeler des sous-programmes PL/SQL depuis des méthodes Java

JDBC et SQLJ permettent d'appeler des procédures stockées PL/SQL depuis des méthodes Java.

La fonction PL/SQL `empfonction` renvoie la fonction d'un employé. Elle prend un argument de type `NUMBER` et retourne une valeur de type `VARCHAR2`.

```
CREATE OR REPLACE FUNCTION empfonction (empid NUMBER) RETURN VARCHAR2 IS
empGrade VARCHAR2(50);
BEGIN
SELECT job INTO empGrade FROM emp WHERE empno = empid;
RETURN empGrade;
END;
```

Testons `empfonction` depuis SQL*Plus :

```
SQL> select empfonction(7788) as Fonction from dual;
Fonction
-----
ANALYST
```

Le bloc d'instructions Java suivant remplit ces fonctions :

- il appelle la fonction PL/SQL `empfonction` ;
- il enregistre la valeur de retour dans une chaîne de caractères ;
- il affiche cette chaîne.

```
String chaine;
#sql chaine = {VALUES(empfonction(:empid))};
System.out.println("Fonction : "+chaine);
```

Le fichier `SalInfo.sqlj` suivant utilise ce bloc d'instructions :

```
import sqlj.runtime.ref.*;
import java.sql.*;

public class SalInfo {
    public SalInfo() throws SQLException {
        // Enregistrement du pilote JDBC Oracle
        DriverManager.registerDriver(new oracle.jdbc.driver.OracleDriver());
        // Ouverture d'une session dans la base
        // Définition d'un contexte pour les instructions SQLJ
        try {
            Connection conn = DriverManager.getConnection("jdbc:oracle:thin:@nomServeur:1521:orcl",
```

```
        "scott","tiger");
        DefaultContext.setDefaultContext(new DefaultContext(conn));
    }
    catch (SQLException exception) {
        System.out.println("Impossible d'ouvrir la session");
        System.err.println(exception);
        System.exit(1);
    }
}

public static void main (String [] args) throws SQLException {
    SalInfo application= new SalInfo();
    application.fonction(7788);
}

public static void fonction (int empid) throws SQLException {
    String chaine;
    #sql chaine = {VALUES(empfonction(:empid))};
    System.out.println("Fonction : "+chaine);
}
}
```

L'exécution de ce code génère le message suivant :

```
Fonction : ANALYST
```

Cet exemple a montré comment invoquer simplement des procédures stockées PL/SQL à partir de méthodes Java grâce au standard SQLJ.

Résumé

Ce chapitre a présenté l'architecture Java d'Oracle 10g. Vous avez découvert :

- le fonctionnement de la machine virtuelle Java ;
- le déploiement des objets Java dans la base de données ;
- les possibilités d'utilisation de Java en complément du langage SQL.

Vous êtes désormais prêt à développer des logiciels évolutifs, sécurisés et portables.

XML et Oracle 10g

Dans ce chapitre :

- présentation générale d'XML ;
- stockage de contenu XML dans Oracle 10g ;
- manipulation de contenu XML avec les outils Oracle 10g.

XML, langage universel pour l'échange de données, s'est imposé comme référence en la matière. Oracle supporte cette norme depuis ses débuts, et accompagne les évolutions XML au fil des nouvelles versions de ses outils. C'est une fois de plus le cas, notamment grâce à XML DB et au support du XML Schéma qui facilitent la manipulation des données et la communication entre applications. Les dernières versions de ses outils sont livrées avec la base de données ou téléchargeables à partir du site <http://www.oracle.com/technology/index.html>.

À propos d'XML

Introduction

XML, eXtensible Markup Language, est un langage extensible de balisage de documents qui pallie la rigidité d'HTML (HyperText Markup Language) et la complexité de SGML (Structured Generalized Markup Language). Après sa ratification par le W3C (World Wide Web Consortium) en février 1998, ce langage est rapidement devenu un standard pour l'échange d'informations sur le Web. XML permet en effet de représenter sous forme de texte la structure propre des données, sans ambiguïté d'interprétation.

Le format HTML indique simplement la façon dont l'information qu'il contient est présentée. Prenons le cas d'un tableau HTML. L'extrait du fichier HTML présenté ci-après montre comment est défini un tableau. Ce tableau affiche :

- le numéro d'un employé ;
- son nom ;
- sa fonction ;
- son salaire.

Les informations de deux employés sont affichées. Sans l'en-tête du tableau, il serait difficile d'attribuer une signification aux mots SMITH et CLERK et aux nombres 7 369 et 800. Grâce à l'en-tête, nous savons que :

- 7 369 est le numéro de l'employé ;
- SMITH son nom ;
- CLERK sa fonction ;
- 800 son salaire.

```
<html>
<head><title>Liste des employés</title></head>
<body>
  <table border="2" bordercolor="#404D6A">
    <tr bgcolor="#89C7A4">
      <td>empID</td><td>empNom</td>
      <td>empFonction</td><td>empSal</td>
    </tr>
    <tr>
      <td>7369</td><td>SMITH</td>
      <td>CLERK</td><td>800</td>
    </tr>
    <tr>
      <td>7902</td><td>FORD</td>
      <td>ANALYST</td><td>3000</td>
    </tr>
    <tr>
      <td>7499</td><td>ALLEN</td>
      <td>SALESMAN</td><td>1600</td>
    </tr>
  </table>
</body>
</html>
```

Cet exemple met en évidence une faiblesse du standard HTML : la présentation des données est significative. Imaginons en effet que les titres des colonnes soient A, B, C et D : les données du fichier HTML seraient alors difficilement utilisables. De plus, si l'on

veut afficher uniquement le nom des employés sous forme d'une liste à puces, il faut construire un nouveau fichier HTML. Pour ajouter un nouvel employé, il est alors nécessaire de modifier les deux fichiers HTML.

XML permet de pallier les inconvénients liés au langage HTML. Dans un fichier au format XML, il est possible grâce aux balises d'identifier le contenu de l'information. Il est alors facile pour l'utilisateur comme pour l'ordinateur de comprendre la signification du document. Regardons comment enregistrer avec XML les informations relatives aux employés. Nous reprenons l'exemple précédent. Notre document XML est construit de la façon suivante :

- empListe représente l'élément racine ;
- les éléments empDef correspondent aux employés ;
- Le contenu de empDef représente un ensemble d'éléments (les données) caractérisant un employé.

```
<?xml version="1.0"?>
  <empListe>
    <empDef>
      <empID>7369</empID>
      <empNom>SMITH</empNom>
      <empFonction>CLERK</empFonction>
      <empSal>800</empSal>
    </empDef>
    <empDef>
      <empID>7902</empID>
      <empNom>FORD</empNom>
      <empFonction>ANALYST</empFonction>
      <empSal>3000</empSal>
    </empDef>
    <empDef>
      <empID>7499</empID>
      <empNom>ALLEN</empNom>
      <empFonction>SALESMAN</empFonction>
      <empSal>1600</empSal>
    </empDef>
  </empListe>
```

Ce document XML est facilement utilisable. À chaque information est associée une signification. Ainsi, XML et HTML sont complémentaires et chacun fait son travail : XML représente le contenu de l'information tandis que HTML est dédié à sa présentation.

Structure d'un document XML, les DTD et XML Schéma

Les DTD (Document Tag Définition)

Premier avantage du langage XML par rapport à HTML : l'abandon des balises figées. Avec XML, il est possible de créer ses propres balises représentant le sens et la structure

des données. Les balises peuvent être définies lors de leur utilisation dans un document XML, ou de manière formelle dans une DTD (Définition d'un type de document ou Document Type Definition), qui précise la grammaire d'une classe de document XML. Vous pouvez ainsi adapter votre application aux différents changements, en corrigeant ou ajoutant de nouvelles balises.

Un document XML comprend :

- un prologue ;
- un texte balisé.

Dans cet exemple de DTD, l'élément `empDef` est constitué d'un numéro d'employé, de son nom, d'une fonction et d'un salaire (tous ces éléments sont obligatoires) :

```
<!ELEMENT empDef (empID, empNom, empFonction, empSal)>
<!ELEMENT empID (#PCDATA)>
<!ELEMENT empNom (#PCDATA)>
<!ELEMENT empFonction (#PCDATA)>
<!ELEMENT empSal (#PCDATA)>
```

XML Schéma

Au fur et à mesure que XML a été utilisé dans des applications réelles, les utilisateurs se sont rendu compte que l'utilisation de DTD pour valider la structure et le contenu d'un document XML était insuffisante. Par exemple, dans le cas d'échange de méta-données au format XML dans une solution d'intégration d'application, il est vraiment indispensable de définir des contraintes très rigoureuses sur les documents XML transférés.

Les principales limitations de la DTD sont les suivantes :

- Elle ne supporte pas les types de données autres que les chaînes de caractères, ce qui est une limitation pour décrire des métadonnées standard, des schémas de base de données, etc.
- Les applications ont besoin de définir des structures de document plus flexible que ce que la DTD peut fournir.

C'est pour répondre à cette problématique que XML Schéma a été défini. L'objectif de ce standard est donc de décrire de la façon la plus précise possible un document XML en utilisant une structure de type XML. Le XML Schéma permet donc de valider de manière précise un document XML par sa structure ainsi que par le type de ses données (simple : chaîne de caractères, entiers... ou plus complexes : énumération, restriction...).

XML Schéma n'est pas un complément de la DTD mais un remplaçant car les informations fournies par la DTD sur la structure du document XML sont incluses dans XML Schéma.

Reprenons l'exemple de document XML, contenant une liste d'employés eux-même définis par un identifiant (un entier), leur nom (une chaîne de caractères), leur fonction

(analyst, clerk, manager ou salesman) et leur salaire (un réel), pour en déduire un XML Schéma de validation.

```
<?xml version="1.0"?>
<xsd:schema targetNamespace="http://www.ici.org" xmlns="http://www.ici.org"
  ➔ xmlns:xsd="http://www.w3.org/2000/10/XMLSchema" elementFormDefault="qualified">
  <xsd:element name="empListe">
    <xsd:complexType>
      <xsd:sequence>
        <xsd:element name="empDef" minOccurs="1" maxOccurs="unbounded">
          <xsd:complexType>
            <xsd:sequence>
              <xsd:element name="empID" type="xsd:int"/>
              <xsd:element name="empNom" type="xsd:string"/>
              <xsd:element name="empFonction">
                <xsd:simpleType>
                  <xsd:restriction base="xsd:string">
                    <xsd:enumeration value="ANALYST"/>
                    <xsd:enumeration value="CLERK"/>
                    <xsd:enumeration value="SALESMAN"/>
                    <xsd:enumeration value="MANAGER"/>
                  </xsd:restriction>
                </xsd:simpleType>
              </xsd:element>
              <xsd:element name="empSal" type="xsd:float"/>
            </xsd:sequence>
          </xsd:complexType>
        </xsd:element>
      </xsd:sequence>
    </xsd:complexType>
  </xsd:element>
</xsd:schema>
```

Mais cet exemple reste relativement simple car on peut effectuer des validations sur des champs plus complexes, par exemple un numéro de série de pièces qui peut avoir 2 formes :

- (1 chiffre)-(5 chiffres)-(3 chiffres)-(1 chiffre) ;
- (1 chiffre)-(3 chiffres)-(5 chiffres)-(1 chiffre).

```
<xsd:element name="num_serie">
  <xsd:simpleType>
    <xsd:restriction base="xsd:string">
      <xsd:pattern value="\d{1}-\d{5}-\d{3}-\d{1}|\d{1}-\d{3}-\d{5}-\d{1}"/>
    </xsd:restriction>
  </xsd:simpleType>
</xsd:element>
```

Dans la suite de ce chapitre, lorsqu'il sera fait référence à un « schéma XML », il s'agira de documents XML Schéma W3C.

Construction d'un document XML avec une DTD ou un schéma XML

Un document conforme à la structure et aux règles syntaxiques XML est dit bien formé. Il doit suivre les règles suivantes :

- il débute par la déclaration XML `< ?xml version= "1.0" ?> ;`
- ajout d'un élément si la validation est faite par une DTD. Exemple : `<!DOCTYPE empListe SYSTEM "employees.dtd" > ;`
- tous les éléments sont contenus dans un élément racine auquel on peut ajouter les attributs référençant le XML Schéma. Par exemple : `<empListe xmlns="http://www.ici.org" xmlns:xsi="http://www.w3.org/2000/10/XMLSchema-instance" xsi:schemaLocation="http://www.ici.org employees.xsd"> ;`
- les éléments sont stockés dans un arbre ;
- les éléments non vides doivent comporter une balise de début et de fin.

Un document XML conforme à une DTD ou à un XML Schéma est dit valide. Lors de l'analyse de ce document XML, il est vérifié qu'il est conforme au fichier de validation (DTD ou XML Schéma) et que toutes les données respectent les règles définies. Lors du stockage de données provenant d'un document XML dans une base, le document de validation peut être utilisé pour valider la structure, garantissant ainsi que les données insérées correspondent bien aux colonnes de la table. Si un document XML est généré par des données provenant d'une base, le document XML en résultant sera implicitement valide.

Présentation et affichage des données XML avec les feuilles de style

L'utilisation d'XML permet en outre de séparer le contenu d'un document de sa présentation.

À propos des feuilles de style et HTML

Au début de ce chapitre, nous avons comparé les documents HTML et XML. Nous avons vu qu'un document HTML contient à la fois des données et des indications sur leur mise en page. Créer un site Web nécessite une charte graphique ; c'est elle qui confère une identité propre au site Web. Chaque page HTML d'un site est formatée individuellement. La mise en page concerne le format des chaînes de caractères (police, taille, style, couleur), la justification des paragraphes, le positionnement des images, etc. La charte graphique peut évoluer et chaque modification est reportée sur toutes les pages les unes après les autres. Les feuilles de style simplifient l'application d'une charte graphique sur un site Web. Le W3C a établi un standard de feuille de style associée au langage HTML, à savoir la CSS Cascading Style Sheet. Une feuille de style définit les styles que l'on pourra appliquer sur les éléments des pages HTML. Par exemple, chaque niveau de titre dans une hiérarchie aura son propre style spécifiant sa police, sa taille, son style, sa justification. Pour un même site, on pourra créer plusieurs feuilles de style. Chaque page HTML est associée à une feuille de style utilisée au moment de l'affichage. Ainsi, il

suffira de modifier les caractéristiques d'un style dans la feuille de style et les pages HTML appelant ce style seront automatiquement mises à jour.

Les feuilles de style XML

Les données XML définissent la structure et le contenu, puis une feuille de style leur est appliquée afin d'en définir la présentation. Les données XML peuvent être présentées de manières différentes (tant en ce qui concerne leur apparence que leur organisation), et ce grâce à la feuille de style. Par exemple, la présentation d'une interface peut dépendre des utilisateurs qui vont y avoir accès : en fonction de leur profil, de leur type de navigateur ou d'autres critères, ils choisiront une feuille de style différente. Une feuille de style peut également permettre de transformer des données en un format prédéfini, destiné à une application spécifique qui reçoit et prépare des données. Les feuilles de style sont utilisables sur un serveur comme sur un poste client.

Le langage des feuilles de style d'XML constitue également l'un des sujets de travail du W3C. XSL (eXtensible Stylesheet Language) fournit des feuilles de style qui permettent le passage d'XML à HTML ou à tout autre langage balisé (WML...), la réorganisation ou le filtrage des données, dans le but de les afficher dans un navigateur Web.

Utilisation d'XML

Il existe différentes manières de concevoir une application Internet avec XML. Parmi les nombreux cas d'utilisation d'XML, voici les deux situations que nous traiterons dans ce chapitre :

- une présentation personnalisée des données ;
- l'échange des données entre plusieurs applications.

Présentation personnalisée des données

XML est de plus en plus utilisé pour personnaliser des données provenant de différents navigateurs, périphériques ou utilisateurs. L'utilisateur de documents XML mis en forme avec des feuilles de style peut transformer et présenter des données à l'attention de tout un ensemble de clients, y compris des navigateurs Web graphiques. Vous pouvez ainsi concevoir votre application sans vous soucier du client qui affichera les données résultat.

Par exemple, pour retrouver la présentation initiale sous forme de tableau des données XML de notre liste d'employés, il faudrait utiliser la feuille de style :

```
<?xml version="1.0" encoding="iso-8859-1"?>
<xsl:stylesheet version="1.0" xmlns:xsl="http://www.w3.org/1999/XSL/Transform">
  <xsl:template match="/">
    <html>
      <head><title>Liste des employés</title></head>
      <body>
        <xsl:apply-templates/>
      </body>
```

```
</html>
</xsl:template>
<xsl:template match="empListe">
  <table border="2" bordercolor="#404D6A">
    <tr bgcolor="#89C7A4">
      <td>empID</td><td>empNom</td>
      <td>empFonction</td><td>empSal</td>
    </tr>
  <xsl:apply-templates />
</table>
</xsl:template>
<xsl:template match="empDef">
<tr>
  <td><xsl:value-of select="empID"/></td>
  <td><xsl:value-of select="empNom"/></td>
  <td><xsl:value-of select="empFonction"/></td>
  <td><xsl:value-of select="empSal"/></td>
</tr>
</xsl:template>
</xsl:stylesheet>
```

L'utilisation conjointe d'XML et d'XSL facilite ainsi la création et la maintenance de sites Web dynamiques. Il est possible d'en modifier l'aspect en changeant simplement la feuille des styles XSL, tout en conservant la logique ou le code de la base.

Échange de données entre plusieurs applications

Un des défis pour les développeurs d'applications est de relier entre elles des données issues de plusieurs applications. En se concentrant sur les données sans les lier au réseau ou protocole de communication, XML rend la communication entre applications plus simple.

Les applications peuvent partager les données XML avec ou sans l'aide d'un XML Schéma commun (ou d'une DTD). Cependant, l'utilisation d'un XML Schéma commun rend l'échange des données plus fiable et leur gestion plus simple.

En général, le flux d'échange des données entre les applications partageant un XML Schéma commun est le suivant :

- l'application émettrice produit un document XML fondé sur le XML Schéma commun ;
- le document XML est envoyé à l'application de réception ;
- à sa réception, l'application teste la validité des données XML, traite le document et écrit les données dans la base ;
- l'application de réception transmet le document XML initial (ou un document régénéré) à l'application suivante.

Interrogation de documents XML : Langage W3C XML Path (XPath)

XPath est un langage utilisé pour effectuer des recherches, des interrogations ou mettre en place une navigation hypertexte dans des documents XML dont le nom vient de la notation proche de celle utilisée dans les URL. Il se base sur la structure logique abstraite du document plutôt que sur sa syntaxe. Sa conception permet également d'effectuer des recherches de correspondance, c'est-à-dire de vérifier si un nœud correspond à une expression prédéfinie. L'implémentation de XPath dans XML DB n'est pas complète : les chemins (*path*) ayant pour type de valeur de retour des booléens, des nombres ou des chaînes ne sont pas supportés.

XPath représente un document XML sous la forme d'un arbre de nœuds de différents types :

- nœuds d'élément ;
- nœuds d'attributs ;
- nœuds de texte.

Chemins de localisation (Location Paths)

Les chemins de localisation sont des expressions XPath qui permettent de trouver un nœud. Un chemin consiste en une succession de nœuds séparés par des « / ». Des mots-clés permettent d'interroger l'arborescence en fonction du nœud courant et sont utilisables via deux syntaxes différentes, une normale et une simplifiée.

La notation complète ne rentre pas dans le cadre de cet ouvrage, aussi nous n'examinerons que quelques exemples en notation simplifiée et supportés par XML DB :

- « / » : représente un chemin absolu vers l'élément recherché ;
- « // » : représente tous les éléments du document qui correspondent au critère suivant ;
- « * » : représente tous les éléments ;
- « [x] » : précise une occurrence dans la liste : x est son numéro d'apparition ;
- « @ » : spécifie un attribut ;
- « child::node() » : sélectionne les sous-éléments du nœud courant ;
- « attribute::* » : sélectionne les attributs du nœud courant.

Exemples

Considérons le fichier XML suivant :

```
<?xml version="1.0"?>
<ListeConsultants xmlns="http://www.digora.com"
xmlns:xsi="http://www.w3.org/2000/10/XMLSchema-instance">
```

```
xsi:schemaLocation="http://www.digora.com/consultants.xsd">
  <Consultant id="1">
    <Nom>Razibus</Nom>
    <Prenom>Jean-Charles</Prenom>
  </Consultant>
  <Consultant id="2">
    <Nom>Morel</Nom>
    <Prenom>Renaud</Prenom>
  </Consultant>
  <Consultant id="3">
    <Nom>Cossin</Nom>
    <Prenom>Laurent</Prenom>
  </Consultant>
  <Consultant id="4">
    <Nom>Lebeau</Nom>
    <Prenom>Raphaël</Prenom>
  </Consultant>
  <Consultant id="5">
    <Nom>Louis</Nom>
    <Prenom>Nicolas</Prenom>
  </Consultant>
</ListeConsultants>
```

Les expressions XPath peuvent s'écrire de la façon suivante.

Sélection de l'élément racine ListeConsultants :

```
■ /ListeConsultants
```

Sélection des sous-éléments Consultant de l'élément ListeConsultants :

```
■ /ListeConsultants/Consultant
```

Sélection des sous-éléments Nom des éléments Consultants qui sont enfants de ListeConsultant :

```
■ /ListeConsultants/Consultant/Nom
```

Sélection de tous les éléments Consultant :

```
■ //Consultant
```

Sélection de tous les éléments Consultant qui sont enfants de ListeConsultant :

```
■ //ListeConsultant/Consultant
```

Sélection de tous les sous-éléments Consultant qui sont enfants de ListeConsultant :

```
■ //ListeConsultant/Consultant/*
```

Sélection du consultant n°1 :

```
■ /ListeConsultant/Consultant[1]
```

Sélection du dernier consultant :

```
■ /ListeConsultant/Consultant[last()]
```

Sélection des consultants qui ont un attribut id :

```
■ /ListeConsultant/Consultant[@id]
```

Les produits Oracle XML

Oracle fournit plusieurs composants, utilitaires et autres interfaces qui peuvent tirer profit de la technologie XML en relation avec une base de données. Les produits proposés dépendent des conditions d'application, des préférences de programmation et des environnements de développement et de mise en œuvre. Les produits suivants prennent en charge XML :

- Oracle 10g XML DB ;
- Oracle Text (anciennement nommé *interMedia* Text) ;
- XML Parser et XSLT Processor (Java, C, C++, PL/SQL) ;
- XML Class Generator (Java et C++);
- XML SQL Utility pour Java ;
- XDK JavaBeans ;
- XSQL Servlet ;
- Oracle SOAP.

Stockage XML dans le référentiel Oracle XML DB

XML DB était une des grandes nouveautés d'Oracle9i release 2. Il s'agit d'un référentiel de stockage hiérarchique de documents XML, à la manière d'un système de fichiers. L'accès à ce référentiel se fait par des protocoles tels que WebDAV, FTP ou HTTP. On retrouve dans ce nouveau produit intégré à Oracle 10g des fonctionnalités qui proviennent d'Oracle 10g Internet File System.

Nous allons détailler la gestion d'arborescence, la gestion de versions, la sécurité et les protocoles d'accès.

Installation d'Oracle XML DB

XML DB fait normalement partie des composants installés par défaut et il est possible de configurer les protocoles et le référentiel avec une installation de base de données personnalisée.

Si XML DB n'a pas été installé lors de la mise en place de la base de données, il est possible de l'installer a posteriori de deux manières, soit avec l'assistant de configuration de base de données (DBCA), soit manuellement à l'aide de scripts.

Installation avec l'assistant de configuration de base de données (DBCA)

L'assistant effectue automatiquement :

- la création du tablespace dédié au référentiel ;
- la configuration des trois protocoles supportés ;
- FTP est configuré pour écouter sur le port 2100 ;
- HTTP/WebDAV est configuré pour écouter sur le port 8080.

Pour que l'accès anonyme soit permis, il faut débloquer le compte ANONYMOUS de la base de données.

Installation manuelle par des scripts

La première opération à effectuer est la création d'un tablespace qui va stocker les données du référentiel. Utilisez pour cela la commande `create tablespace`.

Les scripts d'installation de XML DB sont situés dans le dossier

`ORACLE_HOME\rdbms\admin` et doivent être lancés en tant qu'utilisateur SYS.

Script `catqm.sql`

Ce script va créer les tables et les vues de XML DB.

Syntaxe : `catqm.sql <mot de passe XDB> <nom du tablespace XDB> <nom du tablespace TEMP>`

Exemple :

```
catqm.sql xdb_pass XDB_TS TEMP
```

Script `catxdbj.sql`

Ce script va charger les objets Java de XML DB. Si vous venez de lancer `catqm.sql`, reconnectez-vous sous SYS avant de lancer `catxdbj.sql`.

Syntaxe :

```
catxdbj.sql
```

Manipulations post-installation

- Ajoutez le paramètre suivant avec la commande `ALTER SYSTEM` dans une session SQL/Plus sous SYS.

```
alter system set DISPATCHERS="(PROTOCOL=TCP) (SERVICE=<sid>XDB)" scope=BOTH;
```

- Vérifiez vos changes avec la commande SHOW PARAMETER

```
show parameter DISPATCHERS
NAME                TYPE                VALUE
dispatchers         string              (PROTOCOL=TCP) (SERVICE=<sid>XDB)
```

- Redémarrez la base de données et le listener

Suppression d'Oracle XML DB

Pour réinstaller XML DB il faut tout d'abord le désinstaller en supprimant l'utilisateur Oracle qui lui correspond puis en supprimant le tablespace qui lui est associé. Utilisez les commandes décrites ci-dessous :

```
drop user xdb cascade;
alter tablespace <XDB_TS_NAME> offline;
drop tablespace <XDB_TS_NAME> including contents;
```

Réinstallation d'Oracle XML DB

Configuration d'Oracle XML DB

La configuration de XML DB peut être faite via Enterprise Manager ou via le fichier de configuration /sys/xdmconfig.xml. Ce fichier est modifiable pendant le fonctionnement du serveur. Chaque session cliente utilise les paramètres du fichier de configuration courant lors de sa connexion et les conserve sauf si une commande de rechargement explicite du fichier est exécutée.

Le fichier de configuration xdbconfig.xml

Ce fichier est stocké dans le référentiel et se conforme au schéma XML xdbconfig.xsd, situé dans le chapitre *Oracle XML DB-Supplied XML Schemas and Additional Examples* de *Oracle® XML DB Developer's*.

Tag racine : <xdbconfig>

Ce tag est divisé en deux sous-sections : <sysconfig> et <userconfig>.

Tag <sysconfig>

Ce tag concentre la configuration système, comme par exemple le support de la sensibilité à la casse du texte, la durée de vie maximum d'une ACL, etc. La configuration spécifique des protocoles est enregistrée dans le tag <protocolconfig>.

Il est subdivisé de la façon suivante :

```
<sysconfig>
  paramètres généraux
  <protocolconfig> ... </protocolconfig>
</sysconfig>
```

Tag <userconfig>

Ce tag va contenir les paramètres que vous souhaiteriez y ajouter.

Tag <protocolconfig>

Sous le tag <common> on va retrouver les paramètres qui s'appliquent à tous les protocoles. La configuration des protocoles FTP et HTTP se trouve respectivement sous les tags <ftpconfig> et <httpconfig>.

La structure de ce tag est simple :

```
<protocolconfig>
  <common> ... </common>
  <httpconfig> ... </httpconfig>
  <ftpconfig> ... </ftpconfig>
</protocolconfig>
```

Tag <httpconfig>

Sous le tag <httpconfig> se trouve encore une section qui paramètre les applications Web qui peuvent y être déployées : <webappconfig>.

Exemple de fichier de configuration :

```
<xdbconfig xmlns="http://xmlns.oracle.com/xdb/xdbconfig.xsd"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:schemaLocation="http://xmlns.oracle.com/xdb/xdbconfig.xsd"
<sysconfig>
  <acl-max-age>900</acl-max-age>
  <invalid-pathname-chars>,</invalid-pathname-chars>
  <call-timeout>300</call-timeout>
  <max-session-use>100</max-session-use>
  <default-lock-timeout>3600</default-lock-timeout>
  <xdbcore-logfile-path>/sys/log/xdblog.xml</xdbcore-logfile-path>
  <xdbcore-log-level>1</xdbcore-log-level>
  <protocolconfig>
    <common>
      <extension-mappings>
        <mime-mappings>
          <mime-mapping>
            <extension>au</extension>
            <mime-type>audio/basic</mime-type>
          </mime-mapping>
          <mime-mapping>
            <extension>avi</extension>
            <mime-type>video/x-msvideo</mime-type>
          </mime-mapping>
          <mime-mapping>
            <extension>bin</extension>
            <mime-type>application/octet-stream</mime-type>
          </mime-mapping>
        </mime-mappings>
      </extension-mappings>
    </common>
  </protocolconfig>
</sysconfig>
```

```

    <lang-mappings>
      <lang-mapping>
        <extension>en</extension>
        <lang>english</lang>
      </lang-mapping>
    </lang-mappings>

    <charset-mappings>
    </charset-mappings>

    <encoding-mappings>
      <encoding-mapping>
        <extension>gzip</extension>
        <encoding>zip file</encoding>
      </encoding-mapping>
      <encoding-mapping>
        <extension>tar</extension>
        <encoding>tar file</encoding>
      </encoding-mapping>
    </encoding-mappings>
  </extension-mappings>

  <session-pool-size>50</session-pool-size>
  <session-timeout>6000</session-timeout>
</common>

<ftpconfig>
  <ftp-port>2100</ftp-port>
  <ftp-listener>local_listener</ftp-listener>
  <ftp-protocol>tcp</ftp-protocol>
  <logfile-path>/sys/log/ftplog.xml</logfile-path>
  <log-level>0</log-level>
  <session-timeout>6000</session-timeout>
</ftpconfig>

<httpconfig>
  <http-port>8080</http-port>
  <http-listener>local_listener</http-listener>
  <http-protocol>tcp</http-protocol>
  <session-timeout>6000</session-timeout>
  <server-name>XDB HTTP Server</server-name>
  <max-header-size>16384</max-header-size>
  <max-request-body>2000000000</max-request-body>
  <logfile-path>/sys/log/httplog.xml</logfile-path>
  <log-level>0</log-level>
  <servlet-realm>Basic realm="XDB"</servlet-realm>
<webappconfig>
  <welcome-file-list>
    <welcome-file>index.html</welcome-file>

```

```

        <welcome-file>index.htm</welcome-file>
    </welcome-file-list>
    <error-pages>
</error-pages>
    <servletconfig>
        <servlet-mappings>
            <servlet-mapping>
                <servlet-pattern>/oradb/*</servlet-pattern>
                <servlet-name>DBURIServlet</servlet-name>
            </servlet-mapping>
        </servlet-mappings>
        <servlet-list>
            <servlet>
                <servlet-name>DBURIServlet</servlet-name>
                <display-name>DBURI</display-name>
                <servlet-language>C</servlet-language>
                <description>Servlet for accessing DBURIs</description>
                <security-role-ref>
                    <role-name>authenticatedUser</role-name>
                    <role-link>authenticatedUser</role-link>
                </security-role-ref>
            </servlet>
        </servlet-list>
    </servletconfig>
</webappconfig>
</httpconfig>
</protocolconfig>
</sysconfig>

<userconfig><numusers>40</numusers></userconfig>

</xdbconfig>

```

API de configuration dynamique

Dans le package DBMS_XDB se trouvent des fonctions qui permettent de gérer la configuration d'Oracle XML DB dynamiquement.

Fonction CFG_GET()

La fonction CFG_GET() récupère la configuration sous forme d'objet XMLType.

```
DBMS_XDB.CFG_GET() RETURN SYS.XMLTYPE
```

Fonction CFG_UPDATE()

La fonction CFG_UPDATE() met à jour la configuration avec un nouvel objet XMLType.

```
DBMS_XDB.CFG_UPDATE(nouvelleconfig SYS.XMLTYPE)
```


Fonction CFG_REFRESH()

Cette fonction met à jour la configuration de la session avec la version du fichier de configuration actif dans le référentiel.

```
DBMS_XDB.CFG_REFRESH()
```

Cette méthode est utile lorsque vous venez de mettre à jour la configuration et que vous souhaitez que votre session courante la prenne en compte.

Gestion d'arborescence

Oracle XML DB permet le stockage de documents XML sous la forme d'une arborescence d'objets, à la manière d'un système de fichiers. Cette arborescence est constituée de ressources (nœuds) qui peuvent être des documents XML ou des dossiers. Chaque ressource est identifiée par un chemin qui est de la forme « racine-ressource-séparateur-ressource... ».

Il existe deux moyens d'accéder à ces ressources :

- l'accès navigationnel ;
- l'accès basé sur l'interrogation de vues.

L'accès navigationnel

Il passe forcément par l'un des protocoles XML DB qui accèdent directement au référentiel. Les moyens d'interrogation sont le SQL, JNDI, JDBC, HTTP, FTP et WebDAV.

L'accès basé sur l'interrogation de vues

Il passe forcément par le langage SQL et ceux qui l'utilisent : JDBC, les JavaBeans, le PL/SQL. Il s'agit d'une interrogation de deux vues : PATH_VIEW et RESOURCE_VIEW qui permettent d'effectuer des opérations sur les ressources stockées dans le référentiel.

Vue RESOURCE_VIEW

Cette vue contient une ligne par ressource enregistrée dans le référentiel XML DB.

Description :

<u>Column</u>	<u>Datatype</u>
RES	SYS.XMLTYPE(XMLSchema "http://xmlns.oracle.com/xdm/XDBResource.xsd" Element "Resource")
ANY_PATH	VARCHAR2(4000)
RESID	RAW(16)

Vue PATH_VIEW

Cette vue contient une ligne par chemin d'accès unique à une ressource présente dans le référentiel.

Description :

Column	Datatype
PATH	VARCHAR2(1024)
RES	SYS.XMLTYPE(XMLSchema "http://xmlns.oracle.com/xdb/XDBResource.xsd" Element "Resource")
LINK	XMLTYPE
RESID	RAW(16)

API d'interrogation du référentiel XML DB

En plus des deux vues ci-dessus, Oracle 10g fournit une API d'interrogation composée d'opérateurs. Ces outils complètent l'exploitation des vues.

UNDER_PATH : cet opérateur permet de récupérer les sous-éléments (les ressources) contenus dans un dossier.

EQUALS_PATH : cet opérateur permet de récupérer une ressource positionnée à un chemin défini.

PATH : ce mot-clé est auxiliaire, il permet de récupérer le chemin relatif d'une ressource.

DEPTH : ce mot-clé est auxiliaire, il permet de récupérer la profondeur d'une arborescence en dessous d'une ressource.

Exemples :

Sélection des chemins qui commencent par /sys :

```
select any_path from resource_view where any_path like '/sys%';
```

Récupération du contenu d'une ressource :

```
select any_path from resource_view where under_path(res, '/sys') = 1;
```

Liste de tous les chemins relatifs à /sys sur trois niveaux :

```
select path(1) from resource_view where under_path (res, 3, '/sys',1)=1;
```

Liste des chemins et de leur profondeur relatifs à /sys :

```
select path(1) PATH,depth(1) depth from path_view where under_path(RES, 3, '/sys',1)=1;
```

Vérification de la localisation d'un chemin :

```
select any_path from resource_view where equals_path(res, '/sys') > 0;
```

Insertion de données dans le référentiel XML DB via les vues

Exemple d'insertion de données dans le référentiel à travers la vue RESOURCE_VIEW :

```
insert into resource_view values(sys.xmltype.createxml('
<Resource xmlns="http://xmlns.oracle.com/xdb/XDBResource.xsd"
➤xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:schemaLocation="http://xmlns.oracle.com/xdb/XDBResource.xsd
➤http://xmlns.oracle.com/xdb/XDBResource.xsd">
<Author>Jean-Charles Razibus</Author>
<DisplayName>Exemple</DisplayName>
<Comment>Commentaire d'exemple</Comment>
<Language>fr</Language>
<CharacterSet>ASCII</CharacterSet>
<ContentType>text/plain</ContentType>
</Resource>'), '/public/digora', NULL);
```

Suppression du référentiel XML DB via les vues

Suppression d'une ressource :

```
delete from resource_view where any_path = '/home/digora'
```

Suppression récursive de dossiers non vides :

```
delete from resource_view where under_path(res, '/home/digora.xml') = 1;
```

Gestion de version

XML DB propose de gérer de multiples versions de chaque ressource. En version 9.0.1, à chaque modification du contenu du document, les changements précédents étaient perdus. Depuis Oracle9i R2, XML DB permet de conserver un historique de toutes les versions d'une ressource modifiée à plusieurs reprises.

À l'utilisation, ce système est assez flexible :

- il offre la possibilité d'activer/désactiver le contrôle de version au niveau de la ressource ;
- l'incrémentation des versions est automatique ;
- la manipulation d'une ancienne version est aussi simple que la version courante ;
- le rechargement d'une version précédente est aisé.

Le package de gestion se nomme DBMS_XDB_VERSION.

Mise en place des versions

La mise en place de la gestion de version n'est pas active par défaut et elle n'est pas possible sur les dossiers, les listes d'accès et les ressources basées sur un schéma XML. C'est à la mise en place qu'est créée la première version d'une ressource.

Exemple de mise en place de la gestion de version :

```
declare
  resid DBMS_XDB_VERSION.RESID_TYPE;
begin
  resid := DBMS_XDB_VERSION.MakeVersioned('/home/digora/exemple.html');
end;
/
```

Attention, cette opération doit être validée par un ordre commit.

Il n'existe pas de chemin pointant directement sur une ancienne version d'une ressource, c'est pourquoi on trouve une notion d'*identifiant de version*. Cet identifiant peut être récupéré à l'aide de la méthode `DBMS_XDB.getResourceID()`. Avec cet ID, il est possible de travailler sur l'ancienne version, notamment de la recharger comme version courante, etc.

Exemple d'affichage d'une ressource ID (n'oubliez pas de lancer la commande 'SET SERVEROUTPUT ON' pour voir le tracé) :

```
DECLARE
  resid DBMS_XDB_VERSION.RESID_TYPE;
BEGIN

  -- 'Check-out' pour mise à jour du contenu du VCR
  DBMS_XDB_VERSION.Checkout('/home/digora/exemple.html');

  -- Utilisez resource_view pour mise à jour de digora.xml
  UPDATE resource_view
  SET res=sys.xmltype.createxml(
    '<Resource
      xmlns="http://xmlns.oracle.com/xdb/XDBResource.xsd"
      xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
      xsi:schemaLocation="http://xmlns.oracle.com/xdb/XDBResource.xsd
        http://xmlns.oracle.com/xdb/XDBResource.xsd">
      <Author>Jane Doe</Author>
      <DisplayName>versample</DisplayName>
      <Comment>Voila la mise a jour ?? </Comment>
      <Language>fr</Language>
      <CharacterSet>ASCII</CharacterSet>
      <ContentType>text/plain</ContentType>
    </Resource>')
  WHERE any_path = '/home/digora/exemple.html';

  -- 'Check-in' de la mise à jour
  resid := DBMS_XDB_VERSION.Checkin('/home/digora/exemple.html ');

  dbms_output.put_line('Resid est : '||resid);

  -- La dernière mise à jour peut être obtenue par resid et ces prédécesseurs
  -- peut être obtenue en utilisant les fonctions getPredecessor()
  -- ou getPredsbyResId().

END;
/
```

Utilisation des versions

L'accès aux ressources versionnées se fait comme l'accès à n'importe quelle ressource de XML DB, par contre les opérations de mise à jour diffèrent légèrement. Le travail sur ces ressources se fait par des extractions/réinsertions (checkin/checkout). Ce mécanisme est classique dans les systèmes avec gestion de version : lorsque vous voulez travailler sur le document vous « l'extrayez » du système, il est alors en lecture seule pour les autres utilisateurs. Une fois votre travail terminé, vous « réinsérez » le document dans le système où il est enregistré sous une nouvelle version et redevient modifiable pour tout le monde.

Exemple d'extraction/réinsertion d'un document :

```
DBMS_XMLDB_VERSION.CheckOut('/home/digora/exemple.html');  
DBMS_XMLDB_VERSION.CheckIn('/home/digora/exemple.html');
```

Sécurité des versions

La sécurité est gérée par des listes d'accès et chaque ancienne version d'un document a la même liste d'accès que sa version courante. La gestion de la sécurité dans XML DB sera abordée plus loin dans cet ouvrage.

API PL/SQL de gestion des ressources XML DB

Cette API stockée dans le package DBMS_XMLDB permet de gérer le référentiel XML DB et ses ressources à l'aide du langage PL/SQL.

Elle permet les opérations suivantes :

- la gestion des ressources du référentiel ;
- la gestion de la sécurité et les listes d'accès ;
- la gestion de la configuration du référentiel ;
- la régénération des index du référentiel.

Les méthodes createResource() et createFolder() permettent de créer les ressources et les dossiers qui pourront être manipulés avec les autres méthodes.

Manipulation des ressources

Création de dossiers et de ressources :

```
declare  
    retour boolean;  
begin  
    retour := dbms_xmldb.createfolder('/public/digora');  
    retour := dbms_xmldb.createresource('/public/digora/digora.xml', '<Nom>Digora</Nom>');  
    retour := dbms_xmldb.createresource('/public/digora/lebeau.xml', '<Nom>Lebeau</Nom>');  
    commit;  
end;  
/
```

Création de liens :

```
call dbms_xdb.link('/public/digora/digora.xml', '/public/digora', 'lien_digora.xml');
call dbms_xdb.link('/public/digora/lebeau.xml', '/public/digora', 'lien_lebeau.xml');
commit;
```

Suppression de ressources :

```
call dbms_xdb.deleteresource('/public/digora/digora.xml');
call dbms_xdb.deleteresource('/public/digora/lien_digora.xml');
call dbms_xdb.deleteresource('/public/digora/lebeau.xml');
call dbms_xdb.deleteresource('/public/digora/lien_lebeau.xml');
commit;
```

Gestion des paramètres XML DB

Récupération de la configuration du référentiel XML DB :

```
select dbms_xdb.cfg_get() from dual;
```

Mise à jour de la configuration XML DB :

```
declare
  config xmltype;
begin
  config := dbms_xdb.cfg_get();
  -- Utilisation de updatexml() pour modifier
  -- la configuration
  dbms_xdb.cfg_update(config);
end;
/
```

Régénération des index du référentiel

Exemple de suppression et reconstruction d'index :

```
connect system/manager
delete from xdb.xdb$h_index;
commit;
execute dbms_xdb.RebuildHierarchicalIndex;
```

API Java/JNDI de gestion des ressources XML DB

Cette API est l'équivalent de DBMS_XDB mais en Java, elle est conçue pour être utilisée avec Java et JNDI. JNDI est l'API de Sun utilisée pour interroger les services de répertoire, comme DNS, LDAP, NDS, etc. Dans XML DB, JNDI est utilisée pour localiser les ressources et renvoie des objets XMLType qui supportent à la fois les interfaces DOM et JavaBean. Une fois récupérée, l'instance XMLType peut être transtypée en instance d'interface DOM ou en instance de JavaBean. Alternativement, il est possible de manipuler l'objet XMLType directement en SQL ou d'en mettre à jour des parties avec la fonction SQL UPDATE_VAL().

Pour localiser une ressource, JNDI propose une méthode `lookup()` qui retourne :

- une instance du `JavaBean` si le résultat de la recherche est basé sur un schéma XML enregistré ou un document `XMLDocument` contenant les données ;
- une instance de l'interface `XML DB ressource`.

La localisation d'une ressource via JDBC se fait par une interrogation de la vue `RESSOURCE_VIEW`.

Une fois la ressource localisée et récupérée, il est possible d'utiliser l'API `XMLType` pour Java afin de la manipuler.

Utilisation de JNDI

JNDI est performante mais ne peut effectuer des recherches que selon un chemin XPath.

Exemple d'interrogation d'un fichier `exemple.xml` avec DOM et JNDI pour récupérer les valeurs du nom et de l'âge d'une personne :

```
Context ctx = new InitialContext(env);
Document po = (Document)ctx.lookup("/exemples/exemple.xml");
Node n;
n = po.getElementsByTagName("Nom").item(1);
String titre = n.getNodeValue();
n = po.getElementsByTagName("Age").item(1);
int age = new Boolean(n.getNodeValue()).booleanValue();
```

Exemple d'interrogation d'un fichier `exemple.xml` avec les `JavaBean` pour récupérer les valeurs du nom et de l'âge d'une personne :

```
Context contexte = new InitialContext(env);
Exemple exemple = contexte.lookup("/exemples/exemple.xml");
String nom = exemple.getNom();
int age = exemple.getAge();
```

Utilisation de JDBC

Pour localiser une ressource, JDBC est moins performant que JNDI mais il peut utiliser des ordres `SELECT` pour rechercher une ressource autrement que par son chemin.

Exemple d'interrogation d'un fichier XML avec JDBC pour récupérer un document DOM :

```
PreparedStatement pstmt = con.prepareStatement(
    "SELECT r.RESOLVE_PATH('/exemples/exemple.xml') FROM XDB$RESOURCE r");
pstmt.executeQuery();
XMLType donneesXML = (XMLType)pst.getObject(1);
Document leDocument = (Document) donneesXML.getDOM();
```

Sécurité des ressources XML DB

Dans le référentiel XML DB, la sécurité est gérée au niveau de l'élément. Cela signifie que les droits peuvent être affectés individuellement à chaque ressource d'un dossier. Le contrôle se fait par des listes d'accès qui sont affectées à une ou plusieurs ressources, il s'agit là d'un mécanisme standard que l'on retrouve dans Windows, en Java, etc.

Le principal

Il s'agit d'une entité qui peut se voir attribuer des privilèges sur une ressource, notamment :

- un utilisateur de la base de données ;
- un rôle de base de données.

Les privilèges

Un privilège est un droit qui peut être attribué à un principal. Il peut être :

- un agrégat : un privilège constitué de plusieurs autres privilèges ;
- atomique : un privilège constitué d'un droit unique.

Les privilèges sous forme d'agrégats simplifient la gestion des droits lorsqu'ils deviennent nombreux à mettre en place.

Élément de liste d'accès (ACE : Access Control Entry)

Une entrée de liste d'accès est un sous-élément de liste d'accès constitué d'une opération (autorisation, refus), d'un principal (utilisateur, groupe) et d'un privilège qui doit être autorisé/refusé à l'utilisateur/groupe défini.

Listes d'accès (ACL : Access Control List)

Une liste d'accès est un ensemble d'éléments de listes d'accès. Leur convention de nommage est la suivante : `privilege_users_acl.xml`.

Listes d'accès nommées (Named ACL)

Une liste d'accès nommée est une liste d'accès qui est elle-même une ressource. Ce faisant, elle peut être partagée entre plusieurs autres ressources, ce qui augmente la flexibilité. Ces listes ont un nom unique et peuvent être restreintes à certaines instances de documents XML.

Listes d'accès par défaut (Default ACLs)

Lors de l'ajout d'un document au référentiel XML DB, il y a deux possibilités d'affectation d'une liste d'accès par défaut :

- utiliser celle par défaut du dossier parent ;
- spécifier une liste particulière.

Bootstrap ACL

Cette liste d'accès est différente des autres en ce sens qu'elle est protégée par son propre contenu (lecture seule) et qu'elle s'applique à toutes les autres listes d'accès qui sont créées. En effet, il s'agit de la liste d'accès qui va placer toutes les autres en lecture seule pour tout le monde sauf pour les groupes XDBADMIN et DBA. Elle se trouve dans `/sys/acls/bootstrap_acl.xml`.

Utilisation des listes d'accès

Exemple de liste d'accès :

```
<acl description="monACL"
xmlns="http://xmlns.digora.com/acl.xsd"
xmlns:dav="DAV:"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:schemaLocation="http://xmlns.digora.com/acl.xsd
http://xmlns.digora.com/acl.xsd">
  <ace>
    <principal>OWNER</principal>
    <grant>true</grant>
    <privilege><all/></privilege>
  </ace>
</acl>
```

Mise en place de la liste d'accès ALL_OWNER_ACL :

```
call dbms_xdb.setacl('/public/digora/digora.xml','/sys/acls/all_owner_acl.xml');
commit;
```

Récupération de la liste d'accès affectée à un document :

```
select dbms_xdb.getacldocument('/public/digora/digora.xml') from
dual;
```

Création d'une liste d'accès et application de la liste à une ressource :

[illegible]

```

                                DAV:http://xmlns.oracle.com/xdb/dav.xsd">
    <principal>XDB</principal>
    <grant>true</grant>
    <privilege>
      <all/>
    </privilege>
  </ace>';
  ace := xmltype.createxml(ace_data);
  r := dbms_xdb.changeprivileges('/public/digora/digora.xml', ace);
  dbms_output.put_line('retval = ' || r);
  commit;
end;
/

```

Accès au référentiel XML DB par, FTP, HTTP et WebDAV

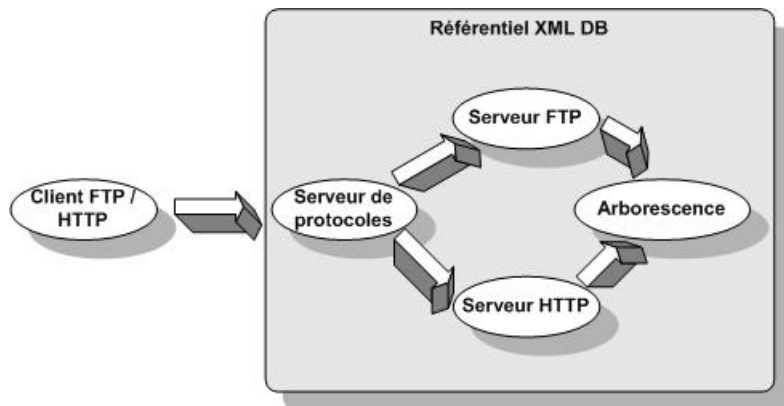
Oracle XML DB Protocol Server

Le serveur de protocoles supporte les protocoles Internet standard pour permettre l'accès à l'arborescence XML DB. Cet accès direct se fait facilement, sans installation de produits supplémentaires sur les postes client.

Pour optimiser les performances, le serveur de protocoles utilise un pool de sessions et le protocole HTTP 1.1.

Figure 21-1

Serveur de protocoles
XML DB



Configuration du serveur de protocoles

Le serveur de protocoles se configure par un fichier stocké dans le dossier du référentiel /xdbconfig.xml. Les paramètres communs se trouvent dans le dossier /xdbconfig/sysconfig/protocolconfig/common, ceux du serveur ftp se trouvent dans /xdbconfig/sysconfig/protocol-

config/ftpconfig et ceux du serveur http dans /xdbconfig/sysconfig/protocolconfig/http-config.

Les paramètres de ces fichiers de configuration sont des plus classiques, on y retrouve notamment les directives suivantes :

Serveur FTP :

- ftp-port ;
- ftp-protocol ;
- session-timeout.

Serveur HTTP :

- http-port ;
- http-protocol ;
- session-timeout ;
- server-name ;
- max-header-size ;
- max-request-body ;
- webappconfig/welcome-file-list.

Accès par FTP

Le protocole FTP est implémenté de façon standard, suivant la RFC959, mis à part les fonctions optionnelles suivantes :

- Append ;
- Allocate ;
- Account ;
- Abort.

L'accès se fait de façon classique, avec un client FTP qui se connecte au port 2100 par défaut défini dans le fichier xdbconfig.xml.

Accès par HTTP

Le protocole HTTP est implémenté également de façon standard, suivant la RFC2616, et supporte l'extension que sont les « cookies », décrits dans la RFC2109. Seules les fonctionnalités suivantes ne sont pas présentes :

- la compression gzip ;

- l'authentification Digest ;
- la fonction Trace ;
- les directives de mise en cache ;
- le format de log Web commun ;
- la gestion multi-home.

L'accès se fait par le port non standard 8080.

Accès par WebDAV (Web Distributed Authoring and Versioning)

WebDAV est un protocole standard qui permet à l'utilisateur de visualiser une arborescence à travers Internet. Il s'agit en fait d'une extension du protocole HTTP 1.1 dont l'implémentation dans XML DB est conforme à la RFC2518. L'accès à l'arborescence via un navigateur Web est donc possible, le plus commun étant la fonctionnalité Dossiers Internet de Windows 2000. Les fonctionnalités suivantes ne sont pas supportées :

- les listes d'accès WebDAV ;
- le verrouillage sans limitation de profondeur hiérarchique.

Pour créer un nouveau Dossier Internet dans Internet Explorer, suivez la procédure ci-dessous :

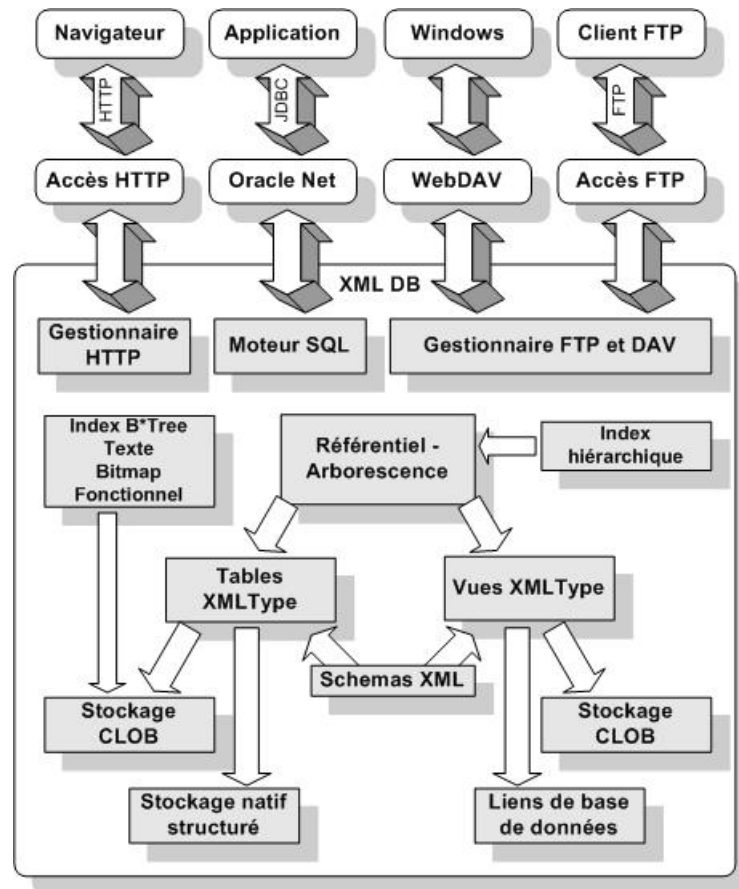
- ouvrez les Favoris réseau ;
- double-cliquez sur Ajout d'un favori réseau ;
- saisissez l'adresse du serveur : `http://[serveur]:<port HTTP>` ;
- cliquez sur Suivant ;
- saisissez un nom pour ce favori réseau ;
- cliquez sur Terminer.

Vous pouvez désormais accéder au référentiel XML DB comme n'importe quel disque Windows.

Schéma global d'architecture XML DB

Figure 21-2

Architecture XML DB



Stockage XML traditionnel dans Oracle 10g

Oracle 10g prend en charge les normes Internet, Java et XML. Il est ainsi possible d'exécuter une application Oracle XML en utilisant Oracle JServer (Oracle 10g intègre la machine virtuelle Java).

C'est dans cette logique d'intégration d'XML au sein de la base que l'on retrouve Oracle Text, pour le traitement de documents XML. En effet, Oracle Text est capable d'indexer et d'effectuer des recherches poussées sur les documents XML.

Avec les types objet d'Oracle 10g, on pourra récupérer la structure d'une donnée XML dans un objet. Cette possibilité permet de mieux gérer les données XML en conférant une plus grande lisibilité et une meilleure conception des documents XML.

Dans Oracle 10g, il y a deux catégories possibles de stockage de code XML. On peut, selon la structure du document XML :

- le stocker en tant que document unique ;
- le stocker en tant que données.

Stocker un document XML en tant que document unique

Ce type de stockage est intéressant si le contenu du document XML est statique ou que sa mise à jour s'effectue uniquement par un remplacement complet du document. Souvent, il s'agit d'un article de presse, de publicités, de contrats... Ce mode de stockage vous laisse encore le choix de stocker les documents XML dans la base de données, dans un champ de type CLOB (Character Large Object), dans un champ de type XMLType, dans une table de type XMLType, ou en dehors et de retrouver facilement ces documents via BFILES.

Stockage d'un document XML dans une colonne CLOB

Création de la table contenant le document :

```
CREATE TABLE docs_xml
(
  id NUMBER PRIMARY KEY,
  ajout_par VARCHAR2(50),
  docs CLOB);
```

L'insertion des documents XML peut se faire de différentes façons. La première est l'insertion de manière directe d'un texte dans un CLOB :

```
insert into docs_xml values(2,'More1',
'<FILM>
  Le film du mois
  <TITRE LANG="fr">Il faut sauver le soldat Ryan</TITRE>
  <TITRE LANG="us">Saving private Ryan</TITRE>
  <ACTEURS>
    <ACTEUR NATIONALITE="us">Tom Hanks</ACTEUR>
  </ACTEURS>
  <SUPPORT>DVD</SUPPORT>
</FILM>');
```

Le deuxième mode d'insertion est celui utilisant SQL*Loader. On l'utilise en général pour le chargement de base à partir d'un fichier texte préformaté. Dans notre cas, il est nécessaire de créer un fichier de contrôle contenant les informations sur le formatage des données, et sur le lieu où stocker ces données. Exemple de fichier de contrôle Load.ctl :

```
LOAD DATA
INFILE *
INTO TABLE docs_xml
REPLACE
FIELDS TERMINATED BY ','
```

```
( ID          SEQUENCE(MAX,1),
  AJOUT_PAR CHAR,
  FICHER FILLER CHAR,
  DOCS LOBFILE (FICHER) TERMINATED BY EOF NULLIF FICHER = 'NONE'
)
BEGINDATA
Fred,ews.xml
Fred,spr.xml
```

Ce fichier de contrôle indique que les données seront chargées dans la table docs_xml. Chaque ligne de données sources doit contenir une chaîne de caractères qui sera la valeur du champ AJOUT_PAR et le nom d'un fichier à charger en tant que LOB dans le champ DOCS de la table. Ces deux champs doivent être séparés par une virgule. Enfin, la colonne ID de la table est remplie automatiquement par une séquence. Dans cet exemple, les données chargées sont placées à la fin du fichier de contrôle après BEGINDATA mais elles pourraient aussi être dans un fichier séparé, référencé par INFILE nom_fichier. Le chargement des données se fait par la commande :

```
sqlldr userid=scott/tiger control=load.ctl
```

Enfin, il reste la possibilité d'insérer des documents à l'aide de fonctions (ou procédures) développées en Java, C++ ou PL/SQL (avec par exemple le package DBMS_LOB).

Une fois nos documents stockés, il faut pouvoir y faire des recherches. C'est ici qu'intervient Oracle Text. Cependant, pour pouvoir faire des recherches, Oracle Text nécessite la création d'un index sur la colonne contenant les documents XML :

```
CREATE INDEX docs_xml_idx ON docs_xml(docs)
INDEXTYPE IS CTXSYS.CONTEXT
PARAMETERS ('DATASTORE CTXSYS.DIRECT_DATASTORE SECTION GROUP
➔CTXSYS.PATH_SECTION_GROUP');
```

Ce code permet de créer un index appelé docs_xml_idx sur le champ DOCS de la table docs_xml. L'index est de type CONTEXT (appartenant à l'utilisateur ctxsys) et il utilise DIRECT_DATASTORE comme paramètre de stockage ainsi que PATH_SECTION_GROUP comme paramètre de filtrage. Cela signifie que Oracle Text va créer automatiquement la section pour chaque nouvelle balise du document XML et permettre la recherche en utilisant les chemins.

Par exemple, si on a dans un document XML :

```
<doc titre="conte de fée" >Il était une fois ... </doc>
```

On peut créer un groupe de section spécifique, permettant de rechercher l'attribut titre de la section doc :

```
exec CTX_DDL.create_SECTION_GROUP('groupe', 'XML_SECTION_GROUP');

exec ctx_ddl.add_attr_section('groupe','titre','doc@titre');
```

Mais, bien que tous les utilisateurs aient le droit de créer des index, il est nécessaire qu'ils aient le rôle CTXAPP pour utiliser ces packages PL/SQL d'Oracle Text.

On obtient alors le paramétrage suivant pour l'index utilisant le groupe de section précédente :

```
CREATE INDEX docs_xml_idx ON docs_xml(docs)
INDEXTYPE IS CTXSYS.CONTEXT
PARAMETERS ('DATASTORE CTXSYS.DIRECT_DATASTORE SECTION GROUP groupe');
```

Cependant, dans le cas d'ajout ou de suppression de nouveaux documents XML, il faut demander une modification de l'index par :

```
ALTER INDEX docs_xml_idx REBUILD ONLINE PARAMETERS('SYNC');
```

Ce code permet de synchroniser l'index ONLINE (de manière non bloquante). Mais on peut utiliser les paramètres OPTIMIZE FULL ou OPTIMIZE FAST pour optimiser l'index.

Pour éviter d'avoir à répéter souvent cette commande, on peut recréer l'index à chaque modification en utilisant l'option « sync (On Commit) » lors de la création de l'index.

Stockage non structuré d'un document dans une colonne XMLType

Oracle9i introduisait un nouveau type de données, XMLType, pour faciliter la gestion native de documents XML dans la base de données.

Oracle 10g complète les possibilités de ce type, en permettant notamment son utilisation côté client ainsi qu'en lui ajoutant le support de XML Schéma.

Voici un aperçu des propriétés que possède ce type :

- il peut être utilisé comme type de colonne dans une table ou une vue ;
- en PL/SQL, il peut être paramètre, variable ou valeur de retour ;
- en tant qu'instance d'objet, il peut représenter un document XML ;
- il possède des méthodes natives de traitement pour extraire, créer et indexer des données XML stockées dans Oracle 10g ;
- ces méthodes sont accessibles via des API en PL/SQL et en Java.

Lorsqu'une colonne est de type XML Type et qu'aucune spécification XML Schéma n'y est associée, le stockage est alors assuré par un champ CLOB sous-jacent. C'est le stockage déstructuré. Dans le cas contraire, si une spécification XML Schéma est associée au document, ce dernier sera traité après son insertion. Il va être décomposé et stocké dans un ensemble de tables ou vues objet relationnelles lui correspondant. Il s'agit du stockage structuré.

Le stockage structuré conserve des informations comme :

- l'ordre de leurs éléments et leurs attributs ;
- la distinction entre les éléments et leurs attributs ;
- les types de données XML basiques (boolean, QName...) ;
- les contraintes XML (*facets*) comme les listes numérotées.

Il est possible de passer du stockage XML Type natif et structuré au stockage LOB déstructuré en utilisant les imports et les exports de la base de données.

Syntaxe de l'API XML Type :

```
CREATE TABLE [schema.] table OF XMLTYPE
[XMLTYPE XMLType_storage] [XMLSchema_spec];
```

Exemple de création d'une table avec une colonne XMLType :

```
CREATE TABLE docs_xml(
  id NUMBER(3),
  nom VARCHAR2(35),
  contenu XMLTYPE);
```

et d'insertion de données dans cette table :

```
INSERT INTO docs_xml(id,nom,contenu) VALUES
(1,'Gino',XMLType(
'<Employe whNo="100">
<Nom>Louis</Nom>
</Employe >'));

```

Dans cet exemple, nous avons créé une instance de l'objet XML Type en utilisant son constructeur XMLType().

Interrogeons cette table pour récupérer la valeur de la colonne XMLType :

```
SELECT
dx.contenu.extract('/Employe/Nom/text()').getStringVal() "Nom"
FROM docs_xml dx;
Nom
-----
Louis
```

Mise à jour d'une colonne XML Type à l'aide d'une instruction UPDATE :

```
UPDATE docs_xml SET contenu = XMLType
('<Employe whNo="100">
<Nom>Morel</Nom>
</Employe>');
```

Mise à jour d'une colonne XML Type à l'aide d'une instruction UPDATEXML :

```
UPDATE docs_xml
SET contenu = UPDATEXML(contenu,
'/Employe/Nom/text()', 'Morel');
```

Il est également possible de créer une table de type XML Type, où le stockage et déstructuré par défaut :

```
CREATE TABLE po_xtab OF XMLType;
```

Stockage d'un document XML en dehors de la base

Dans ce cas, seul le chemin d'accès au document est enregistré dans la base. Création de la table référençant le document XML :

```
CREATE TABLE docs_xml  
(  
  id NUMBER PRIMARY KEY,  
  ajout_par VARCHAR2(50),  
  docs VARCHAR2(200));
```

Comme pour le stockage direct en base, il est nécessaire de créer un index pour pouvoir faire des recherches grâce à Oracle Text :

```
CREATE INDEX docs_xml_idx ON docs_xml(docs)  
INDEXTYPE IS CTXSYS.CONTEXT  
PARAMETERS ('FILTER CTXSYS.NULL_FILTER DATASTORE CTXSYS.FILE_DATASTORE SECTION GROUP  
➔CTXSYS.PATH_SECTION_GROUP');
```

Comme précédemment, il est nécessaire de tenir l'index à jour.

Stocker un document XML en tant que données

Cette situation survient si la structure du document XML est bien formée et que les données contenues sont susceptibles d'être mises à jour ou exploitées par des applications compatibles avec XML. En général, un document XML contient des éléments dont la structure est plutôt complexe (commandes, factures...). Oracle 10g permet, grâce au type objet, de récupérer la structure des données dans la base en utilisant les types d'objets, les références d'objets et les collections. Il existe deux solutions pour stocker et préserver la structure des documents XML dans un modèle relationnel objet :

- stocker les attributs des éléments dans des tables relationnelles et définir des vues objet pour conserver les structures des éléments XML ;
- stocker les éléments XML dans une table de type objet.

Les données stockées dans des types objet peuvent être mises à jour, reformatées ou participer à des requêtes SQL.

Oracle fournit l'utilitaire XML SQL Utility pour stocker des documents XML sous forme d'objets relationnels et, inversement, traduire des objets relationnels sous forme de document XML.

Si la structure d'un document XML n'est pas compatible avec la structure du schéma de la base de données, il faut transformer les données dans un format correct avant de les insérer dans la base. Il est possible de réaliser cette transformation avec les feuilles de

style XSL ou des procédures spécifiques. Il sera parfois nécessaire de conserver le document XML intact, sans modification. Sinon, il est possible de définir des vues objet correspondant à la structure du document XML et d'utiliser les triggers instead-of pour réaliser les opérations de transformation adéquates en vue d'une mise à jour des données de la base.

Elles permettent de construire rapidement des objets en combinant les données stockées XML de différentes manières et de conserver ainsi :

- les données structurées (clients, commandes...) dans un type d'objet relationnel : Exemple de documents à sauvegarder ;
- les données non structurées (description, définition...) dans un champ de type CLOB.

Stocker un document XML dans des tables objets

Prenons le cas d'un document XML contenant les coordonnées d'un client :

```
<ROWSET>
<ROW num="1">
  <CLIENT>
    <CLIENTID>1044</CLIENTID>
    <PRENOM>Paul</PRENOM>
    <NOM>Astoria</NOM>
    <ADRESSE>
      <RUE>123 Cherry Lane</RUE>
      <CODE_POSTAL>94132</CODE_POSTAL>
      <VILLE>SF</VILLE>
      <PAYS>CA</PAYS>
    </ADRESSE>
  </CLIENT>
</ROW>
</ROWSET>
```

Ce document nécessite deux objets :

- un objet adresse contenant la rue, le code postal, la ville et le pays du client ;
- un objet client contenant l'identifiant, le prénom, le nom et l'adresse du client.

On en déduit donc les objets à implémenter :

```
create type type_adresse as object
(
  rue varchar2(40),
  ville varchar2(20),
  pays varchar2(10),
  code_postal varchar2(10)
);
/
create type type_client as object
(
  clientid number(10),
```

```
prenom varchar2(20),  
nom varchar2(20),  
adresse type_adresse  
);  
/
```

On peut ensuite créer la table correspondant au client :

```
create table client_tab ( client type_client);
```

Stocker un document XML à l'aide des vues

Dans ce cas, chaque donnée du client est stockée dans un champ de la table client :

```
create table client  
( clientid number(10),  
  prenom varchar2(20),  
  nom varchar2(20),  
  rue varchar2(40),  
  code_postal varchar2(10),  
  ville varchar2(20),  
  pays varchar2(10)  
);
```

Ensuite, il faut créer les objets du document XML (voir précédemment), puis une vue qui donne la structure du document XML :

```
create view vue_client OF XMLType WITH OBJECT ID  
  (EXTRACT(OBJECT_VALUE, '/client/clientid').getnumberval())  
as  
select SYS_XMLGEN(type_client(clientid, prenom, nom,  
  type_adresse(rue, ville, pays, code_postal)))  
from client;
```

L'outil Oracle XML SQL Utility permet ensuite de récupérer les données des vues pour reformer le document XML.

Stocker un document XML de façon structurée avec XML Type et XML Schéma

Oracle 10g XML DB utilise des schémas XML pour paramétrer la façon dont seront traités les documents XML qui y seront stockés. Le support de XML Schéma est étendu aux fonctionnalités suivantes :

- la validation W3C des schémas XML ;
- la validation de documents XML par rapport à des schémas XML enregistrés ;
- la génération de schémas XML à partir de types d'objets ;
- le référencement d'un schéma XML d'un autre utilisateur ;
- la génération de structures de mapping dans la base de données à partir d'un schéma XML (incluant les types, les collections et les tables) ;

- la création de tables, vues ou colonnes XML Type à partir de schémas XML enregistrés ;
- la manipulation et l'interrogation de tables XML Type basées sur un schéma XML ;
- le stockage automatique des documents dans les tables de mapping s'ils correspondent à un schéma XML enregistré.

C'est DBMS_XMLSCHEMA, un package PL/SQL livré uniquement dans la release 2 d'Oracle 10g, qui va servir à la gestion des schémas XML. Avant toute référence, un schéma XML doit être référencé dans XML DB : c'est la méthode DBMS_XMLSCHEMA.registerSchema() qui s'en charge. Elle prend deux paramètres :

- un document XML source, sous forme de VARCHAR2, CLOB, XML Type ou URI Type ;
- une URL XML Schéma : elle correspond à une URL qui permettra aux documents XML d'indiquer à quel schéma ils se conforment.

Une fois l'enregistrement effectué :

- tous les documents se conformant au schéma XML enregistré et spécifiant l'URL de ce schéma seront automatiquement traités par XML DB ;
- des tables et des colonnes faisant référence au schéma XML peuvent être créées pour y stocker le document traité.

Stockage dans une table avec une colonne XML Type

Voyons un exemple d'enregistrement d'un schéma XML utilisant des types complexes :

```
declare
document varchar2(1000) := '<schema
targetNamespace="http://www.digora.com/typecommande.xsd"
xmlns:cmd="http://www.digora.com/typecommande.xsd"
xmlns="http://www.w3.org/2001/XMLSchema">
<complexType name="TypeCommande">
  <sequence>
    <element name="Numero" type="decimal"/>
    <element name="Société">
      <simpleType>
        <restriction base="string">
          <maxLength value="100"/>
        </restriction>
      </simpleType>
    </element>
    <element name="LigneCommande" maxOccurs="1000">
      <complexType>
        <sequence>
```

```

        <element name="Objet">
            <simpleType>
                <restriction base="string">
                    <maxLength value="1000"/>
                </restriction>
            </simpleType>
        </element>
        <element name="Prix" type="float"/>
    </sequence>
</complexType>
</element>
</sequence>
</complexType>
<element name="Commande" type="cmd:TypeCommande"/>
</schema>;
begin
dbms_xmlschema.registerSchema('http://www.digora.com/typecommande.xsd', document);
end;

```

Création d'une table avec une colonne de type XML Type basée sur le schéma XML TypeCommande :

```

create table commande_tab(
    id number,
    contenu XMLType
)
xmltype column contenu
XMLSCHEMA "http://www.digora.com/typecommande.xsd" element "Commande";

```

Insertion de valeurs dans la table commande_tab :

```

insert into commande_tab values (1,
xmltype('<Commande xmlns="http://www.digora.com/typecommande.xsd"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:schemaLocation="http://www.digora.com/typecommande.xsd
http://www.digora.com/typecommande.xsd">
<Numero>1001</Numero>
<Societe>Oracle</Societe>
<LigneCommande>
<Objet>Livre1</Objet>
<Prix>40</Prix>
</LigneCommande>
<LigneCommande>
<Objet>Livre2</Objet>
<Prix>50</Prix>
</LigneCommande>
</Commande>')));

```

Stockage de documents dans une table de type XML Type

Création d'une table de type XML Type basée sur le schéma XML TypeCommande :

```
CREATE TABLE commande_tab2 OF XMLTYPE XMLSCHEMA "http://www.digora.com/
↳typecommande.xsd" ELEMENT "Commande";
```

Insertion de valeurs dans la table commande_tab2 :

```
insert into commande_tab2 values (
xmltype('<Commande xmlns="http://www.digora.com/typecommande.xsd"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:schemaLocation="http://www.digora.com/typecommande.xsd
http://www.digora.com/typecommande.xsd ">
<Numero>1001</Numero>
  <Societe>Oracle</Societe>
    <LigneCommande>
      <Objet>Livre1</Objet>
      <Prix>40</Prix>
    </LigneCommande>
    <LigneCommande>
      <Objet>Livre2</Objet>
      <Prix>50</Prix>
    </LigneCommande>
  </Commande>');$
```

Suppression de schémas XML

La suppression d'un schéma XML du référentiel peut s'accompagner de deux options :

- Force : suppression du schéma XML enregistré même s'il y a des dépendances ;
- Cascade : suppression de tous les objets associés au schéma XML, comme les types objet et les tables de stockage.

Suppression avec option Force et Cascade :

```
EXEC dbms_xmlschema.deleteSchema('http://www.digora.com/
↳typecommande.xsd',dbms_xmlschema.DELETE_CASCADE_FORCE);
```

Rechercher dans un document XML

On a trois possibilités de recherche dans un document XML suivant son mode de stockage :

- par Oracle Text ;
- en utilisant les fonctionnalités de XML Type ;
- par un requêtage simple pour les documents stockés en tant que données (mais s'il nécessite des champs de type CLOB, on revient au premier cas).

Recherche par Oracle Text classique avec l'index Context

Pour que la recherche avec Oracle Text soit possible, il faut avant tout créer un index sur la colonne XML Type :

```
DROP TABLE docs_xml ;

CREATE TABLE docs_xml(id NUMBER(3), nom VARCHAR2(35),
contenu XMLTYPE);

CREATE INDEX docs_xml_index ON
docs_xml(contenu) indextype is ctxsys.context;
```

Il est à noter que, dans la release 2 d'Oracle 10g, il n'est plus nécessaire de posséder les privilèges QUERY_REWRITE ou GLOBAL_QUERY_REWRITE pour créer des index sur des colonnes XML Type comme c'était le cas avec la version 9.0.1.

Une fois le stockage et l'indexation des documents XML réalisés, la recherche est on ne peut plus simple : elle se résume à une requête SQL avec une contrainte spécifique.

Prenons l'exemple des documents XML suivants stockés dans la base :

```
insert into docs_xml values
(1,'NEWHOOK',
'<FILM>
  Le film du jour
  <TITRE>Eyes wide shut</TITRE>
  <ACTEURS>
    <ACTEUR NATIONALITE="us">Tom Cruise</ACTEUR>
    <ACTEUR NATIONALITE="us">Nicole Kidman</ACTEUR>
  </ACTEURS>
  <SUPPORT>DVD</SUPPORT>
</FILM>' ) ;
```

et

```
insert into docs_xml values
(2,'NEWHOOK',
'<FILM>
  Le film du mois
  <TITRE LANG="fr">Il faut sauver le soldat Ryan</TITRE>
  <TITRE LANG="us">Saving private Ryan</TITRE>
  <ACTEURS>
    <ACTEUR NATIONALITE="us">Tom Hanks</ACTEUR>
  </ACTEURS>
  <SUPPORT>DVD</SUPPORT>
</FILM>');
```

N'oubliez pas de mettre à jour votre index :

```
alter index docs_xml_index REBUILD ONLINE PARAMETERS('SYNC');
```


La requête suivante nous retournerait l'identifiant id et le champ ajout_par des documents contenant le texte DVD entre les balises Support :

```
SELECT id, ajout_par FROM docs_xml  
WHERE CONTAINS(contenu, 'DVD WITHIN SUPPORT')>0;
```

Les fonctions WITHIN, CONTAINS et SCORE ont été complétées par de nouveaux opérateurs de recherche dans les documents XML : INPATH et HASPATH :

- INPATH permet de vérifier si une chaîne de caractères apparaît dans le chemin spécifié ;
- HASPATH permet de vérifier si le chemin XPath donné existe dans le document.

Mot-clé WITHIN

C'est l'un des opérateurs de base de la recherche dans un document XML. Il permet de rechercher un texte contenu dans des balises précises :

```
'DVD WITHIN SUPPORT'
```

Cette requête permet de tester si le texte DVD est contenu dans les balises Support.

Cependant, cet opérateur montre ses limites dans les opérations plus complexes comme pour les requêtes suivantes qui retournent toutes les deux les mêmes réponses :

```
SELECT id, nom FROM docs_xml  
WHERE CONTAINS(contenu, '(Ryan WITHIN TITRE) WITHIN FILM')>0;  
  
SELECT id, nom FROM docs_xml  
WHERE CONTAINS(contenu, '(Ryan WITHIN FILM) WITHIN TITRE')>0;
```

Donc, avec cet opérateur, il n'y a pas moyen de déterminer qui est inclus dans qui.

Mot-clé INPATH

C'est là qu'intervient l'opérateur INPATH. INPATH permet de faire des recherches plus précisément dans des documents XML en utilisant un moyen plus pointu : l'expression de chemin.

La syntaxe est : <texte recherché> INPATH(<expression de chemin>).

L'expression de chemin est basée sur le standard XPath mais n'est pas XPath car il y a quelques différences substantielles.

On utilise les exemples de documents XML décrivant des films pour les différents types de recherche.

La recherche au sein d'une balise de plus haut niveau :

```
'Le film du jour INPATH(FILM)'  
ou bien 'Le film du jour INPATH(/FILM)'  
ou encore 'Le film du jour INPATH(./FILM)'
```

Cette requête permet de chercher les documents où le texte « Le film du jour » est contenu entre les balises racines (et racines seulement) <FILM> et </FILM>.

La recherche simple au sein d'une balise :

```
'Tom Cruise INPATH(//ACTEUR)'  
Équivalent de 'Tom Cruise WITHIN ACTEUR'
```

Cette requête permet de chercher les documents où le texte « Tom Cruise » est contenu entre les balises ACTEUR et cela n'importe où dans le document.

La recherche Parent/Enfant :

```
'Nicole Kidman INPATH(//ACTEURS/ACTEUR)'
```

Cette requête permet de chercher les documents où le texte « Nicole Kidman » est contenu dans les balises ACTEUR, elles-mêmes enfants directes des balises ACTEURS.

La recherche à un niveau manquant :

```
'Nicole Kidman INPATH(FILM/*/ACTEUR)'
```

Cette requête permet de chercher les documents où le texte « Nicole Kidman » est contenu dans les balises ACTEUR, elles-mêmes petits-enfants directes des balises FILM. En fait, le symbole * remplace un et un seul élément.

La recherche de descendant :

```
'Nicole Kidman INPATH(FILM//ACTEUR)'
```

Cette requête permet de chercher les documents où le texte « Nicole Kidman » est contenu dans les balises ACTEUR, elles-mêmes descendantes, à quelque niveau que ce soit, des balises FILM.

La recherche suivant l'attribut :

```
'us INPATH(ACTEUR/@NATIONALITE)'
```

Cette requête permet de chercher les documents où le texte « us » est contenu dans l'attribut NATIONALITE d'une balise ACTEUR.

La recherche avec test d'existence, de descendant, d'attribut :

```
'valeur INPATH(A[expression])'
```

Cette requête permet de rechercher les documents où le texte « valeur » est contenu entre les balises A, qui elles-mêmes vérifient l'expression expression. Cette expression peut être un chemin (vérifie si ce chemin existe), un attribut (vérifie si cet attribut existe), un test sur une valeur d'attribut, un test sur une valeur de balise...

```
'valeur INPATH(A[B AND @C="val_attr"])'
```

Celle-ci permet de rechercher s'il existe une balise A contenant « valeur » et si elle a une balise fille directe B et un attribut C de valeur val_attr.

La recherche combinant les recherches précédentes :

```
'valeur INPATH(A//[B[@ATTR=5]/C/D])'
```

Cette requête permet de rechercher s'il existe une balise C contenant le texte « valeur » et qui est la fille directe d'une balise C, elle-même fille directe d'une balise B dont l'attribut ATTR a la valeur 5 et qui elle-même est apparentée à une balise A.

Mot-clé HASPATH

Contrairement à INPATH, HASPATH ne vérifie pas si la valeur à un chemin donnée existe mais si le chemin lui-même existe :

```
■ 'HASPETH(//SUPPORT)'
```

Cette requête vérifie s'il existe une ou plusieurs balises Support à n'importe quel niveau dans le document.

Il est aussi possible de faire un test sur la valeur de balise :

```
■ 'HASPETH(//SUPPORT = "DVD")'
```

Dans ce cas, on vérifie s'il existe une ou plusieurs balises *Support* contenant "DVD".

Recherche par Oracle Text avec le nouvel index CTXXPATH

Le nouveau type de stockage XML Type permet d'utiliser de nouvelles fonctions. En effet, XML Type met directement à disposition les opérateurs suivant :

- Extract : fonction prenant en paramètre une chaîne de caractères représentant une expression XPath et retournant un type XML Type. Elle renvoie la sous-partie pointée par l'expression XPath.
- ExistsNode : fonction qui, comme son nom l'indique, vérifie si un nœud existe. Elle prend une chaîne de caractères représentant une expression XPath et retourne un nombre.

Exemple :

```
■ SELECT doc.contenu.extract('/ACTEUR/text()').getstringval() AS acteur
FROM docs_xml doc
WHERE doc.contenu.existsnode('/FILM/ACTEURS') = 1 AND doc.contenu.extract('/FILM/
➔TITRE/text()').getstringval() LIKE 'Eyes wide shut';
```

Dans cet exemple, on choisit d'afficher la valeur du contenu des balises ACTEUR où qu'elles soient dans le document XML s'il existe un nœud ACTEURS fils de FILM et que la valeur de la balise TITRE fils de la balise FILM est Eyes wide shut.

Malheureusement, contrairement à l'opérateur CONTAINS, la fonction existsNode() ne tire pas profit des index Oracle Text. Pour pallier ce manque et optimiser la rapidité des recherches XPath avec existsNode(), Oracle 10g release 2 introduit un nouveau type d'index : CTXXPATH.

Cet index prend en charge la recherche, les caractères joker et les prédicats d'égalité de chaîne de caractères dans les requêtes XPath.

Création d'un index CTXXPATH :

```
■ CREATE INDEX docs_xml_xpath_index ON
Docs_xml(contenu) indextype is ctxsys.ctxxpath;
```

Exemple de recherche :

```
SELECT *
FROM docs_xml d
WHERE existsNode(d.contenu, '/Employe[@Nom="Louis"]') = 1;
```

Il faut privilégier `existsNode()` et `extract()` car leurs résultats sont plus significatifs que `CONTAINS`. En effet, Oracle Text ignore les espaces et certains attributs, aussi les résultats de recherche de cette dernière fonction sont moins précis.

Transformation et validation d'instances XML Type

Il existe deux outils qui permettent de transformer des données XML via les feuilles de style. Il est ainsi possible de changer la présentation d'un document XML en lui appliquant une feuille de style pour obtenir un document au format XML transformé : HTML, WML... Ces outils sont le moteur XSLT fourni avec Oracle XDK (XML Development Kit) et les méthodes XML Transform du type XML Type.

XSLT Processor

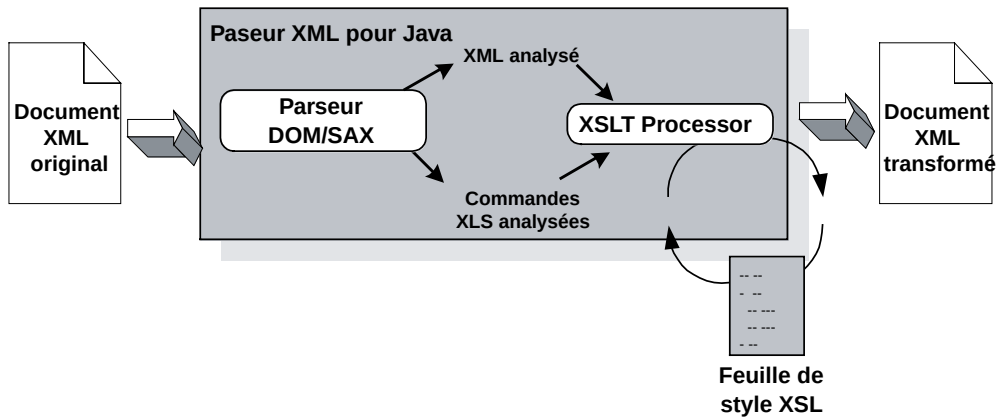


Figure 21-3

XSL Transformation Processor

Si on reprend le document contenant la liste des employés avec sa feuille de style définie précédemment :

```
<?xml version="1.0"?>
<?xml-stylesheet type="text/xsl" href="employees.xsl"?>
<empListe>
  <empDef>
    <empID>7369</empID>
```

```

    <empNom>SMITH</empNom>
    <empFonction>CLERK</empFonction>
    <empSal>800</empSal>
  </empDef>
  <empDef>
    <empID>7902</empID>
    <empNom>FORD</empNom>
    <empFonction>ANALYST</empFonction>
    <empSal>3000</empSal>
  </empDef>
</empListe>

```

On peut tester la transformation XSL par un programme Java fournit par Oracle :

```
java oracle.xml.parser.v2.oraxsl employees_xsl.xml employees.xsl
```

Ou bien développer un programme utilisant les API fourni par Oracle.

Dans tout les cas on obtient :

```

<html>
  <head>
    <title>Liste des employés</title>
  </head>
  <body>
    <table border="2" bordercolor="#404D6A">
      <tr bgcolor="#89C7A4">
        <td>empID</td>
        <td>empNom</td>
        <td>empFonction</td>
        <td>empSal</td>
      </tr>
      <tr>
        <td>7369</td>
        <td>SMITH</td>
        <td>CLERK</td>
        <td>800</td>
      </tr>
      <tr>
        <td>7902</td>
        <td>FORD</td>
        <td>ANALYST</td>
        <td>3000</td>
      </tr>
    </table>
  </body>
</html>

```

Transformation d'instance XML Type

En release 2, l'objet XML Type d'Oracle 10g permet d'effectuer des transformations XSLT décrites ci-dessus sans avoir à utiliser manuellement le XSLT processor.

Les méthodes XMLTransform() et XMLType.transform() sont identiques, mais la première s'applique au SQL et PL/SQL, alors que la seconde s'applique au langage Java.

Les paramètres sont les suivants :

- XML Type XMLdocument ;
- XML Type FeuilleDeStyle (la feuille de style est un document XML).

La méthode renvoie une instance XML Type correspondant à l'application de la feuille de style sur le document XML fourni en entrée.

Il est également possible d'utiliser l'API PL/SQL XSLT Processor. Cette manipulation sera décrite au chapitre dédié à l'API PL/SQL.

Validation d'instance XML Type

Avec la vérification du format du document XML, il est souvent utile de savoir s'il respecte un schéma XML défini. Par défaut, Oracle 10g vérifie si le document XML est bien formé et si le document correspond à un schéma XML enregistré. Oracle ajoute quelques tests de conformité succincts. Comme la validation XML conforme au standard W3C est lourde en traitement, quand les instances XML Type sont créées, stockées ou interrogées, les documents ne sont pas complètement vérifiés.

La validation complète d'un document XML doit être faite manuellement à l'aide des commandes suivantes :

- XMLIsValid() : il s'agit d'un opérateur SQL qui vérifie si l'instance XML Type fournie en entrée est conforme à un schéma XML donné. Le statut de validation de l'instance n'est pas modifié.
- schemaValidate() : cette procédure valide une instance XML Type par rapport à un schéma XML défini. Si c'est le cas, le statut de cette instance est positionné à VALIDATED.
- isSchemaValidated() : cette procédure vérifie si le statut de l'instance XML Type fournie est positionné sur VALIDATED ou non.
- setSchemaValidated() : cette procédure permet de spécifier manuellement l'état d'une instance XML Type.
- isSchemaValid() : est l'équivalent Java de l'opérateur SQL XMLIsValid().

Exemple de vérification de la validation de données XML stockées :

```
SELECT c.contenu.isSchemaValid('http://www.digora.com/commande.xsd','purchaseOrder')
FROM commande_tab c;
```

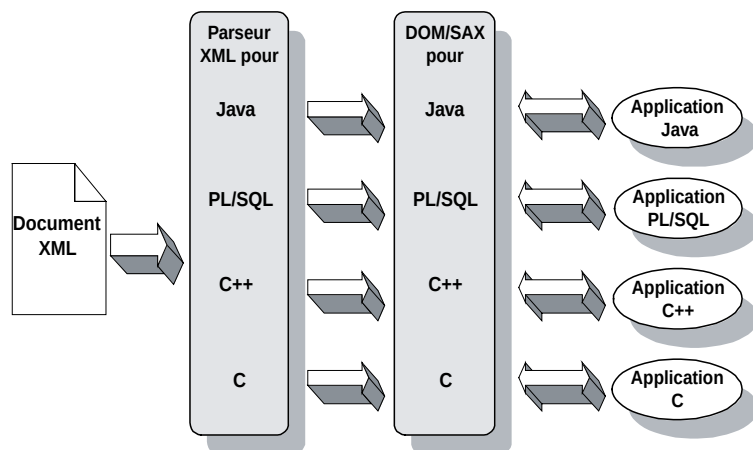
Exemple de contraintes pour forcer la validation avant le stockage :

```
CREATE TABLE maTable OF XMLType
(CHECK (XMLIsValid(sys_nc_rowinfo$) = 1))
XMLSchema "http://www.digora.com/typecommande.xsd" element "Commande";
```

Traitement XML : XML Parser v2 et API XML Type

Oracle fournit un ensemble d'analyseurs XML pour Java, C, C+ et PL/SQL. Chacun de ces analyseurs est un composant XML autonome, qui traite les documents XML (ou DTD ou XML Schéma) de telle sorte qu'ils puissent être utilisés par une application. L'analyseur est disponible sur toutes les plates-formes Oracle et prend en charge les interfaces DOM (Document Object Model) et SAX (Simple API for XML), hormis le parseur pour PL/SQL qui n'accepte que l'interface DOM. Pour compléter ces outils, le nouveau type XML Type d'Oracle 10g release 2 possède une API de gestion complète, développée à la fois en PL/SQL et en Java.

Figure 21-4
XML Parseur



Interfaces DOM et SAX

On distingue deux sortes d'interfaces, qui présentent chacune des avantages et des inconvénients. Il faut avant toute chose connaître les traitements qui vont être réalisés sur les documents. Il est possible d'opter, soit pour une approche événementielle, soit pour une structure arborescente. Le parseur d'Oracle peut utiliser deux API standard : SAX et DOM.

SAX (Simple API for XML) : interface à approche événementielle

La première API repose sur une approche événementielle, c'est-à-dire que l'analyseur lit séquentiellement le fichier de données en reconnaissant et en interprétant chaque balise. Les événements analysés sont reportés dans l'application par des fonctions de renvoi (*callbacks*). Puis, l'application implémente une fonction appelée handler pour chaque élément.

Voici l'exemple d'un handler qui écrit sur la sortie standard chaque fois qu'il rencontre une balise ouvrante ou fermante :

```
import org.XML.SAX.HandlerBase ;
import org.XML.SAX.AttributesList

public class Handler extends HandlerBase
{
    public void startElement (String nom, AttributeList attribut)
    {
        System.out.println("Ouverture : "+nom) ;
    }

    public void endElement (String nom)
    {
        System.out.println("Fermeture :"+nom);
    }
}
```

Un programme Java utilise ensuite ce handler afin de réaliser plusieurs traitements pour lancer ensuite l'analyse du document XML :

```
import org.XML.SAX.Parser ;
import org.XML.SAX.DocumentHandler ;
import org.XML.SAX.helpers.ParserFactory

public class Exemple
{
    static final String parserClass = "foo.bar.XML.SAXDriver" ;
    public static void main (String args [])
        throws Exception
    {
        // Création du parseur
        Parser parser = ParserFactory.makeParser(parserClass);

        // Création du handler
        DocumentHandler handler = new Handler();

        // Enregistrement du handler
        parser.setDocumentHandler(handler);
    }
}
```



```
// Analyse du document et de son URL
for (int i=0 ; i<args.length ; i++)
{
    parser.parse(args[i]);
}
}
```

Le document XML ci-dessous (enregistré sous le nom de `employees.xml`) :

```
<?xml version="1.0"?>
<empListe>
  <empDef>
    <empNom>SMITH</empNom>
    <empFonction>CLERK</empFonction>
  </empDef>
  <empDef>
    <empNom>FORD</empNom>
    <empFonction>ANALYST</empFonction>
  </empDef>
</empListe>
```

est donc transformé en la série d'événements ci-après, et ce grâce à l'appel de la commande Java Exemple file : chemin `.../employees.xml`

```
Ouverture : empListe
Ouverture : empDef
Ouverture : empNom
Fermeture : empNom
Ouverture : empFondtion
Fermeture : empFonction
Fermeture : empDef
Ouverture : empDef
Ouverture : empNom
Fermeture : empNom
Ouverture : empFondtion
Fermeture : empFonction
Fermeture : empDef
Fermeture : empListe
```

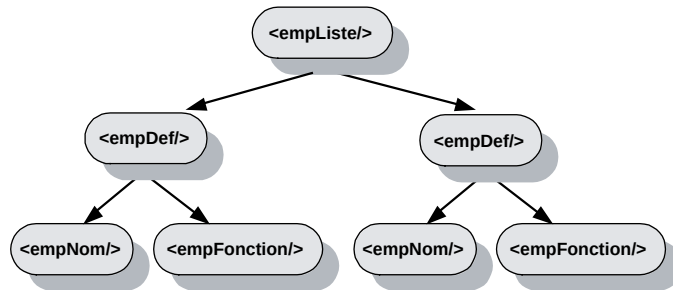
DOM (Modèle d'objet représentant un document) : interface de type arborescente

Ce deuxième type d'interface définit la structure logique d'un document et fournit les moyens d'accès aux données (les classes et méthodes permettant de naviguer dans l'arborescence). Le DOM est utile lors de manipulations structurales de l'arborescence XML, telles qu'ajouter, effacer ou modifier des éléments ainsi que leur contenu.

Le DOM crée l'arborescence d'après l'analyse du document XML. Ainsi, pour le document XML ci-dessous, trois niveaux représentant l'élément racine, les éléments et le groupe d'éléments vont être créés.

Figure 21-5

*Représentation
d'une hiérarchie
d'employés par un DOM*



Exemple : XML Parseur

Dans l'exemple suivant sont utilisés :

- XML Parseur pour Java v2, afin d'analyser un document XML ;
- l'interface DOM, pour manipuler les données XML.

Voici le code du fichier AnalyseDOM.java :

```
import java.io.*;
import java.net.*;
import org.w3c.dom.*;
import org.w3c.dom.Node;

import oracle.xml.parser.v2.DOMParser;

public class AnalyseDOM
{
    static public void main(String[] argv)
    {
        // Instancie un analyseur
        DOMParser analyseur = new DOMParser();

        try {
            // Transforme le nom du fichier XML en URL
            URL url = fichier2URL(argv[0]);

            // Analyse le fichier XML
            analyseur.parse(url);

            // Récupère le document
            Document doc = analyseur.getDocument();

            // Appelle la méthode qui affiche les éléments du document
```

```
        System.out.println("Liste des éléments du fichier "+argv[0]+" : ");
        System.out.println();
        listeElements(doc);
    }

    catch (Exception e)
    {
        System.out.println(e.toString());
    }
}

static void listeElements(Document doc)
{
    NodeList listeNœuds;
    Element elmnt;
    Node nœud;
    int nbreNœuds;

    // Récupère la liste de tous les nœuds descendants du nœud principal
    listeNœuds = doc.getElementsByTagName("*");

    // Renvoie le nombre de nœuds dans la collection listeNœuds
    nbreNœuds = listeNœuds.getLength();

    // Affiche le nom des nœuds de la collection listeNœuds
    for (int i=0; i < nbreNœuds; i++)
    {
        elmnt = (Element)listeNœuds.item(i);
        System.out.println("Nom de l'élément : "+elmnt.getTagName());
    }
}

// Méthode qui transforme un nom de fichier en URL
static URL fichier2URL(String nomFichier)
{
    URL url = null;
    try
    {
        url = new URL(nomFichier);
    }
    catch (MalformedURLException ex)
    {
        File f = new File(nomFichier);
        try
        {
            String path = f.getAbsolutePath();
            String fs = System.getProperty("file.separator");
            if (fs.length() == 1)
            {
                char sep = fs.charAt(0);
```

```

        if (sep != '/')
            path = path.replace(sep, '/');
        if (path.charAt(0) != '/')
            path = '/' + path;
    }
    path = "file://" + path;
    url = new URL(path);
}
catch (MalformedURLException e)
{
    System.out.println("Impossible de créer une URL");
    System.exit(0);
}
}
return url;
}
}

```

Appliquons ce programme au fichier XML `employees.xml` :

```

<?xml version="1.0" encoding="ISO-8859-1" standalone="no" ?>
<!DOCTYPE empListe SYSTEM "employees.dtd" >
<empListe>
  <empDef>
    <empID>7369</empID>
    <empNom>SMITH</empNom>
    <empFonction>CLERK</empFonction>
    <empSal>800</empSal>
  </empDef>
  <empDef>
    <empID>7902</empID>
    <empNom>FORD</empNom>
    <empFonction>ANALYST</empFonction>
    <empSal>3000</empSal>
  </empDef>
  <empDef>
    <empID>7499</empID>
    <empNom>ALLEN</empNom>
    <empFonction>SALESMAN</empFonction>
    <empSal>1600</empSal>
  </empDef>
</empListe>

```

Ce fichier XML est associé à la DTD `employees.dtd` :

```

<?xml version="1.0" encoding="ISO-8859-1" ?>
<!ELEMENT empListe (empDef*) >
<!ELEMENT empDef (empID,empNom,empFonction,empSal) >
<!ELEMENT empID (#PCDATA)>
<!ELEMENT empNom (#PCDATA)>
<!ELEMENT empFonction (#PCDATA)>
<!ELEMENT empSal (#PCDATA)>

```

Pour vérifier le comportement de ce programme, nous lançons :

```
■ java AnalyseDOM employes.xml
```

Nous obtenons un affichage de type :

Liste des éléments du fichier employes.xml :

```
Nom de l'élément : empListe
Nom de l'élément : empDef
Nom de l'élément : empID
Nom de l'élément : empNom
Nom de l'élément : empFonction
Nom de l'élément : empSal
Nom de l'élément : empDef
Nom de l'élément : empID
Nom de l'élément : empNom
Nom de l'élément : empFonction
Nom de l'élément : empSal
Nom de l'élément : empDef
Nom de l'élément : empID
Nom de l'élément : empNom
Nom de l'élément : empFonction
Nom de l'élément : empSal
```

Nouvelle API PLSQL de Manipulation XML

Les différentes API PLSQL disponibles pour manipuler les objets XML Type sont les suivantes :

- PL/SQL DOM API pour XML Type : DBMS_XMLDOM ;
- PL/SQL XML Parseur pour XML Type : DBMS_XMLPARSER ;
- PL/SQL XSLT Processor pour XML Type : DBMS_XSLPROCESSOR.

PL/SQL DOM API pour XMLType

Il s'agit d'une implémentation PLSQL de l'interface DOM. Elle permet de traiter les documents, éventuellement en les validant si une DTD est fournie.

La nouveauté apportée par cette API est le support XML Schéma.

Création d'un document DOM avec XML Type :

```
declare
    document dbms_xmldom.DOMDocument;
    buffer varchar2(32767);

begin
    document := dbms_xmldom.newDOMDocument(sys.xmltype('<personne><nom>Digora</nom>'
    <img alt="arrow icon" data-bbox="235 905 255 915"/></personne>'));
    dbms_xmldom.writeToBuffer(document, buffer);
```

```
        dbms_output.put_line(buffer);  
    end;  
    /
```

Création d'un nœud d'éléments à partir d'un document vide :

```
declare  
    document dbms_xmldom.DOMDocument;  
    element dbms_xmldom.DOMELEMENT;  
    noeud dbms_xmldom.DOMNode;  
  
begin  
    document := dbms_xmldom.newDOMDocument;  
    element := dbms_xmldom.createElement(document, 'MON_ELEMENT');  
    noeud := dbms_xmldom.makeNode(element);  
    dbms_output.put_line(dbms_xmldom.getNodeName(noeud));  
    dbms_output.put_line(dbms_xmldom.getNodeValue(noeud));  
    dbms_output.put_line(dbms_xmldom.getNodeType(noeud));  
end;  
    /
```

PLSQL XML Parseur pour XML Type

Le parseur XML est l'implémentation PL/SQL des mécanismes permettant l'extraction des valeurs et de la structure des documents. Il construit un arbre qui représente le document et qui peut être appelé depuis n'importe quel programme PL/SQL.

Traitement d'un document XML avec le XML Parseur PLSQL :

```
declare  
    document VARCHAR2(2000);  
    documentDom dbms_xmldom.domdocument;  
    noeud dbms_xmldom.domnode;  
    monParser dbms_xmlparser.Parser;  
  
begin  
    document := '<animal><race>Chien</race></animal>';  
    monParser := dbms_xmlparser.newParser;  
    dbms_xmlparser.parseBuffer(monParser, document);  
    documentDom := dbms_xmlparser.getDocument(monParser);  
    noeud := dbms_xmldom.makeNode(documentDom);  
end;  
    /
```

La dernière ligne de cet exemple permet d'utiliser l'API DOM sur le document.

PL/SQL XSLT Processor pour XML Type

Les méthodes du package DBMS_XSLPROCESSOR permettent d'appliquer des feuilles de style à des documents XML facilement. C'est une bonne alternative au pilotage manuel du XSLT Processor décrit au chapitre des transformations XML Type.

Exemple de transformation :

```

declare
    document VARCHAR2(2000);
    xsldocument VARCHAR2(2000);
    monParser dbms_xmlparser.Parser;
    documentDom dbms_xmldom.domdocument;
    documentDomXslt dbms_xmldom.domdocument;
    xsl dbms_xslprocessor.stylesheet;
    documentDomFragement dbms_xmldom.domdocumentfragment;
    noeud dbms_xmldom.domnode;
    processeur dbms_xslprocessor.processor;
    buffer varchar2(2000);

begin
    document := '<Employe><Numero>1</Numero> <Prenom>Jean-Charles
    ➡</Prenom><Nom>Razibus</Nom> <Salaire>1000</Salaire><Emploi>Consultant</Emploi>
    ➡</Employe>';
    xsldocument :=
        '<?xml version="1.0"?>
        <xsl:stylesheet version="1.0"
        xmlns:xsl="http://www.w3.org/1999/XSL/Transform">
        <xsl:output encoding="utf-8"/>
        <xsl:template match="">
        <xsl:copy>
        <xsl:apply-templates select="*|text()"/>
        <xsl:sort select="name(.)" data-type="text" order="ascending"/>
        </xsl:apply-templates>
        </xsl:copy>
        </xsl:template>
        <xsl:template match="text()">
        <xsl:value-of select="normalize-space(.)"/>
        </xsl:template>
        </xsl:stylesheet>';

    monParser := dbms_xmlparser.newParser;
    dbms_xmlparser.parseBuffer(monParser, document);
    documentDom := dbms_xmlparser.getDocument(monParser);
    dbms_xmlparser.parseBuffer(monParser, xsldocument);
    documentDomXslt := dbms_xmlparser.getDocument(monParser);
    xsl := dbms_xslprocessor.newstylesheet(documentDomXslt, '');
    processeur := dbms_xslprocessor.newProcessor;
    --Application de la feuille de style
    documentDomFragement := dbms_xslprocessor.processxsl(processeur, xsl, documentDom);
    noeud := dbms_xmldom.makenode(documentDomFragement);
    -- L'API DOM pour XMLType peut être utilisée ici
    dbms_xmldom.writetobuffer(noeud, buffer);
    dbms_output.put_line(buffer);

end;
/

```

Nouvelle API Java de manipulation XML

Les différentes API Java disponibles pour manipuler les objets XML Type sont les suivantes :

- Java DOM API pour XML Type ;
- JavaBean API pour XML Type.

Java DOM API pour XML Type

Cette API est le pendant Java de l'API PL/SQL DOM API pour XML Type. Elle est implémentée via le package `oracle.xdb.dom` et permet de traiter des documents XML, qu'ils soient basés sur XML Schéma ou non, et ce, quel que soit leur mode de stockage dans Oracle 10g.

L'API est compatible avec les recommandations DOM du W3C en version 1.0 et 2.0.

Accès aux documents stockés dans XML DB avec JNDI et JDBC

Deux API peuvent être utilisées pour extraire les données stockées dans XML DB : JNDI et JDBC.

La méthode `JNDI lookup()` permet de récupérer un document XML à partir de XML DB en fonction d'un chemin XPath donné. Si le document est basé sur un schéma XML, un JavaBean est généré et une instance de ce bean est renvoyée, sinon c'est une interface DOM qui est retournée. JNDI n'est utilisable qu'en environnement serveur.

Exemple :

```
Document x = (org.w3c.dom.Document)ctx.lookup("/usr/gino/typecommande.xml");
```

En JDBC, il est possible d'interroger une colonne de type XML Type pour récupérer une interface XML Type que l'on pourra manipuler avec l'API DOM Java. La génération de JavaBeans n'est pas supportée car JDBC est conçu pour fonctionner à la fois en environnement client et en environnement serveur.

Exemple simple d'interrogation d'une colonne XML Type :

```
import oracle.xdb.XMLType;

...

OraclePreparedStatement stmt = (OraclePreparedStatement)conn.prepareStatement
    ("select contenu from commande_tab");

ResultSet monRset = stmt.executeQuery();

OracleResultSet monORset = (OracleResultSet) monRset;

while(monORset.next()) {
    XMLType commandxml = XMLType.createXML(monORset.getOPAQUE(1));
    Document document = (Document) commandxml.getDOM();
}
```


Exemple d'utilisation de la méthode getObject() pour récupérer directement un objet XML Type :

```
import oracle.xdb.XMLType;
...
OraclePreparedStatement preparedStmt =
    (OraclePreparedStatement) conn.prepareStatement(
        "select contenu from commande_tab");
ResultSet monRset = preparedstmt.executeQuery();
OracleResultSet monOrset = (OracleResultSet)      monRset;
while(monOrset.next()) {
    XMLType commandxml = (XMLType)monOrset.getObject(1);
    String buffer = commandxml.getStringVal();
}
```

Manipulation des documents stockés dans XML DB avec JDBC

Avec JDBC, il est possible d'ajouter, de supprimer et de mettre à jour des colonnes XML Type. La méthode est simple : on construit un CLOB avec les données XML que l'on mappe à un ordre SQL Select, Insert ou Update, puis on crée un objet XML Type avec le constructeur XMLType() dans le code SQL.

Exemple d'illustration de cette méthode :

```
OraclePreparedStatement monStmt =
    (OraclePreparedStatement) conn.prepareStatement(
        "insert into docs_xml values (7,'NEWHOOK', XMLType(?) )");
String buffer = "<Employe><Numero>4</Numero><Nom>Newhook</Nom></Employe>";
monStmt.setString(1,buffer);
monStmt.execute();
```

Exemple de création d'un objet DOM et stockage dans XML DB selon un schéma XML référencé :

```
import oracle.xdb.XMLType;
import org.w3c.dom.Document;
import org.w3c.dom.Element;
...
OraclePreparedStatement monstmt =
    (OraclePreparedStatement) conn.prepareStatement(
        "insert into docs_xml docs_xml values (8,'NEWHOOK', XMLType(?) )");
String buffer = "<Employe><Numero>1</Numero><Nom>Cossin</Nom></Employe>";
XMLType bufferXML = XMLType.createXML(conn, buffer);
```

```
Document document = (Document)bufferXML.getDOM();
Element root = document.createElement("Employe");
document.insertBefore(document, root);
monstmt.setObject(1,bufferXML);
monstmt.execute();
```

JavaBean API pour XML Type

Cette API est spécialisée dans le traitement de documents basés sur un schéma XML enregistré dans XML DB et est optimisée pour fonctionner côté serveur. Son utilisation est idéale pour :

- les applications serveur ;
- les documents XML Schéma.

Son fonctionnement s'appuie sur une génération automatique de JavaBeans mappés sur des schémas XML : une instance de JavaBean représente une instance de document XML conforme au schéma XML.

Chaque nom de JavaBean généré est unique. Le nom du package `oracle.xdb.bean` est ajouté au nom du schéma XML. Si le schéma XML définit un nom dans une balise `Bean-Name`, c'est ce nom qui sera utilisé, sinon c'est le nom de l'élément qui sera utilisé comme nom de classe.

Par exemple, imaginons que, dans un fichier `typeCommande.xsd`, on trouve un élément `Commande`, le nom du package sera `Commande1` et le nom de la classe sera `Commande`.

Les JavaBeans sont générés automatiquement lors de l'enregistrement du schéma XML dans XML DB lorsque le drapeau (*flag*) `genbean` est placé à VRAI (TRUE), mais ils peuvent aussi être générés manuellement à l'aide de la fonction `generateBean()` du package `DBMS_XMLSCHEMA`.

Les classes générées sont stockées dans le serveur pour utilisation et une copie des fichiers source est placée dans le sous-dossier `bean/` de votre dossier personnel (*home*) du repository XML DB.

Exemple de génération automatique, fichier `exemple.xsd` :

```
<schema targetNamespace="http://www.digora.com/exemple.xsd" version="1.0">
  <element name = "Consultant">
    <complexType>
      <sequence>
        <element name = "Nom" type = "string"/>
        <element name = "Prenom" type = "string"/>
        <element name = "Specialite" type = "string"/>
        <element name = "Experience" type = "positiveInteger"/>
      </sequence>
    </complexType>
  </element>
</schema>
```

```
</sequence>
</complexType>
</element>
</schema>
```

Code source du JavaBean généré par le fichier exemple.xsd :

```
/* Fichier bean généré par Oracle XML DB */
package oracle.xdb.bean.exemple1;
import org.w3c.dom.*;
import oracle.xdb.dom.*;
import oracle.xdb.bean.*;
import java.util.Vector;
public class Consultant extends XMLTypeBean {
    public Consultant(XMLType owner, long xob, long kidnum){
        super(owner, xob, kidnum);
    }
    public Consultant(){
        super(null, 0, 0);
    }
    public int getNom(){
        return super.getScalarString(1);
    }
    public void setNom(String val){
        super.setScalar(0, val);
        return;
    }
    public String getPrenom(){
        return super.getScalarString(1);
    }
    public void setPrenom(String val){
        super.setScalar(1, val);
        return;
    }
    public String getSpecialite(){
        return super.getScalarString(2);
    }
    public void setSpecialite(String val){
        super.setScalar(2, val);
        return;
    }
    public int getExperience(){
        return super.getScalarint(3);
    }
    public void setExperience(int val){
        super.setScalar(3, val);
        return;
    }
}
```

Code source de la procédure stockée générée par le fichier exemple.xsd :

```
Hashtable env = new Hashtable();
env.put(Context.PROVIDER_URL, "/");
env.put(Context.INITIAL_CONTEXT_FACTORY, "oracle.xdb.spi.XDBContextFactory");
Context ctx = (Context)new InitialContext(env);
Consultant obj = (Consultant)ctx.lookup("/exemple.xml");
return obj.getFirstNome();
```

XML Class Generator pour Java

Le générateur de classes XML pour Java crée des fichiers source Java à partir d'une DTD ou de XML Schéma. Celui-ci se révèle utile lorsqu'une application doit envoyer un message XML à une autre application fondée sur le même document de validation ou lors de la création d'un document XML à partir d'un fichier Web. En utilisant les classes créées, l'application Java construit un fichier XML respectant les normes du document de validation utilisé. Le générateur de classes travaille conjointement avec l'analyseur Oracle pour Java, qui a pour objectif d'analyser le fichier de validation d'entrée et de l'aiguiller vers le générateur de classes.

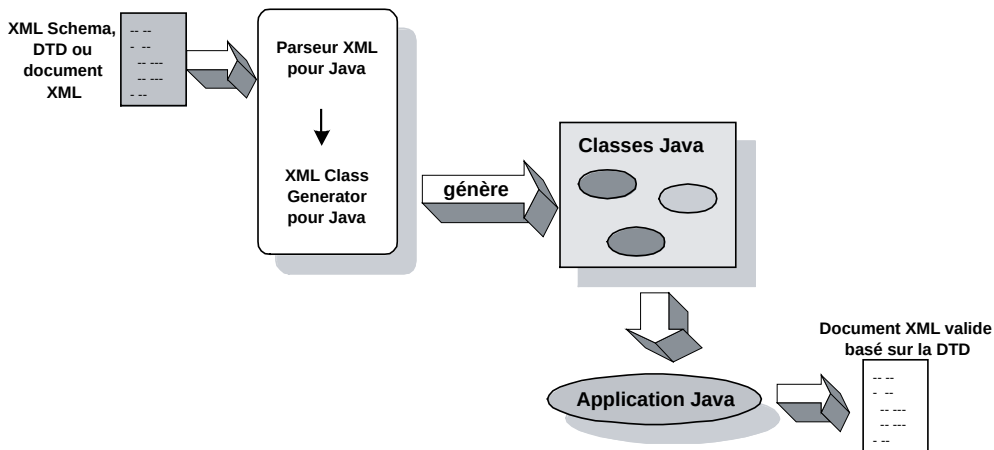


Figure 21-6

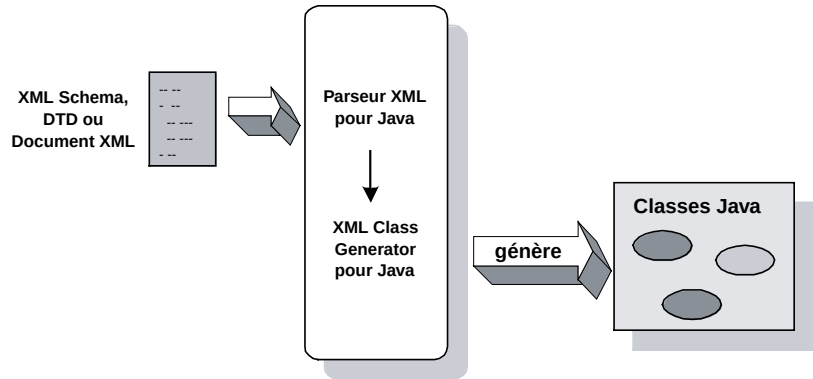
Génération d'un document XML grâce à XML Class Generator

L'exemple suivant montre comment le générateur de classes XML peut être utilisé pour préparer une DTD et générer des classes Java à partir de cette DTD. Il indique également comment les classes peuvent conduire à la construction d'un document XML valide.

Générer des classes Java à partir d'une DTD

Figure 21-7

Génération de classes java
via XML Class Generator



À partir de ce document XML :

```

<?xml version="1.0" encoding="ISO-8859-1" standalone="no" ?>
<!DOCTYPE empListe SYSTEM "employes.dtd" >
<empListe>
  <empDef>
    <empID>7369</empID>
    <empNom>SMITH</empNom>
    <empFonction>CLERK</empFonction>
    <empSal>800</empSal>
  </empDef>
  <empDef>
    <empID>7902</empID>
    <empNom>FORD</empNom>
    <empFonction>ANALYST</empFonction>
    <empSal>3000</empSal>
  </empDef>
  <empDef>
    <empID>7499</empID>
    <empNom>ALLEN</empNom>
    <empFonction>SALESMAN</empFonction>
    <empSal>1600</empSal>
  </empDef>
</empListe>
  
```

et de la DTD suivante :

```

<?xml version="1.0" encoding="ISO-8859-1" ?>
<!ELEMENT empListe (empDef*) >
<!ELEMENT empDef (empID,empNom,empFonction,empSal) >
<!ELEMENT empID (#PCDATA)>
<!ELEMENT empNom (#PCDATA)>
<!ELEMENT empFonction (#PCDATA)>
<!ELEMENT empSal (#PCDATA)>
  
```

on utilise la commande ci-dessous pour générer les classes correspondantes à la DTD précédente, et donc au document XML précédent :

```
java oracle.xml.classgen.oracg -c -dtd employes.xml
```

Les classes générées seront:

- empListe.java ;
- empDef.java ;
- empID.java ;
- empNom.java ;
- empFonction.java ;
- empSal.java.

Exemple de classe générée empNom.java :

```
/* DO NOT EDIT THIS FILE - it is machine-generated */
/* File: EmpNom.java - generated by XML Class Generator version 2.0.0 at Wed Jul 06
 20:04:27 CEST 2005 */

import oracle.xml.parser.v2.DTD;
import oracle.xml.io.XMLObjectInput;
import oracle.xml.io.XMLObjectOutput;
import oracle.xml.comp.CXMLContext;
import oracle.xml.classgen.CGNode;
import oracle.xml.classgen.InvalidContentException;
import java.io.ObjectInput;
import java.io.ObjectOutput;
import java.io.IOException;
import java.io.Externalizable;
public class EmpNom extends CGNode implements Externalizable
{

    public EmpNom()
    {
        super("empNom");
        isValidating = true;
    }

    public EmpNom(String theData) throws InvalidContentException
    {
        this();
        super.addData(theData);
        isValidating = true;
    }

    public void addData(String theData) throws InvalidContentException
    {
        super.addData(theData);
    }
}
```

```
public void deleteData(String theData) throws InvalidContentException
{
    super.deleteData(theData);
}

public void addCDATASection(String theData) throws InvalidContentException
{
    super.addCDATASection(theData);
}

public void deleteCDATASection(String theData) throws InvalidContentException
{
    super.deleteData(theData);
}

public DTD getDTDDNode()
{
    return Empliste.globalDTD;
}

public void validateContent() throws InvalidContentException
{
    super.validateContent();
}

public Empliste getDocument()
{
    return (Empliste)super.getCGDocument();
}

public void writeExternal(ObjectOutput out)
    throws IOException
{
    XMLObjectOutput xout = new XMLObjectOutput(out);
    CXMLContext cxmlContext = new CXMLContext();
    super.writeExternal(xout, cxmlContext);
}

public void readExternal(ObjectInput in)
    throws IOException, ClassNotFoundException
{
    XMLObjectInput xin = new XMLObjectInput(in);
    CXMLContext cxmlContext = new CXMLContext();
    super.readExternal(xin, cxmlContext);
}

}
```

De la même manière, on peut utiliser un XML Schéma pour générer les classes pour la construction de documents XML valides. XML Class Generator, avec les XML Schémas, génère des classes pour tous les éléments de haut niveau (élément, simpletype, complextype).

Par exemple, dans le cas de notre liste d'employés, prenons le XML Class Generator pour Java :

```
java oracle.xml.classgen.oracg -c -schema employes.xsd -package empclass

file:/c:/Documents and Settings/Dave/My Documents/eyerolles/45XML/sources/classgen/
  ➤ xml Schema/employes.xsd<Line 2, Column 156>: XML-24530: (Error) Element 'schema'
  ➤ has invalid namespace: 'http://www.w3.org/2000/10/XMLSchema'
file:/c:/Documents and Settings/Dave/My Documents/eyerolles/45XML/sources/classgen/
  ➤ xml Schema/employes.xsd<Line 21, Column 53>: XML-24528: (Error) Invalid reference:
  ➤ 'http://www.w3.org/2000/10/XMLSchema:float'
```

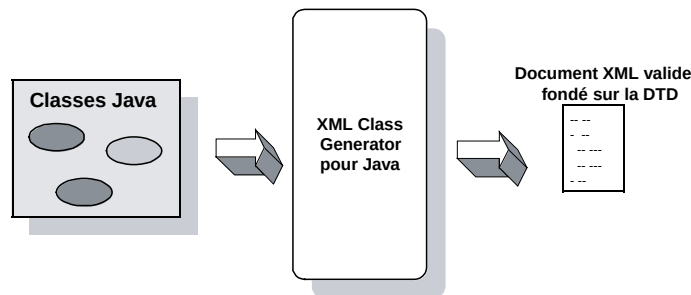
La classe EmpListe.java est générée car, dans ce XML Schéma, empListe est le seul élément de haut niveau. Cette classe permet alors de construire des documents XML valide pour le XML Schéma qui a servi à la générer.

Créer un document valide XML à partir de classes

À partir des classes générées depuis la DTD

Figure 21-8

Génération d'un document XML valide à partir de classes Java via XML Class Generator



Le code Java ci-dessous montre comment des classes Java peuvent être utilisées pour créer un document XML valide. Deux éléments sont générés : emp1 et emp2. Pour construire l'arborescence d'héritage, les différents éléments sont regroupés dans emp1 et emp2. Ces éléments sont ensuite ajoutés en tant que nœuds à l'élément empDef :

```
import oracle.xml.classgen.*;
import oracle.xml.parser.*;

public class EmpGen
{
```



```
public static void main (String args[])
{
    try
    {
        EmpListe liste = new EmpListe();
        // DTD dtd = liste.getDTDNode();

        // Nouvel employé emp1
        EmpDef emp1 = new EmpDef();
        EmpID empid1 = new EmpID("7369");
        EmpNom enom1 = new EmpNom("SMITH");
        EmpFonction fonct1 = new EmpFonction("CLERK");
        EmpSal sal1= new EmpSal("800");

        // Nouvel employé emp2
        EmpDef emp2 = new EmpDef();
        EmpID empid2 = new EmpID("7902");
        EmpNom enom2 = new EmpNom("FORD");
        EmpFonction fonct2 = new EmpFonction("ANALYST");
        EmpSal sal2= new EmpSal("3000");

        // Ajout des éléments à emp1
        emp1.addNode(empid1);
        emp1.addNode(enom1);
        emp1.addNode(fonct1);
        emp1.addNode(sal1);

        // Ajout des éléments à emp2
        emp2.addNode(empid2);
        emp2.addNode(enom2);
        emp2.addNode(fonct2);
        emp2.addNode(sal2);

        // Ajout de emp1 comme nœud de la racine liste
        liste.addNode(emp1);
        // Ajout de emp2 comme nœud de la racine liste
        liste.addNode(emp2);

        liste.validateContent();
        liste.print(System.out);
    }
    catch (Exception e)
    {
        System.out.println(e.toString());
        e.printStackTrace();
    }
}
```

L'application Java ci-dessus crée le document XML comme suit :

```
<?xml version="1.0"?>
<!DOCTYPE empListe SYSTEM "employee.dtd">
<empListe>
  <empDef>
    <empID>7369</empID>
    <empNom>SMITH</empNom>
    <empFonction>CLERK</empFonction>
    <empSal>800</empSal>
  </empDef>
  <empDef>
    <empID>7902</empID>
    <empNom>FORD</empNom>
    <empFonction>ANALYST</empFonction>
    <empSal>3000</empSal>
  </empDef>
```

À partir des classes générées depuis le XML Schéma

Construisons donc avec ces classes Java un document XML contenant les données concernant l'employé Martin :

```
import oracle.xml.classgen.CGXSElement;
import oracle.xml.classgen.SchemaClassGenerator;
import oracle.xml.classgen.InvalidContentException;
import oracle.xml.parser.v2.XMLOutputStream;
import java.io.OutputStream;
import empclass.*;

public class ListeEmployes
{
    static OutputStream output = System.out;
    static XMLOutputStream out = new XMLOutputStream(output);

    public static void main(String args[])
    {
        ListeEmployes listeemployes = new ListeEmployes();
        try
        {
            Empliste.Empliste_Type emplistetype = new Empliste.Empliste_Type();
            Empliste.Empliste_Type.EmpDef.EmpDef_Type empdeftype
            ➡= new Empliste.Empliste_Type.EmpDef.EmpDef_Type();

            // Ajout de l'identifiant
            Empliste.Empliste_Type.EmpDef.EmpDef_Type.EmpID id
            ➡= new Empliste.Empliste_Type.EmpDef.EmpDef_Type.EmpID();
            id.setType(new Integer(1234));
            empdeftype.addEmpID(id);
```

```
// Ajout du nom
Empliste.Empliste_Type.EmpDef.EmpDef_Type.EmpNom nom
➡= new Empliste.Empliste_Type.EmpDef.EmpDef_Type.EmpNom();
nom.setType("MARTIN");
empdeftype.addEmpNom(nom);

// Ajout de la fonction
Empliste.Empliste_Type.EmpDef.EmpDef_Type.EmpFonction fct
➡= new Empliste.Empliste_Type.EmpDef.EmpDef_Type.EmpFonction();
fct.setType(new Empliste.Empliste_Type.EmpDef.EmpDef_Type.EmpFonction.
➡EmpFonction_Type("MANAGER"));
empdeftype.addEmpFonction(fct);

// Ajout du salaire
Empliste.Empliste_Type.EmpDef.EmpDef_Type.EmpSal sal
➡= new Empliste.Empliste_Type.EmpDef.EmpDef_Type.EmpSal();
sal.setType(new Float(10000.));
empdeftype.addEmpSal(sal);

// Création de l'objet définissant l'employé
Empliste.Empliste_Type.EmpDef empdef = new Empliste.Empliste_Type.EmpDef();
empdef.setType(empdeftype);

// Ajout de l'employé à la définition de la liste
emplistetype.addEmpDef(empdef);

// Création de la liste des employés
Empliste empliste = new Empliste();
empliste.setType(emplistetype);
empliste.print(out);
out.writeNewLine();
out.flush();
}
catch(InvalidContentException e)
{
    System.out.println(e.getMessage());
    e.printStackTrace();
}
catch(Exception e)
{
    System.out.println(e.getMessage());
    e.printStackTrace();
}
}
}
```

Après une compilation de toutes les classes, en lançant le programme, on obtient :

```
C:\>java ListeEmployes

<empliste>
<empDef>
<empID>1234</empID>
<empNom>MARTIN</empNom>
<empFonction>MANAGER</empFonction>
<empSa1>10000.0</empSa1></empDef></empliste>
```

XML SQL Utility pour Java

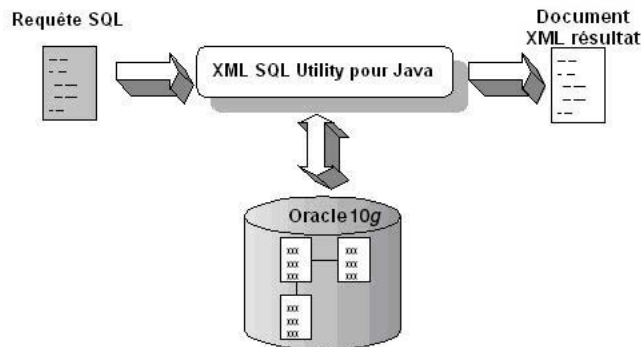
Cet utilitaire est composé de différentes classes Java qui permettent de :

- transmettre la requête à la base de données et générer un document XML (texte ou DOM) ;
- écrire les données XML dans une table de base de données.

Générer un document XML à partir du résultat d'une requête

Figure 21-9

Génération d'un document XML avec XML SQL Utility



XML SQL Utility peut récupérer les requêtes SQL et renvoyer le résultat sous forme d'un document XML.

La structure du document XML résultat se fonde sur la structure interne du schéma de la base de données. Ainsi :

- une colonne deviendra un élément racine ;
- une valeur se transformera en un élément texte uniquement ;
- un objet évoluera vers un élément avec sous-éléments ;
- une collection deviendra une liste d'éléments ;

- une référence objet et une contrainte référentielle se transformeront en IDREF XML (référence, ou liste de références, à un identifiant unique précédemment déclaré par ID).

Cet utilitaire peut générer une représentation du document XML sous forme de chaîne de caractères, tout comme il peut en faire une arborescence d'éléments XML DOM. Utilisez la représentation DOM si vous souhaitez ensuite effectuer quelques changements dans le document XML (à l'aide de XSLT Processor par exemple), sinon préférez-lui la représentation en format chaîne de caractères.

Vous pouvez également utiliser XML SQL Utility pour générer une DTD établie à partir du schéma des tables sur lesquelles sont effectuées les requêtes. Le fichier DTD généré servira comme entrée dans le générateur de classes, qui créera l'ensemble des classes refondées sur les éléments du DTD. Vous pouvez ensuite écrire le code Java utilisant ces classes pour générer une infrastructure derrière un formulaire Web. À partir de cette infrastructure, le formulaire Web récupérera les données et créera un document XML compatible avec le schéma de la base de données. Ces données peuvent être insérées directement dans les tables ou vues correspondantes, sans traitement supplémentaire.

Génération d'un document XML et de données structurées à partir du résultat d'une requête

Dans cet exemple, le code Java construit le fichier XML correspondant à la requête :

SELECT * FROM EMP WHERE JOB = "SALESMAN" ORDER BY ENAME.

```
import java.sql.*;
import java.math.*;
import oracle.xml.sql.query.*;
import oracle.jdbc.*;
import oracle.jdbc.driver.*;

public class employes
{
    public static void main(String args[]) throws SQLException
    {
        DriverManager.registerDriver(new
            oracle.jdbc.driver.OracleDriver());

        // Initialisation de la connexion JDBC
        Connection conn = DriverManager.getConnection
            ("jdbc:oracle:oci8:scott/tiger@monServeur");

        // Création et exécution de la requête JDBC
        // Les données sont récupérées dans un objet ResultSet
        Statement stmt = conn.createStatement();
        ResultSet res = stmt.executeQuery("select * from emp
            where job='SALESMAN' order by ename");

        // Initialisation de la requête XML sur la table EMP
```

```

    OracleXMLQuery reqEmp = new OracleXMLQuery(conn,res);

    // Structure du document XML généré (liste des employés)
    reqEmp.setRowsetTag("empliste"); // Nom de l'élément racine
    reqEmp.setRowTag("empDef");      // Nom des fils
    reqEmp.setRowIdAttrName("numéro");// Nom du numéro de ligne

    // Récupère le document XML des employés dans une chaîne de caractères
    String xmlEmp = reqEmp.getXMLString();

    // Affichage des résultats
    System.out.println(" Liste des employés :\n"+xmlEmp);
}
}

```

Le document XML ci-dessous relatif aux employés a été créé à partir d'un ResultSet. Il comprend toutes les informations sur les employés dont la fonction est SALESMAN.

```

Liste des employés :
<?xml version="1.0"?>
<empliste>
  <empDef numéro="1">
    <EMPNO>7499</EMPNO>
    <ENAME>ALLEN</ENAME>
    <JOB>SALESMAN</JOB>
    <MGR>7698</MGR>
    <HIREDATE>1981-02-20 00:00:00.0</HIREDATE>
    <SAL>1600</SAL>
    <COMM>300</COMM>
    <DEPTNO>30</DEPTNO>
  </empDef>
  <empDef numéro="2">
    <EMPNO>7654</EMPNO>
    <ENAME>MARTIN</ENAME>
    <JOB>SALESMAN</JOB>
    <MGR>7698</MGR>
    <HIREDATE>1981-09-28 00:00:00.0</HIREDATE>
    <SAL>1250</SAL>
    <COMM>1400</COMM>
    <DEPTNO>30</DEPTNO>
  </empDef>
  <empDef numéro="3">
    <EMPNO>7844</EMPNO>
    <ENAME>TURNER</ENAME>
    <JOB>SALESMAN</JOB>
    <MGR>7698</MGR>
    <HIREDATE>1981-09-08 00:00:00.0</HIREDATE>
    <SAL>1500</SAL>
    <COMM>0</COMM>
    <DEPTNO>30</DEPTNO>
  </empDef>

```

```

<empDef numéro="4">
  <EMPNO>7521</EMPNO>
  <ENAME>WARD</ENAME>
  <JOB>SALESMAN</JOB>
  <MGR>7698</MGR>
  <HIREDATE>1981-02-22 00:00:00.0</HIREDATE>
  <SAL>1250</SAL>
  <COMM>500</COMM>
  <DEPTNO>30</DEPTNO>
</empDef>
</empListe>

```

Ce second exemple génère directement le document XML à partir de la requête sans utiliser l'objet ResultSet. La requête SQL `SELECT * FROM DEPT` apparaît alors dans l'objet `OracleXMLQuery`.

```

import java.sql.*;
import java.math.*;
import oracle.xml.sql.query.*;
import oracle.jdbc.*;
import oracle.jdbc.driver.*;

public class deptDefs
{
    public static void main(String args[]) throws SQLException
    {
        DriverManager.registerDriver(new
            oracle.jdbc.driver.OracleDriver());

        // Initialisation de la connexion JDBC
        Connection conn = DriverManager.getConnection
            ("jdbc:oracle:oci8:scott/tiger@monServeur");

        // Initialisation de la requête XML sur la table DEPT
        OracleXMLQuery reqDept = new OracleXMLQuery
            (conn,"select * from dept");

        // Structure du document XML généré (liste des départements)
        reqDept.setRowsetTag("deptListe"); // Nom de l'élément racine
        reqDept.setRowTag("deptDef"); // Nom des fils

        // Récupère le document XML des employés dans une chaîne de caractères
        String xmlDept = reqDept.getXMLString();

        // Affichage des résultats
        System.out.println(" Liste des départements :\n"+xmlDept);
    }
}

```

Voici le document XML généré qui retourne toutes les lignes de la table DEPT :

Liste des départements :

```
<?xml version="1.0"?>
<deptListe>
  <deptDef num="1">
    <DEPTNO>10</DEPTNO>
    <DNAME>ACCOUNTING</DNAME>
    <LOC>NEW YORK</LOC>
  </deptDef>
  <deptDef num="2">
    <DEPTNO>20</DEPTNO>
    <DNAME>RESEARCH</DNAME>
    <LOC>DALLAS</LOC>
  </deptDef>
  <deptDef num="3">
    <DEPTNO>30</DEPTNO>
    <DNAME>SALES</DNAME>
    <LOC>CHICAGO</LOC>
  </deptDef>
  <deptDef num="4">
    <DEPTNO>40</DEPTNO>
    <DNAME>OPERATIONS</DNAME>
    <LOC>BOSTON</LOC>
  </deptDef>
</deptListe>
```

Écrire du code XML dans une base de données

Il est également possible d'insérer des données XML dans une table objet d'une base Oracle 10g à l'aide de XML SQL Utility.

Figure 21-10

Écriture de code XML
avec XML SQL Utility



En stockant des documents XML, vous préservez la structure des documents. Ce processus est inversé par rapport au processus de génération de classes issu d'un schéma de base de données. Ainsi :

- une balise deviendra une colonne ;
- un élément texte uniquement évoluera vers un type ;
- un élément avec sous-éléments se transformera en un type objet ;

- une liste d'éléments deviendra une collection ;
- les données non structurées, comme une description ou une définition, ne peuvent être sauvegardées dans une table, c'est pourquoi ce type de donnée est stocké en tant que CLOB dans la base de données.

Par exemple, le code Java ci-dessous insère des données dans la table EMP à partir du fichier XML emp.xml (fichier conforme à la structure de la table EMP).

```
import oracle.xml.sql.dml.*;
import java.sql.*;
import oracle.jdbc.driver.*;
import oracle.jdbc.*;
import java.net.*;

public class ecrire
{
    public static void main(String args[]) throws SQLException
    {
        String nomTab = "EMP1"; // Table dans laquelle vont être insérés les nouveaux
        ➡ enregistrements
        String nomFic = "emp.xml"; // Document XML contenant les enregistrements

        DriverManager.registerDriver(new
            oracle.jdbc.driver.OracleDriver());

        // Initialisation de la connexion JDBC
        Connection conn = DriverManager.getConnection
            ("jdbc:oracle:oci8:scott/tiger@monServeur");

        OracleXMLSave req = new OracleXMLSave(conn, nomTab);

        URL url = req.createURL(nomTab);
        int lignes = req.insertXML(url);

        System.out.println(" Insertion réussie : "+lignes+
            " lignes insérées dans "+ nomTab);
        conn.close();
    }
}
```

Avant d'exécuter ce code, il faudra créer la table EMP1 à l'image de la table EMP :

```
create table EMP1 (
    EMPNO NUMBER(4),
    ENAME CHAR(10),
    JOB VARCHAR2(9)
);
```

Le fichier emp.xml utilisé pour remplir cette table est le suivant :

```
<?xml version="1.0" ?>
<empListe>
  <empDef num="1">
    <EMPNO>7369</EMPNO>
    <ENAME>SMITH</ENAME>
    <JOB>CLERK</JOB>
  </empDef>
  <empDef num="2">
    <EMPNO>7499</EMPNO>
    <ENAME>ALLEN</ENAME>
    <JOB>SALESMAN</JOB>
  </empDef>
  <empDef num="3">
    <EMPNO>7521</EMPNO>
    <ENAME>WARD</ENAME>
    <JOB>SALESMAN</JOB>
  </empDef>
</empListe>
java.lang.NoClassDefFoundError: oracle/xdb/XMLType
    at ecrire.main(ecrire.java:21)
Exception in thread "main"
```

Après exécution du code Java, ce message s'affichera :

```
Insertion réussie : 3 lignes insérées dans EMP1
```

Vérifions dans la table EMP1 les enregistrements qui ont été insérés :

```
SQL> select * from EMP1;

   EMPNO  ENAME      JOB
-----
   7369  SMITH        CLERK
   7499  ALLEN        SALESMAN
   7521  WARD         SALESMAN

3 ligne(s) sélectionnée(s).
```

XDK JavaBeans

Il s'agit d'un ensemble de JavaBeans qui facilitent l'utilisation du XML dans les applications développées avec le XML Developer Kit. Certains sont des Beans graphiques, c'est-à-dire des composants utilisés dans les IHM utilisant Swing ou AWT, les autres sont des bibliothèques de méthodes automatisant les traitements XML.

DOMBuilder Bean

Ce Bean construit un arbre DOM à partir d'un document XML. Il encapsule le parseur XML dans une interface de Bean et ajoute des fonctionnalités comme le traitement asynchrone. Utilisé pour traiter plusieurs documents de façon asynchrone et en parallèle dans des threads système, ce Bean peut faire gagner du temps car il se montre 40 % plus rapide qu'un traitement de fichiers XML un par un.

XSL Transformer Bean

Ce Bean permet de faire de la transformation XSLT avec un document XML et une feuille de styles donnés. En combinaison avec les autres Beans, il permet de visualiser directement les résultats de transformations XSLT effectuées dans l'application.

XMLSourceView Bean

Ce Bean est un composant visuel qui permet la visualisation et l'édition de documents XML avec coloration syntaxique et pré ou post-validation avec une DTD. Il s'agit d'un composant technique qui ne peut pas être personnalisé et donc qui a peu d'intérêt dans le cadre d'une application cliente.

XMLTransformPanel Bean

Ce Bean est un composant visuel qui effectue des transformations XSLT, affiche le résultat et permet l'édition des fichiers XML et XML source.

DBViewer Bean

Ce Bean effectue des requêtes SQL dans la base de données et transforme directement les données XML retournées avec une feuille de style XSL. Il possède plusieurs tampons de données, un pour les données XML, un pour les données XSL et un pour le résultat des opérations de transformation. Ces tampons lui confèrent une bonne flexibilité.

DBAccess Bean

Ce Bean maintient des tables CLOB qui stockent plusieurs documents texte et XML. Les tables sont créées à l'aide du script suivant :

```
CREATE TABLE tablename FILENAME CHAR( 16) UNIQUE, FILEDATA CLOB) LOB(FILEDATA)  
STORE AS (DISABLE STORAGE IN ROW)
```

Chaque document qui va être transféré au Bean sera stocké dans une ligne d'une de ces tables.

XMLDiff Bean

Ce Bean visuel compare deux arbres XML DOM, les affiche et met en évidence leurs différences. Les nœuds peuvent être manipulés (ajout, suppression, déplacement) et l'opération de différence peut être relancée.

XSQL Servlet

XSQL Servlet est une servlet Java qui permet de générer un document XML à partir d'interrogation SQL. Non seulement cette servlet gère le flux XML, mais elle peut aussi réaliser ou non la transformation XSL sur le serveur. Pour cela, elle utilise XML Parseur en Java, XML SQL Utility et le moteur de transformation XSL (XSLT).

XSQL Servlet peut être utilisée dans différents cas :

- la construction dynamique de documents XML devant contenir le résultat d'une ou plusieurs requêtes. Les documents ainsi générés peuvent rester au format XML ou être transformés via XSLT ;
- le stockage de données XML en base.

La réalisation d'une application utilisant XSQL Servlet est simple, elle consiste à écrire des fichiers .xsql. Ces fichiers sont des documents XML utilisant des balises spécifiques pour exécuter une requête SQL, appliquer une feuille de style...

Il est aussi possible de spécifier dans un fichier XSQL plusieurs feuilles de style pour répondre à plusieurs clients. L'outil Oracle XSLT Processor appliquera sélectivement la feuille de style appropriée au poste client.

Dans cet exemple :

- avec le navigateur Lynx, le document est transformé selon la feuille de style rowcol_lynx.xsl sur le serveur ;
- avec le navigateur Internet Explorer 5.0, le document est transformé selon la feuille de style rowcol-ie.xsl sur le poste client ;
- pour tous les autres navigateurs, le document est transformé selon la feuille de style rowcol.xsl sur le serveur.

```
<?xml version="1.0"?>
<?xml-stylesheet type="text/xsl" media="lynx" href="rowcol_lynx.xsl" ?>
<?xml-stylesheet type="text/xsl" media="msie 5.0" client="yes"
    href="rowcol-ie5.xsl" ?>
<?xml-stylesheet type="text/xsl" href="rowcol.xsl" ?>
```

Exemple de recherche dans une table via XSQL :

```
<?xml version="1.0"?>
<?xml-stylesheet type="text/xsl" href="empdept.xsl"?>
<xsql:query connection="demo" null-attribute-indicator="true" xmlns:xsql="urn:oracle-xsql">
    SELECT * FROM department
</xsql:query>
```

Résumé

Ce chapitre vous a présenté le langage XML de manière générale, ses avantages et sa simplicité de mise en œuvre, ainsi que le fonctionnement des outils d'Oracle 10g R2 prenant en charge cette technologie.

Les outils suivants :

- Oracle 10g XML DB ;
- Oracle Text (anciennement nommé *interMedia Text*) ;
- XML Parseur et XSLT Processor (Java, C, C++, PL/SQL) ;
- XML Class Generator (Java et C++) ;
- XML SQL Utility pour Java ;
- XDK JavaBeans ;
- XSQL Servlet ;
- Oracle SOAP.

ont été abordés.

Les dernières versions de ces outils sont livrées avec la base de données ou téléchargeables à partir du site <http://technet.oracle.com>.

Par rapport à Oracle 10g, Oracle 10g R2 étend son support des standards du W3C que sont XML Schéma et XPath. De son côté, Oracle Text a été amélioré pour supporter de nouveaux opérateurs et un nouveau type d'index.

Oracle continue à pousser son support natif d'XML et met en œuvre de grands moyens pour faciliter le traitement de ces documents.

Présentation d'Oracle Application Express

Dans ce chapitre :

- la présentation d'Oracle Application Express ;
- le positionnement d'Oracle Application Express par rapport à d'autres outils de développement.

Ce chapitre est consacré à l'une des nouveautés majeures d'Oracle 10g : Application Express. Successeur de l'antique WebDB, le projet Marvel, baptisé officiellement Oracle Application Express, propose un environnement de développement accessible à 100 % depuis un navigateur Internet pour créer simplement des applications Web centrées autour d'une base de données Oracle.

Oracle Application Express est le nouveau nom officiel d'HTML DB. Ce changement de nom ne s'est accompagné d'aucune modification du logiciel. Le marketing Oracle a recherché une cohérence entre la version gratuite Oracle Database Express Edition (la base de données) et Oracle Application Express (l'outil de développement).

Introduction à Oracle Application Express

Oracle Application Express est une suite intégrée de développement d'applications en ligne, qui permet de construire des applications Web dynamiques. Il tire pleinement parti de la puissance de la base de données Oracle, et ce à partir de votre navigateur favori.

Cet environnement de développement Web simplifie le travail des programmeurs :

- la sécurité d'accès est gérée nativement ;
- les pages s'affichent en fonctions des droits des utilisateurs (lecture, modification...) ;
- l'enchaînement des pages est automatique ;
- le développement se fait par des assistants ;
- les menus et sous-menus sont intégrés ;
- la réalisation de formulaires est extrêmement simple ;
- l'import et l'export de données s'effectuent dans différents formats ;
- possibilité de créer des états et des graphiques ;
- possibilité d'appliquer et de modifier des feuilles de style d'une application ;
- le potentiel des développements réalisables par cet outil est très important.

Cet environnement repose sur le PL/SQL, le langage procédural d'Oracle. Présent depuis de nombreuses années dans les bases Oracle, c'est un élément de stabilité et de pérennité. La courbe d'apprentissage de cet outil sera rapide pour des utilisateurs connaissant déjà le PL/SQL. De plus, Oracle Application Express est très simple d'accès. Son utilisation est facilitée par de nombreux assistants qui guident les développeurs.

Quel est le positionnement d'Oracle Application Express ?

Oracle Application Express permet de développer rapidement des applications simples... et elles sont nombreuses ! Dans son exubérance, le marketing Oracle n'hésite pas à positionner Oracle Application Express comme le « Microsoft Access killer ». C'est sans doute aller trop loin mais l'idée est là : créer une structure de base de données, des écrans transactionnels, des menus, des éditions, gérer la sécurité d'accès, etc., en utilisant la puissance d'une base Oracle (procédures stockées, fonctions, triggers...), le tout en mode Web.

Nous sommes donc en présence d'un outil de nouvelle génération. Moins puissant et surtout moins complexe que le serveur d'application Java Oracle Application Server associé à l'outil de développement Oracle JDeveloper, Oracle Application Express possède suffisamment d'atouts pour réaliser de nombreuses applications tactiques.

Les développeurs ayant utilisé l'outil client serveur Oracle Forms y retrouveront certaines similitudes. Doté d'une philosophie de développement de type L4G (langage de 4^e génération), Oracle Application Express est toutefois moins puissant, l'interface Web ne permettant pas de disposer d'une interface aussi sophistiquée. Par exemple, nous serons amenés à naviguer d'écran à écran avant d'accéder à une caractéristique souhaitée.

D'autres langages de développement permettent d'accéder à Oracle, comme le très populaire PHP, ou encore Microsoft .NET. Ce sont là les véritables challengers d'Oracle Application Express pour les applications centrées autour d'une base Oracle.

Oracle propose un autre outil offrant des possibilités de développement d'applications : Oracle Portal. Ce module, intégré à l'offre Oracle Application Server, se positionne plus comme un portail capable d'intégrer des documents, des services et des applications développées avec Oracle Application Express ou en Java.

Outre l'avantage de la simplicité, Oracle Application Express dispose d'un autre atout : son prix. Livré en standard avec toutes les versions d'Oracle Database 10g, il ne nécessite aucune acquisition supplémentaire. Si vous disposez d'un contrat de maintenance Oracle, vous bénéficiez automatiquement des nouvelles versions de la base de données, donc d'Oracle Application Express. Autre atout majeur, vous bénéficiez du support Oracle pour Oracle Application Express au même titre que pour la base de données. C'est un facteur différenciateur comparé à des développements mi-Oracle, mi-logiciels libres.

Oracle Application Express est nativement conçu pour la version 10g de la base, mais il est tout à fait possible de l'utiliser avec une version plus ancienne, la 9i.

Quelles sont les applications cibles ?

Oracle Application Express s'adresse principalement aux applications :

- tactiques ;
- utilisées par plusieurs utilisateurs ;
- devant être rapidement mises en production (quelques jours ou semaines, pas des mois) ;
- pouvant être développées par des développeurs ne possédant pas un « bac + 15 » ;
- suffisamment critiques pour votre business pour qu'elles fonctionnent avec une base de données fiable.

Cela ne signifie pas toutefois que vous ne pouvez pas créer des applications :

- stratégiques ;
- possédant des caractéristiques sophistiquées ;
- auxquelles accèdent de très nombreux utilisateurs.

Prenons l'exemple d'une application permettant :

- de gérer de multiples espaces de travail et de nombreux comptes de développeurs ;
- de gérer des circuits (*workflow*) de demande et d'approbation ;
- d'offrir une interface de développement Web disposant de modèles d'interfaces (*templates*) et d'une gestion automatique des sessions utilisateurs ;
- de télécharger des images et des feuilles de style en cascade (CSS) pour les incorporer dans les applications ;
- supporter des centaines d'utilisateurs ;
- etc.

Si cette description vous est familière, c'est parce que je viens de vous décrire Oracle Application Express. En effet, Oracle Application Express a été développé en utilisant... Oracle Application Express ! Ce logiciel, composé d'environ 10 applications, comporte environ 900 pages, toutes en Oracle Application Express. Bien que spécialisé pour des applications d'une taille relativement restreinte, le concept de « développer simplement les choses simples » peut être étendu.

Les sources d'information

Les URL indiquées ci-dessous sont sous réserve de modifications ultérieures à la parution de cet ouvrage.

Oracle Application Express étant une suite intégrée de développement d'applications en ligne, Oracle a mis à disposition des développeurs de nombreuses ressources disponibles sur Internet.

Le « Product Center » Oracle Application Express

<http://www.oracle.com/technology/products/database/htmldb>

Cette première ressource est située sur le site <http://otn.oracle.com> ou OTN (Oracle Technology Network), qui est le site technique d'Oracle. Elle est identique à l'adresse <http://www.oracle.com/technology>. Elle présente Oracle Application Express et propose des liens vers les autres ressources accessibles.

Tester Oracle Application Express sur Internet

<http://htmldb.oracle.com>

Oracle Application Express étant un outil 100 % Web, Oracle vous propose de l'évaluer et de développer depuis votre navigateur sans rien avoir à installer. Vous ne bénéficiez plus d'aucune excuse pour ne pas le tester !

Oracle Application Express Studio

<http://htmldb.oracle.com/studio>

Cette rubrique partage des applications développées avec Oracle Application Express. Vous pouvez y déposer vos applications et accéder à celles mises à disposition par d'autres utilisateurs. Vous y trouverez d'excellents exemples !

Les « How To » Oracle Application Express

<http://www.oracle.com/technology/products/database/htmldb/howtos/index.html>

Cette rubrique *How to* (Comment faire pour...) est un sous-ensemble de la rubrique plus générale OBE (Oracle By Example) ciblée sur l'apprentissage d'Oracle Application Express.

Le forum Oracle Application Express

<http://forums.oracle.com/forums/forum.jspa?forumID=137>

Le site OTN dispose d'un forum technique consacré à Oracle Application Express. Contrairement au site du support technique d'Oracle Metalink (<http://metalink.oracle.com>) accessible uniquement aux sociétés bénéficiant d'un support technique payant, les forums d'OTN sont libres et gratuits.

Le support Oracle

<http://metalink.oracle.com>

Si votre contrat de maintenance est à jour, vous bénéficiez officiellement du support Oracle pour votre base et pour Oracle Application Express : utilisez-le !

Demandez à « Tom »

<http://asktom.oracle.com>

Tom Kyte est un gourou technologique d'Oracle reconnu pour sa pédagogie. Dans sa rubrique *Ask Tom*, certains sujets abordent Oracle Application Express. Fait important, la rubrique *Ask Tom* a été entièrement développée avec Oracle Application Express.

RCI Informatique

<http://www.rci-informatique.fr>

RCI Informatique est un prestataire français disposant de compétences Oracle Application Express. Il propose des ressources sur son site Web.

L'architecture d'Oracle Application Express

Oracle Application Express est un outil déclaratif hébergé dans une base de données Oracle permettant de développer et de déployer des applications Web agissant sur les données d'une base. Cet environnement permet à plusieurs groupes de travail de développer des applications autonomes comme si elles travaillaient avec leur propre base de données. Grâce à ses fonctionnalités, comme les thèmes graphiques, les feuilles de style CSS, les contrôles de navigation, la gestion dynamique des droits d'accès, les éditions, Oracle Application Express accélère le développement d'applications.

Toutes les applications sont conservées dans des tables spécifiques de la base de données. Lorsque vous créez une application, aucun code n'est généré : des métadonnées sont créées ou modifiées à l'intérieur de la base de données. Le moteur d'exécution d'Oracle Application Express est chargé d'interagir avec le développeur ou l'utilisateur pour exploiter ces méta-données et générer dynamiquement l'application.

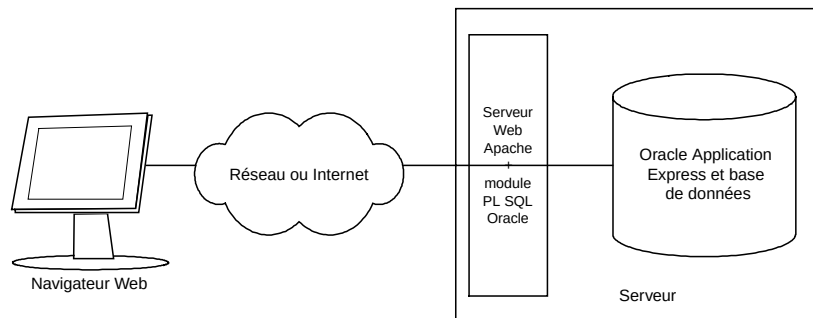
Le statut de chaque session utilisateur est conservé automatiquement dans la base, ne nécessitant aucun codage supplémentaire. Les développeurs peuvent accéder et modifier les sessions par de simples substitutions.

L'architecture technique

Comme l'indique la figure suivante, un simple navigateur suffit pour développer et utiliser des applications Oracle Application Express. Cette d'architecture permet de déployer des applications aussi bien sur un réseau local que sur Internet. Dans ce dernier cas, il faudra toutefois utiliser les règles de l'art en intercalant sur le réseau une DMZ (Zone Démilitarisée) et un pare-feu (*firewall*).

Figure 22-1

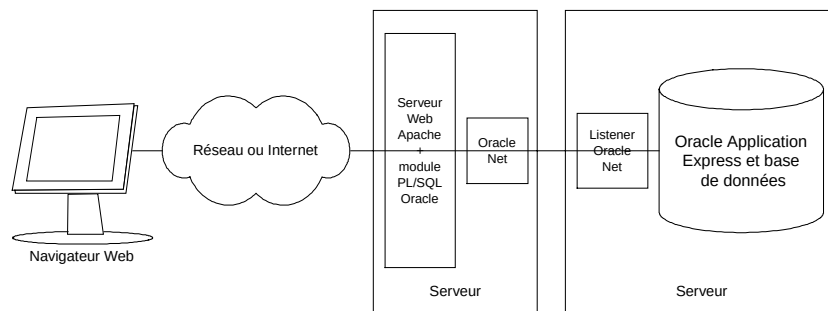
Architecture technique
d'Oracle Application
Express



Fidèle à sa modularité, le serveur Web Apache et le module PL/SQL fournis par Oracle ne sont pas forcément situés sur le même serveur. Il suffit alors d'installer le serveur Web sur un serveur et la base Oracle Application Express sur un autre serveur, de configurer Oracle Net entre ces serveurs (voir le chapitre 15, *Oracle Net, le middleware Oracle*) puis de configurer le DAD (Database Authentication Descriptor) du module PL/SQL Oracle pour utiliser cette architecture.

Figure 22-2

Oracle Application
Express installé
sur deux serveurs



Le serveur HTTP d'Oracle Application Express est un serveur Apache que l'on peut utiliser en développement et en déploiement. Il comprend une passerelle pour accéder au moteur PL/SQL et aux objets de la base dans laquelle est installée Oracle Application Express. La passerelle utilise des DAD (Database Authentication Descriptor) pour se connecter à la base. Ces DAD requièrent une description complète de la chaîne de connexion : nom d'utilisateur Oracle, mot de passe et alias de l'instance. Les DAD sont

utilisés pour transformer les URL en appels spécifiques à une procédure PL/SQL stockée. Ainsi, l'URL suivante :

`http://serveur_web:80/pls/DatabaseAuthenticationDescriptor/user.package.procedure`

va appeler la procédure stockée `procedure` dans le package `package`, dans le schéma de `user`, en utilisant la chaîne de connexion `DAD` sur le port 80 du serveur `serveur_web`. C'est sur ce principe que repose Oracle Application Express.

Configuration requise

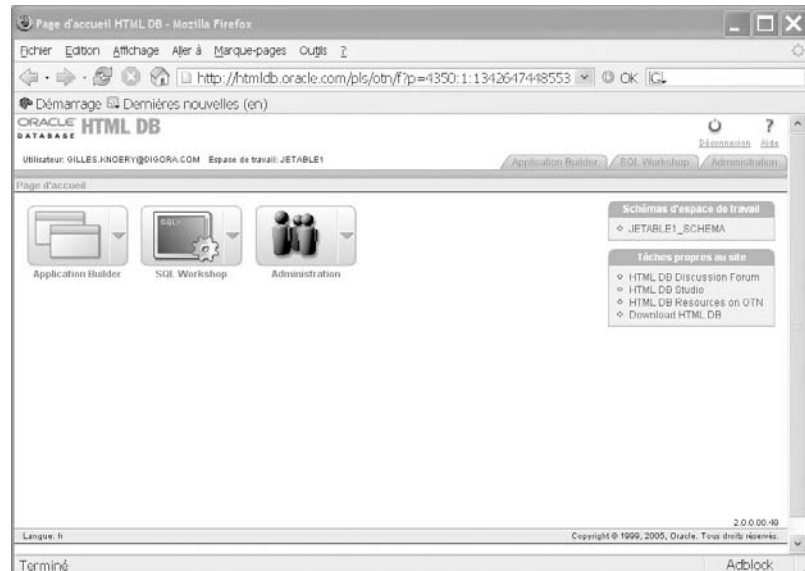
Oracle Application Express, ainsi que le serveur Web, sont livrés sur le CD-Rom « Companion » d'Oracle Database 10g. Leur installation est simple et nécessite la présence d'une base Oracle pré-existante.

Oracle Application Express est une nouveauté d'Oracle Database 10g mais la version actuelle peut aussi être installée dans une version Oracle9i R2 version 9.2.0.3 ou plus. Pour plus d'informations, consultez les ressources techniques indiquées en début de chapitre à ce sujet.

Les composants d'Oracle Application Express

Figure 22-3

Écran d'accueil
d'Oracle Application
Express



La plate-forme de développement d'Oracle Application Express s'articule autour de trois composants :

- Application Builder : pour construire les applications, importer et exporter des données issues d'un tableur ;

- SQL Workshop : créer, modifier et interroger le schéma de données ;
- Administration : gérer les développeurs, les services, l'activité et les utilisateurs avec leurs droits d'accès.

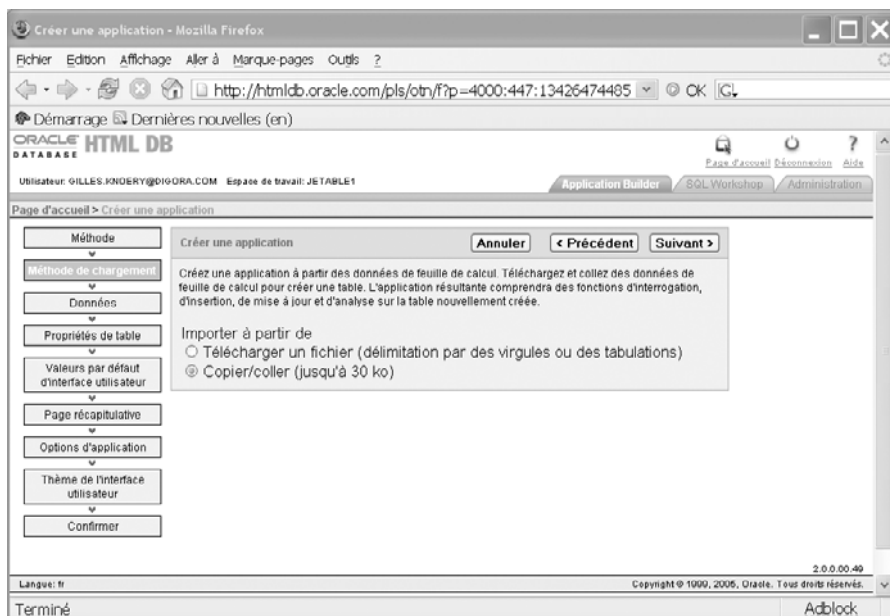
Application Builder

Application Builder permet de créer une interface HTML (ou une application) autour d'objets de la base de données, comme des tables ou des procédures. Une application est constituée d'un ensemble d'écrans HTML générés par la base Oracle Application Express. Ces écrans sont liés entre eux par l'usage de boutons, d'onglets et de liens hypertextes. Une fois l'application réalisée, le moteur Oracle Application Express génère l'application en utilisant des modèles ou templates prédéfinis ainsi que les modifications de l'interface graphique effectuées.

La page constitue la brique de base d'une application. Elle se compose de boutons, de champs, et peut comporter la logique de traitement. Vous pouvez lier une page à une autre suivant vos conditions, effectuer des calculs, des validations, afficher des éditions, des écrans ou des graphiques.

Figure 22-4

Création d'une application à partir de données issues d'un tableur



L'Application Builder permet d'importer ou d'exporter des données entre la base et un navigateur et de créer une application centrée autour des données importées. Les sources de données possibles sont le format texte (séparé par des virgules ou des tabulations), des fichiers au format XML, des feuilles de calcul Excel. Les données exportées sont au format texte ou des documents XML.

Les assistants créent une nouvelle table et insèrent les données sans intervention supplémentaire. Une fois les données chargées dans la base, une application autour de cette table est créée comme avec n'importe quelle autre table de la base de données.

SQL Workshop

SQL Workshop permet de visualiser et d'administrer les objets de la base depuis un navigateur Web. Ses principales fonctions sont :

- exécuter des ordres SQL ;
- télécharger et exécuter des scripts SQL ;
- tracer les ordres SQL exécutés ;
- créer ou modifier les objets de la base de données ;
- interroger simplement les objets de la base ;
- visualiser une partie du dictionnaire de données ;
- naviguer dans les objets de la base.

La gestion des droits d'accès d'Oracle Application Express

Oracle Application Express possède de façon native une gestion des droits d'accès simple et sophistiquée. Cette gestion des droits rend l'usage d'Oracle Application Express possible pour des groupes de développeurs de diverses tailles. C'est l'un des « plus » apportés par l'expérience d'Oracle dans les outils de développement.

Tout d'abord, toutes les applications ne nécessitent pas forcément une connexion. Pour les utilisateurs, vous avez la possibilité de fonctionner en mode connecté ou non connecté. Dans le premier cas, suivant le profil de l'utilisateur, vous pouvez agir sur l'application, modifier son comportement, les enchaînements de pages, les champs visibles, etc. Dans le second cas, le fonctionnement sera identique pour tous. La gestion des utilisateurs revient à l'administrateur de l'application ou peut être incorporée dans votre développement.

La gestion des droits d'accès peut être mise en place en liaison avec un annuaire LDAP, par exemple OID, celui d'Oracle.

Pour les développeurs, trois profils d'administration existent :

- l'administrateur d'Oracle Application Express ;
- l'administrateur d'espaces de travail ou *workspaces* ;
- les développeurs.

L'administrateur d'Oracle Application Express possède tous les droits sur l'environnement Oracle Application Express. C'est le rôle le plus puissant, dont la première action

est d'installer Oracle Application Express. Lors de cette étape, il a la possibilité de configurer ou non les workspaces.

Si les workspaces sont installés, chaque administrateur de workspace possède tous les droits sur son espace de travail. Dans chaque workspace, son administrateur peut créer plusieurs applications.

Enfin, même si aucun workspace n'a été créé, des développeurs sont associés à chaque application. Dans le cas d'utilisation des workspaces, les développeurs devront indiquer le workspace, leur identifiant et leur mot de passe. Seul l'identifiant et le mot de passe du développeur sont nécessaires si les workspaces ne sont pas installés.

Une fois ces concepts présentés, nous vous proposons de nous appuyer, à titre d'exemple, sur le développement d'une application simple.

Résumé

Ce chapitre a brièvement présenté le nouvel outil de développement Oracle Application Express. Cet outil permet de réaliser de véritables applications de type Internet manipulant des données. Ces sites sont complètement dynamiques puisque formés de composants (données et procédures) contenus dans la base de données.

Cet outil de développement rapide d'applications possède de nombreux atouts et il convient de l'évaluer sérieusement avant de se lancer dans des développements utilisant une technologie plus complexe.

Partie V

Administration d'Oracle 10g

Chapitre 23 : Gestion de l'espace disque et des fichiers

Chapitre 24 : Stratégie de sécurité sous Windows

Chapitre 25 : Oracle Enterprise Manager

Chapitre 26 : La sauvegarde d'une base Oracle 10g

Chapitre 27 : Configurer les disques pour Oracle

Chapitre 28 : Optimisation et performances

Gestion de l'espace disque et des fichiers

Dans ce chapitre :

- les différences entre base de données physique et logique ;
- le principe du tablespace ;
- les tablespaces temporaires, d'annulation et de données ;
- le DB_BLOCK_SIZE ;
- la notion de segment, d'extent et de bloc ;
- les différents types de segments ;
- la gestion des fichiers de données ;
- la gestion des fichiers redo-log ;
- la gestion des fichiers de contrôle ;
- la gestion automatique des fichiers par Oracle ;
- les différents types de fichiers d'initialisation.

Une base Oracle est conçue pour stocker des données, les modifier, les supprimer, etc. Toutes ces actions sont conservées dans les fichiers qui composent la base. Mais comment Oracle gère-t-il l'espace disque alloué au cours de toutes ces opérations ? Comment déterminer la taille de la base de données ? Comment contrôler l'évolution des différents constituants et anticiper les problèmes ? De quelle nature peuvent être ces

problèmes ? Ce chapitre répond à toutes ces questions par une approche théorique et pratique, pour vous aider à comprendre et agir au cœur de la gestion de l'espace disque des bases Oracle 10g.

Avant toute action, effectuez une sauvegarde de la base de données et entraînez-vous au préalable sur une base test, sans utilisateur actif ni données stratégiques.

Les apports d'Oracle Enterprise Manager

Oracle Enterprise Manager propose des outils graphiques puissants qui couvrent très bien la gestion de l'espace disque et des fichiers ainsi que toutes les possibilités présentées dans ce chapitre.

Pourquoi alors ne pas les avoir présentés ici ? Nous avons volontairement fait l'impasse sur eux, pour vous faire découvrir les entrailles d'Oracle, de façon à comprendre, au-delà d'une simple interface graphique, la richesse et la puissance des options proposées.

Je vous conseille donc de lire ce chapitre pour comprendre le fonctionnement interne d'Oracle et d'utiliser en parallèle Oracle Enterprise Manager pour son ergonomie. Vous serez ainsi capable de maîtriser à la fois la forme et le fond.

Gestion de l'espace disque

La classification des fichiers décrite au chapitre 9, *Les fichiers d'une base Oracle 10g*, permet de réduire le nombre de ceux qui requièrent une attention particulière de la part de l'administrateur Oracle. De plus, comprendre l'utilisation d'un fichier de données, de contrôle ou redo-log revient à les maîtriser dans leur ensemble, car les fichiers d'une même famille ont un fonctionnement identique.

Base de données physique et logique

L'ensemble des fichiers (données, redo-log, contrôle) visibles par un administrateur Windows constitue la base de données *physique* (figure 23-1).

Un développeur Oracle ou un utilisateur non DBA ne se soucie pas de ces fichiers physiques. De leur point de vue, ils manipulent des tables, des vues ou des index, qui constituent la base de données *logique*. C'est à l'administrateur Oracle qu'incombe le soin de faire le lien entre la base physique et la base de données logique. C'est la notion du tablespace Oracle qui permet d'établir ce lien.

Principe du tablespace

Le tablespace est un concept fondamental du stockage des données dans une base Oracle. Une table ou un index appartient obligatoirement à un tablespace.

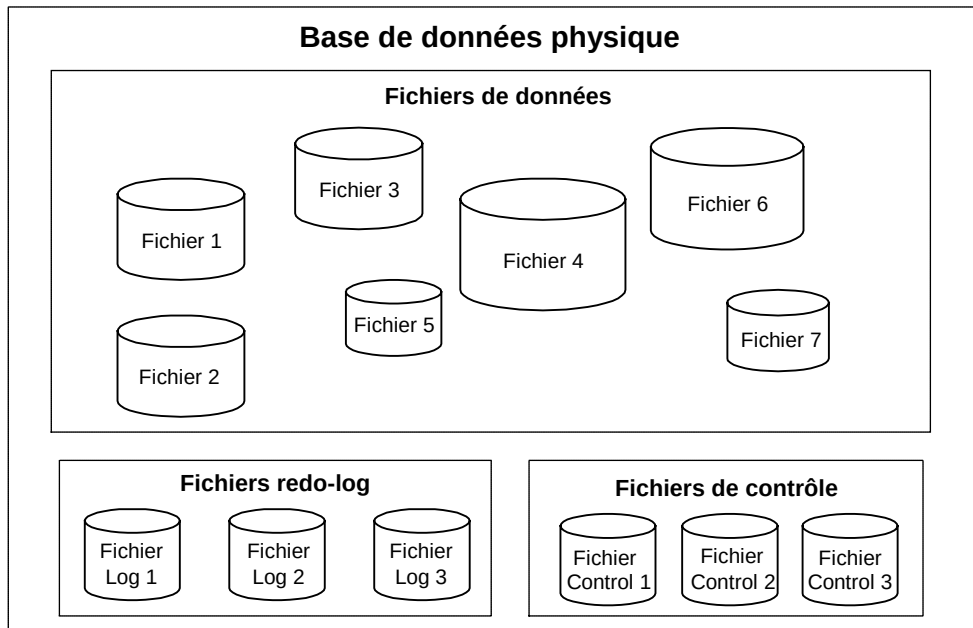
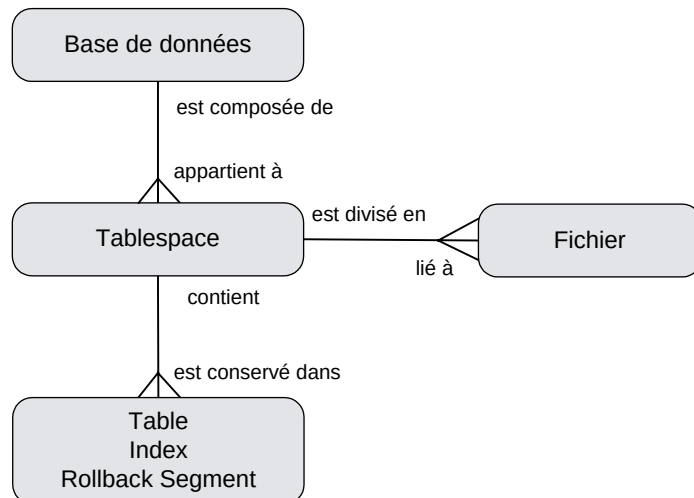


Figure 23-1
Base de données physique

À chaque tablespace sont associés un ou plusieurs fichiers. Tout objet (table, index) est placé dans un tablespace, sans précision du fichier de destination, le tablespace effectuant ce lien. La figure ci-après décrit le modèle conceptuel associé au principe du tablespace :

Figure 23-2
Schéma conceptuel du principe des tablespaces



Lorsque vous créez une table dans un tablespace, vous ne connaissez pas le fichier du tablespace dans lequel elle se trouve. Beaucoup d'administrateurs sont troublés par cette perte de contrôle de l'emplacement des données. Pourtant, ce n'est pas important, et ce pour deux raisons : premièrement, en tant qu'administrateur de base de données, vous aurez suffisamment de travail et il est inutile de vous encombrer de détails relatifs à l'emplacement physique des données au sein d'un fichier. Deuxièmement, vous pouvez toujours créer des tablespaces n'ayant qu'un seul fichier.

À sa création, une base Oracle possède obligatoirement un tablespace appelé SYSTEM, qui contient le dictionnaire de données et le segment de rollback SYSTEM. À ce tablespace est attaché au moins un fichier base. Pour garantir un bon fonctionnement de la base, il faudra créer d'autres tablespaces.

SYSAUX est un nouveau type de tablespace nécessaire à chaque base Oracle 10g. Précédemment, le dictionnaire de données hébergé dans le tablespace SYSTEM contenait aussi toutes les structures des différents outils ou options d'Oracle (RMAN, Text, Ultra Search, Data Mining, etc.). Le dictionnaire de la base est maintenant dans SYSTEM et tous les utilitaires Oracle dans SYSAUX. Le tablespace SYSAUX est obligatoirement créé pendant la création de la base.

En fin, le tablespace utilisateur est indiqué comme support à cet exemple. Il peut y en exister un par application, plusieurs par application, etc.

La figure 23-3 résume cette architecture minimale.

Deux critères essentiels déterminent le nombre et le contenu des tablespaces : la *performance* et l'*organisation*.

Le critère de performance veille à répartir les fichiers de la base de données sur différents disques. Cette répartition vise à limiter les problèmes de contention disque.

Le critère d'organisation vise à ordonner et ranger les données en créant plusieurs tablespaces afin de :

- faciliter les opérations de maintenance : sachant que la plus petite entité cohérente de sauvegarde est le tablespace, le DBA peut en créer plusieurs. Chacun d'eux contient des tables qui nécessitent d'être sauvegardées à la même fréquence ;
- gérer des données utilisées par des applications différentes dans une seule base de données. Cela permet de démarrer un seul jeu de processus Oracle, et d'allouer une seule zone mémoire partagée SGA pour des applications différentes ;
- spécialiser des tablespaces pour des tâches techniques internes à Oracle : l'espace où seront stockés les segments d'annulation (UNDO) et celui où Oracle effectuera ses tris (TEMPORARY).

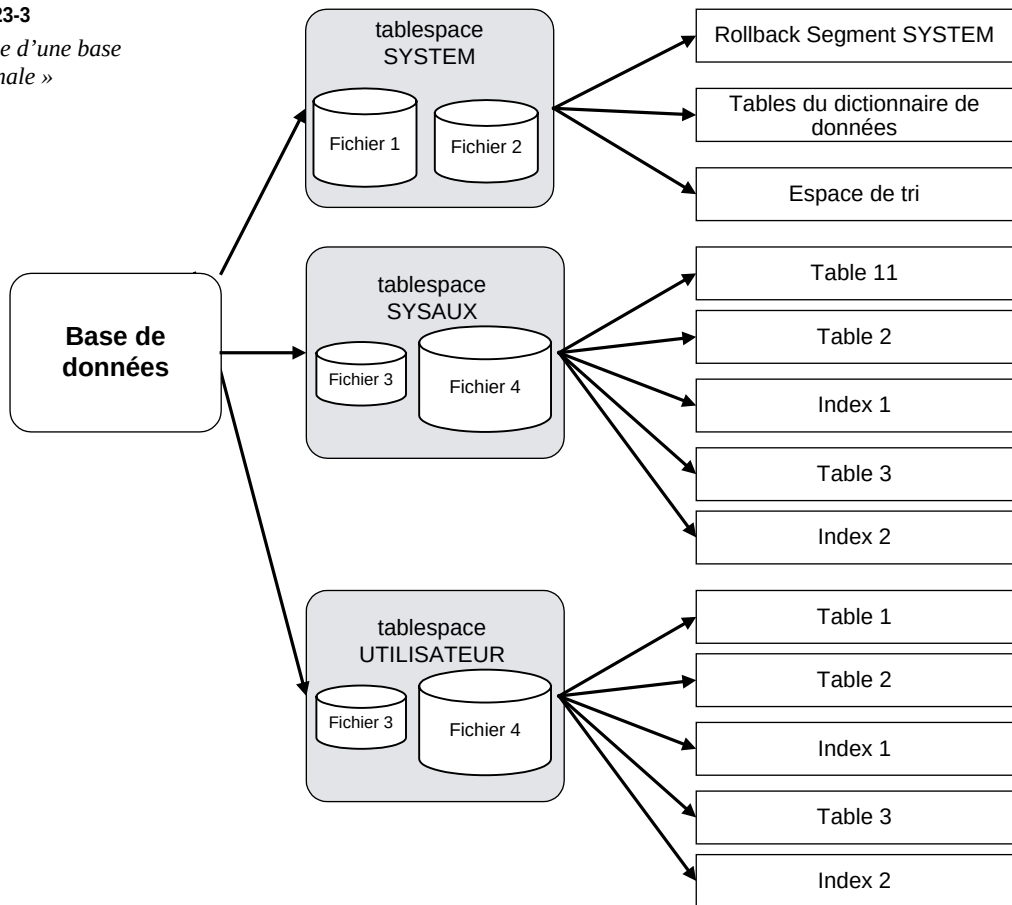
On peut ainsi se diriger vers l'architecture suivante (figure 23-4) :

Dans cet exemple, la base de données est composée :

- du tablespace SYSTEM (obligatoire) ;

Figure 23-3

Principe d'une base
« minimale »

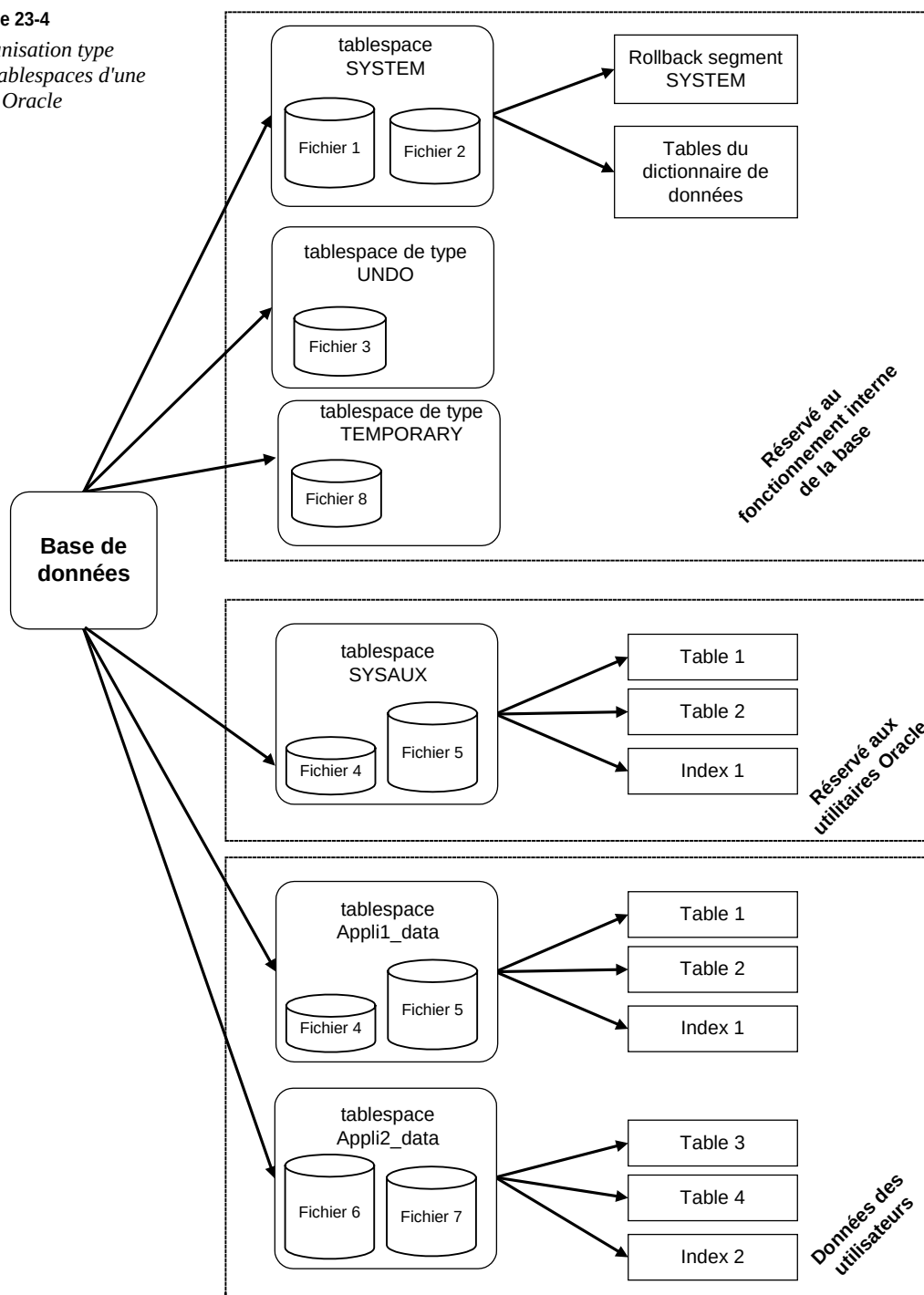


- d'un tablespace de type UNDO pour gérer les données en attente de validation ou d'annulation ;
- d'un tablespace de type TEMPORARY où Oracle disposera d'un espace disque pour effectuer des tris ;
- d'un tablespace SYSAUX réservé à la structure des données des utilitaires Oracle ;
- les deux tablespaces *Appli1_data* et *Appli2_data* sont destinés à contenir les données de deux applications différentes. On peut en imaginer d'autres ayant pour rôle de contenir les index des deux applications.

Depuis Oracle9i, vous n'avez plus besoin de gérer des rollback segments dans un tablespace spécifique. Cette gestion est automatique dans le tablespace de type UNDO. De même, le tablespace de type TEMPORARY est affecté par défaut pour la gestion interne des tris de tous les utilisateurs.

Figure 23-4

Organisation type
des tablespaces d'une
base Oracle



La gestion des tablespaces normaux, SYSAUX, UNDO et TEMPORARY, est étudiée dans ce chapitre.

Le DB_BLOCK_SIZE

Le bloc Oracle est une unité exprimée en octets qui sert d'unité d'échange entre les fichiers, la mémoire et les processus. C'est un multiple de la taille des blocs manipulés par le système Windows. La taille d'un bloc Oracle est précisée lors de la création de la base de données. Sa valeur, une fois la base créée, ne peut plus être modifiée.

Le paramètre définissant la taille du bloc Oracle est le paramètre DB_BLOCK_SIZE, contenu dans le fichier d'initialisation de la base initSID.ora.

Une fois l'instance démarrée, la valeur peut être consultée par la vue V\$PARAMETER :

```
col name for a30
col value for a30
select name, value from v$parameter
       where name = 'db_block_size' ;
```

NAME	VALUE
db_block_size	4096

Attribuer au DB_BLOCK_SIZE une taille importante permet de limiter les accès disques, car chaque opération de lecture « monte » plus de données en mémoire. En cas de modification d'un seul enregistrement situé dans un bloc, celui-ci est écrit dans son intégralité sur disque, générant ainsi une forte charge sur les entrées/sorties disque pour de faibles modifications.

En règle générale, on limitera la taille du DB_BLOCK_SIZE à 2 048 pour des applications privilégiant la mise à jour de données (transactionnelles), on l'augmentera (8 192 et au-delà) pour des applications nécessitant beaucoup de lecture de données (décisionnelles, infocentre, datawarehouse). La valeur 4 096 convient à la majorité des applications.

Pour permettre de déplacer des tablespaces de base à base, Oracle permet maintenant d'accueillir jusqu'à quatre DB_BLOCK_SIZE différents pour une base. Par défaut, tous les tablespaces ont un DB_BLOCK_SIZE identique à celui du tablespace SYSTEM sauf si cette valeur est modifiée lors de la création d'un nouveau tablespace. Cette possibilité dépasse le cadre de ce livre.

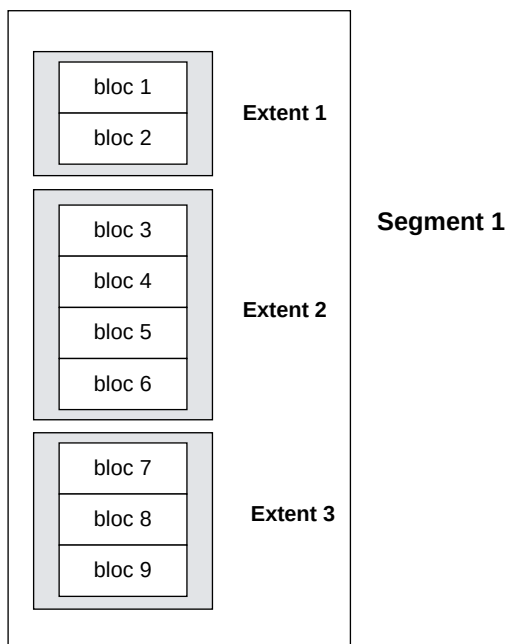
Segments, extents et blocs

Toute la gestion de l'espace disque des fichiers de données d'une base Oracle repose sur les concepts de segment, d'extent et de bloc.

Un ensemble de blocs contigus est nommé un extent (ou extension). C'est un ensemble logique. Un ensemble d'extents compose un segment. Le segment est un ensemble logique supérieur à l'extent.

Figure 23-5

Liens entre bloc, extent, segment



Dans une base Oracle 10g, il existe différents types de segments. Par exemple, l'ensemble des données d'une table est conservé dans un segment de données. De même, un segment d'index contient un index. Les autres types de segment sont utilisés pour le fonctionnement interne d'Oracle. Ce sont les segments temporaires où sont effectués les tris, les segments d'annulation utilisés pour la lecture consistante des données, et les bootstrap segments ou segments d'amorçage.

Lors de la création d'une table ou d'un index, un extent initial est alloué au segment. Si la taille de l'objet augmente, des extents sont ajoutés au segment. Ce mécanisme est la clé de toute la gestion de l'espace disque d'Oracle 10g.

Le premier bloc du premier extent contient le segment header (l'en-tête de segment). Ainsi, lors de la création de la table EMP, le premier bloc du premier extent de la table possède l'en-tête de segment. Entre autres choses, l'en-tête contient la liste de tous les extents du segment ainsi que leur taille. En effet, si les blocs d'un extent sont forcément contigus, les différents extents d'un même segment peuvent être situés n'importe où dans les fichiers du tablespaces. On comprend alors l'importance de la notion d'extent pour les performances d'une base de données. Les opérations les plus lentes d'un système étant

les lectures disque, il faut éviter de multiplier le nombre d'extents d'un segment. Cela garantit alors que tous les blocs du segment sont contigus.

Lorsque l'on conçoit une base de données, il faut être attentif à ces considérations de segment et d'extent, pour anticiper l'évolution de la base et disposer des meilleures performances possibles.

Les différents types de fichiers

Oracle peut créer des fichiers sur des systèmes de fichiers différents :

1. « classique » (NTFS sous Windows, ext3 sous Linux...);
2. « raw device » pour bénéficier de très bonnes performances d'entrées / sorties disque et pour permettre à deux serveurs d'accéder simultanément à des fichiers communs (cas des clusters) ;
3. Automatic Storage Location (ASM), pour mettre directement à disposition d'Oracle non plus des partitions, des systèmes de fichiers, mais directement des disques ! Oracle s'assure alors de toutes les opérations de bas niveau (mirroring, stripping...) et gère à 100 % l'espace alloué.

Ces points sont présentés au chapitre 27, *Configurer les disques pour Oracle*.

La gestion des tablespaces en mode LOCAL

Depuis Oracle9i, une nouvelle possibilité est apparue pour gérer les blocs et les segments. Auparavant, le paramétrage précis de la clause *storage* liée aux tablespaces, tables, index, etc. vous permettait de « piloter » Oracle pour optimiser l'allocation des blocks et des segments. Même si les paramètres de gestion d'espace étaient modifiables par la suite, c'était une opération délicate à assurer car elle nécessitait de connaître la taille des objets et de prévoir leur évolution au moment de leur création.

Maintenant, Oracle propose de gérer automatiquement tout ce fonctionnement interne. Le choix du nouveau mode de gestion ou de l'ancien est fait lors de la création du tablespace et tous les objets qu'il accueillera obéiront à ce mode de gestion. L'ancien mode s'appuie sur le dictionnaire de données, le nouveau mode sur une gestion locale au sein du tablespace.

C'est pourquoi l'ordre `CREATE TABLESPACE` a été modifié. Les options `EXTENT MANAGEMENT LOCAL` et `EXTENT MANAGEMENT DICTIONARY` sont apparues :

```
CREATE TABLESPACE test1
DATAFILE 'c:\oracle\oradata\TEST\test1_01.dbf' SIZE 20M
EXTENT MANAGEMENT LOCAL ;

CREATE TABLESPACE test2
DATAFILE 'c:\oracle\oradata\TEST\test2_01.dbf' SIZE 20M
EXTENT MANAGEMENT DICTIONARY
```

```
DEFAULT STORAGE ( INITIAL 100 K  
                    NEXT 100 K  
                    MINEXTENTS 2  
                    MAXEXTENTS 100  
                    PCTINCREASE 0 );
```

Si aucune indication n'est précisée, Oracle choisit par défaut le mode LOCAL. Seul le tablespace SYSTEM doit obligatoirement être en gestion DICTIONARY. Une fois créé, il est impossible de modifier le mode de gestion d'un tablespace.

Différences entre la gestion locale et dictionnaire

Apparue discrètement avec Oracle8i ; la gestion locale est généralisée avec Oracle 10g. Auparavant, la gestion interne d'allocation d'espace était centralisée au sein du dictionnaire de données de chaque base.

La gestion locale stocke tous les aspects d'allocation de segments à l'intérieur de chaque tablespace. Les informations sont codées au format *bitmap* dans l'en-tête de chaque tablespace. Cette évolution apporte les avantages suivants :

- La gestion locale (interne à chaque tablespace) facilite les opérations lors du déplacement de tablespaces de base à base. Les accès au dictionnaire de données sont limités.
- Le travail de l'administrateur Oracle est facilité, les fastidieuses opérations d'allocation d'espace ne sont plus nécessaires.
- La gestion centralisée dans le dictionnaire imposait des écritures dans les tables correspondantes lors de chaque allocation ou libération d'espace. L'écriture de données *bitmap* est plus rapide et évite ce goulot d'étranglement. Oracle peut supporter encore plus d'utilisateurs.
- Lors de la libération d'espace, le stockage dans les tables du dictionnaire ne permettait pas à Oracle de détecter que deux espaces libres contigus pouvaient être agrégés. Cette opération devait être effectuée par l'ordre ALTER TABLESPACE... COALESCE. La gestion locale automatise cette opération.
- Vous n'avez plus besoin de paramétrer la clause *storage* pour les tablespaces, tables et index. Cette opération est automatique en gestion locale.
- Vous pouvez maintenant déplacer un tablespace d'une base pour la raccrocher à une autre. C'est très utile pour créer très rapidement un environnement de test. La composition du tablespace étant locale, elle se déplace en même temps que le tablespace.

Les options des tablespaces en gestion locale

Ces options ne s'appliquent qu'aux tablespaces en mode EXTENT MANAGEMENT LOCAL.

Lorsqu'un tablespace local est créé, le mode de gestion d'espace pour l'allocation de nouveaux segments peut être précisé :

- **AUTOALLOCATE** : c'est l'option par défaut. Oracle gère automatiquement toutes les allocations d'espace dans le tablespace. Elles peuvent être de tailles différentes ;
- **UNIFORM SIZE** : toute allocation d'espace sera de cette taille (minimum 1 Mo). Le tablespace **TEMPORARY** utilise cette option par défaut. Le tablespace **UNDO** ne peut pas l'utiliser. Cette option est optimale en cas de fréquentes créations / suppressions de tables et d'index. Vous êtes certain que l'espace alloué ou désalloué sera utilisable par n'importe quel autre objet. Le remplissage du tablespace est optimum.

Pour chaque cas, la syntaxe utilisée est alors ;

```
CREATE TABLESPACE test1
  DATAFILE 'c:\oracle\oradata\TEST\test1_01.dbf' SIZE 20M
  EXTENT MANAGEMENT LOCAL AUTOALLOCATE;

CREATE TABLESPACE test1
  DATAFILE 'c:\oracle\oradata\TEST\test1_01.dbf' SIZE 20M
  EXTENT MANAGEMENT LOCAL UNIFORM SIZE 1M;
```

En gestion locale des tablespaces, nous vous recommandons d'utiliser l'option **UNIFORM SIZE**. Si vous voulez piloter l'allocation d'espace, utilisez des tablespaces en mode **AUTOALLOCATE**, voire **DICTIONARY** car la clause **storage** est adaptée à cet usage.

Les options précédentes concernent l'allocation d'espace, c'est-à-dire lorsque des objets sont créés ou augmentent de taille. Un autre aspect que doit gérer une base de données est la récupération d'espace libéré. C'est le cas lors de la suppression d'enregistrements dans une table et de l'impact sur ses index.

Pour un tablespace en mode local, la gestion des espaces libres peut être manuelle ou automatique. Si elle est manuelle, des seuils sont définis. Ils indiquent à Oracle quand il doit effectuer des regroupements d'espace contigus. Si elle est automatique, aucun seuil n'est à préciser.

Par défaut, Oracle propose l'ancienne option manuelle tout en recommandant fortement d'utiliser l'option automatique. D'où vient cette contradiction ? Actuellement, les données de type **LOB** ne peuvent être utilisées dans un tablespace local dont la gestion de l'espace libre est automatique. Oracle propose donc par défaut l'option la plus générale, celle qui permet de stocker tous les types de données possibles. Cette limite sera certainement levée dans une future version.

Si vous avez des doutes concernant le type de données que contiendra le tablespace, utilisez l'ancien mode de gestion manuelle car il a fait ses preuves. Si vous êtes certain qu'aucune donnée de type **LOB** ne sera dans le tablespace, Oracle recommande le mode automatique.

Dans le cas d'une gestion manuelle, les valeurs **PCTUSED**, **FREELIST**, **FREELISTS GROUPS** peuvent être précisées. Nous vous recommandons de ne pas les modifier et de laisser les valeurs par défaut. Comme cette option **MANUAL** est actuellement choisie par défaut, vous pouvez aussi tout simplement ignorer cette clause ;

```
CREATE TABLESPACE test1
  DATAFILE 'c:\oracle\oradata\TEST\test1_01.dbf' SIZE 20M
  EXTENT MANAGEMENT LOCAL AUTOALLOCATE
  SPACE MANAGEMENT MANUAL ;
```

En cas d'une gestion automatique des espaces libres, la syntaxe est alors :

```
CREATE TABLESPACE test1
  DATAFILE 'c:\oracle\oradata\TEST\test1_01.dbf' SIZE 20M
  EXTENT MANAGEMENT LOCAL AUTOALLOCATE
  SPACE MANAGEMENT AUTO ;
```

Grâce aux tablespaces à gestion locale, Oracle a supprimé la cause de nombreuses opérations de réorganisation de données. Les administrateurs Oracle ont ainsi supprimé les nuits ou les week-ends passé à effectuer ces tâches. Profitez-en !

Comment déterminer le mode de gestion des tablespaces ?

Le mode de gestion des différents tablespaces est accessible depuis la vue SYS.DBA_TABLESPACES.

```
select tablespace_name, contents, extent_management
  from dba_tablespaces
```

TABLESPACE_NAME	CONTENTS	EXTENT_MANAGEMENT
-----	-----	-----
SYSTEM	PERMANENT	DICTIONARY
UNDOTBS	UNDO	LOCAL
DRSYS	PERMANENT	LOCAL
EXAMPLE	PERMANENT	LOCAL
INDX	PERMANENT	LOCAL
TEMP	TEMPORARY	LOCAL
TOOLS	PERMANENT	LOCAL
USERS	PERMANENT	LOCAL

La gestion des objets des tablespaces en mode DICTIONARY

Ce paragraphe décrit en détail le travail que l'administrateur Oracle doit effectuer pour un tablespace créé avec le mode EXTENT MANAGEMENT DICTIONARY. Vous pouvez l'ignorer si vous décidez d'utiliser des tablespaces en gestion locale. Nous vous conseillons pourtant de le lire car il présente les actions effectuées automatiquement par Oracle avec la gestion LOCAL. C'est aussi le mode de fonctionnement obligatoire du tablespace SYSTEM.

Oracle recommande fortement de gérer les tablespaces avec le mode LOCAL, vu précédemment. Le mode DICTIONARY est obligatoire pour le tablespace SYSTEM. Ce doit être le seul dans ce cas. Nous conservons la description du mode DICTIONARY, car c'était la seule possibilité des versions précédant Oracle 9i. Vous serez à même de juger la simplicité de ce nouveau mode de gestion par rapport à l'ancien.

Création d'un tablespace en mode *DICTIONARY*

Un tablespace créé en mode de gestion dictionnaire doit contenir la clause `EXTENT MANAGEMENT DICTIONARY`.

```
CREATE TABLESPACE test2
  DATAFILE 'c:\oracle\oradata\TEST\test2_01.dbf' SIZE 20M
  EXTENT MANAGEMENT DICTIONARY
  DEFAULT STORAGE ( INITIAL 100 K
                    NEXT 100 K
                    MINEXTENTS 2
                    MAXEXTENTS 100
                    PCTINCREASE 0 );
```

La clause *storage* fixe les valeurs par défaut qui seront reprises pour tous les segments de table et d'index créés dans le tablespace.

La clause *STORAGE*

Oracle gère de façon identique les extents des segments de données et d'index. La clause de `STORAGE` permet de piloter le mécanisme des extents d'un segment.

La clause `STORAGE` peut être spécifiée lors de la création du segment ou modifiée après. Dans ce cas, elle n'agira pas sur les segments déjà créés, mais uniquement sur les segments futurs.

Par exemple, pour la création de la table `EMP`, nous pouvons utiliser la syntaxe suivante :

```
CREATE TABLE emp (
  ...
  liste des colonnes de la table...
  ...
)
STORAGE (
  INITIAL 100K
  NEXT 100 K
  PCTINCREASE 0 )
TABLESPACE user_data;
```

- `INITIAL` précise la taille de la première extension. Tout segment possède une taille initiale à sa création.
- `NEXT` précise la taille de la deuxième extension. Lorsque la première est remplie, Oracle alloue automatiquement une seconde de `NEXT` octets. Toutes les extensions futures se fondent sur cette valeur.
- `PCT_INCREASE` est un paramètre que nous positionnons toujours à zéro, mais que certaines personnes utilisent. Ce pourcentage dirige l'incrément de taille des extensions suivantes. Par exemple, `INITIAL = 1 Mo`, `NEXT = 1 Mo` et `PCT_INCREASE = 50 %`. La première extension sera de 1 Mo, la deuxième de $1\ 000 + 50\ % (1\ 000) = 1,5\ Mo$, la troisième de 2,25 Mo, etc. Calculez la taille nécessaire pour la vingtième extension !
- `TABLESPACE` indique le tablespace cible dans lequel sera créée la table `EMP`.

C'est grâce à cette clause `STORAGE` que vous pouvez piloter le comportement des segments de données et d'index.

Quelle que soit l'extension, elle ne peut être d'une taille supérieure à la plus grande place libre contiguë, au sein du tablespace cible. Même si vous disposez d'une taille libre importante au total, vous pouvez être bloqué si elle n'est pas contiguë. La vérification de l'espace libre contigu dans un tablespace est abordée dans ce chapitre, au paragraphe traitant des tablespaces.

Que se passe-t-il lorsque aucune clause `STORAGE` n'est définie ?

Dans le cas où aucune clause `STORAGE` n'est définie, Oracle lui attribue automatiquement des valeurs par défaut. En effet, un objet est toujours créé sous un compte utilisateur et, dans ses caractéristiques, un utilisateur indique nécessairement un tablespace par défaut pour créer les segments. Ce tablespace possède toujours des valeurs de `STORAGE` par défaut.

Dans l'exemple suivant, nous interrogeons la vue `SYS.DBA_USERS` pour connaître les tablespaces par défaut et les tablespaces de tri des utilisateurs.

```
select username, default_tablespace, temporary_tablespace
from sys.dba_users;
```

USERNAME	DEFAULT_TABLESPACE	TEMPORARY_TABLESPACE
SYS	SYSTEM	SYSTEM
SYSTEM	SYSTEM	TEMP
SCOTT	USERS	TEMP
PASCAL	USERS	TEMP
JEAN	USERS	TEMP
GILLES	USERS	TEMP
ISABELLE	USERS	TEMP
OEM204	OEM_REPOSITORY	TEMP

Pour connaître les détails de la clause `STORAGE` d'un tablespace particulier, il faut consulter la vue `SYS.DBA_TABLESPACES`.

```
select tablespace_name, initial_extent, next_extent,
min_extents, max_extents, pct_increase
from sys.dba_tablespaces;
```

Tablespace	Ini.Ext	Nex.Ext	Min.Ext	Mext	Pct.In
SYSTEM	10240	10240	1	121	50
USERS	10240	10240	1	121	0
RBS	10240	10240	1	121	0
TEMP	10240	10240	1	121	0
OEM_REPOSITORY	10240	10240	1	121	50

Les caractéristiques de la clause `STORAGE` existent pour chacun des tablespaces.

Comment contrôler le nombre d'extensions d'un segment ?

Nous avons vu que le segment d'une table ou d'un index peut posséder plusieurs extensions. Afin de les visualiser, utilisez la vue SYS.DBA_SEGMENTS pour une lecture globale et SYS.DBA_EXTENTS pour une vue détaillée.

Par exemple, créons une table possédant des extensions :

```
-- connexion sous l'utilisateur SCOTT
connect scott/tiger

-- création de la table TEST
create table test as select * from scott.emp ;

-- création d'enregistrements (double à chaque fois le nombre)
insert into test select * from test;

14 ligne(s) créée(s).

insert into test select * from test;

28 ligne(s) créée(s).

... continuer...

insert into test select * from test;

7168 ligne(s) créée(s).

-- validation
commit;
```

L'ordre SQL suivant visualise le nombre d'extensions de tous les segments propriété de l'utilisateur SCOTT.

```
-- visualisation des extensions sous un compte administrateur
connect system/manager

tttitle center "Place utilisée par les segments de SCOTT " skip -
        center "(tri par type de segment)" skip 2

set pagesize 150
col owner for a8
col segment_name for a17
col segment_type for a9 heading "Type Seg"
col extents for 9999 heading "Ext"
col max_extents for 99999 heading "Mext"

select segment_name,tablespace_name,
       segment_type,extents,max_extents,bytes,owner
from sys.dba_segments
where owner = 'SCOTT'
```

```
order by segment_type desc, segment_name
/
```

Place utilisée par les segments de SCOTT
(tri par type de segment)

Nom.Seg	Nom Tbs	Type Seg	Ext	Mext	Bytes	Owner
ACCOUNT	USERS	TABLE	1	121	10240	SCOTT
BONUS	USERS	TABLE	1	121	10240	SCOTT
DEPT	USERS	TABLE	1	121	10240	SCOTT
EMP	USERS	TABLE	1	121	10240	SCOTT
RECEIPT	USERS	TABLE	1	121	10240	SCOTT
SALGRADE	USERS	TABLE	1	121	10240	SCOTT
TEST	USERS	TABLE	10	121	880640	SCOTT
PK_DEPT	USERS	INDEX	1	121	10240	SCOTT
PK_EMP	USERS	INDEX	1	121	10240	SCOTT

Dans cet ordre SQL, SCOTT ne possède que des segments de type table et index. Ils possèdent tous une seule extension, sauf la table TEST, d'une taille totale de 880 Ko, qui en possède dix. Cet ordre SQL, utilisé à l'annexe 3, *Procédures pour le DBA*, permet de visualiser très rapidement les tables et index qui possèdent des extensions.

L'ordre SQL suivant fournit le détail de chacune des extensions de la table TEST.

```
tttitle center "Détail des extents du segment SCOTT.TEST" skip 2
```

```
select owner, segment_name, segment_type, tablespace_name,
       extent_id, bytes
from sys.dba_extents
where segment_name = 'TEST'
and owner = 'SCOTT'
order by extent_id
/
```

Détail des extents du segment SCOTT.TEST

Owner	Nom. Seg.	Type Seg	Tablespace	EXTENT_ID	BYTES
SCOTT	TEST	TABLE	USERS	0	10240
SCOTT	TEST	TABLE	USERS	1	10240
SCOTT	TEST	TABLE	USERS	2	20480
SCOTT	TEST	TABLE	USERS	3	30720
SCOTT	TEST	TABLE	USERS	4	40960
SCOTT	TEST	TABLE	USERS	5	61440
SCOTT	TEST	TABLE	USERS	6	92160
SCOTT	TEST	TABLE	USERS	7	133120
SCOTT	TEST	TABLE	USERS	8	194560
SCOTT	TEST	TABLE	USERS	9	286720

L'ordre SQL précédent montre le détail de chacune des dix extensions de la table TEST. Leur taille totale est de 880 Ko. La dimension de chaque extent évolue, cela signifie que le paramètre PCTINCREASE de la clause STORAGE n'est pas nul. L'ordre SQL suivant donne le détail de la clause STORAGE de la table TEST.

```

ttitle center "Paramètres de storage de SCOTT.TEST" skip 2

select owner, table_name,
       initial_extent, next_extent,
       min_extents, max_extents, pct_increase
from sys.dba_tables
where table_name = 'TEST'
and owner = 'SCOTT'
/
    
```

Paramètres de storage de SCOTT.TEST

Owner	Table	Ini. Ext.	Next.Ext.	Min. Ext.	Mext	Pct.Increase
SCOTT	TEST	10240	430080	1	121	50

On constate que le segment initial est de 10 Ko, que le PCTINCREASE est de 50 % et que la prochaine extension (la onzième) sera de 430 Ko.

Modifier les caractéristiques de la clause STORAGE

Vous pouvez modifier les caractéristiques de la clause STORAGE pour un tablespace, une table ou un index. Ainsi, si l'on souhaite que la prochaine extension de la table TEST soit de 200 Ko et que le paramètre PCTINCREASE soit à zéro, l'ordre SQL suivant permet de modifier ces caractéristiques liées à la table.

```

alter table scott.test
storage ( next 200k
        pctincrease 0);
    
```

Notez que la clause INITIAL ne peut être modifiée une fois le segment créé.

La vérification des paramètres de stockage de la table TEST indique les nouvelles valeurs.

Paramètres de storage de SCOTT.TEST

Owner	Table	Ini. Ext.	Next.Ext.	Min. Ext.	Mext	Pct.Increase
SCOTT	TEST	10240	200080	1	121	0

Comme évoqué précédemment, la modification des clauses de storage n'affecte pas les extensions existantes, elle influe uniquement sur les futures extensions.

La fragmentation est-elle pénalisante ?

Oui, la fragmentation est pénalisante. En effet, les extensions sont disséminées au sein du tablespace et les déplacements du disque dur sont beaucoup plus fréquents que dans le cas où les extensions sont contiguës. Il faut toutefois rapprocher la taille du segment du nombre d'extensions. Une table de 100 Mo possédant cinq extensions est moins pénalisante qu'un index d'1 Mo doté lui aussi de cinq extensions, car les index ont pour but d'accélérer la recherche des données.

Heureusement, les nouvelles options proposées par Oracle apportent beaucoup de souplesse et évitent beaucoup de travail aux administrateurs.

Défragmenter une table

Avant toute opération de défragmentation, effectuez une sauvegarde de la base. Même s'il s'agit d'une opération mineure, cela assure une sécurité parfaite, quelle que soit la situation rencontrée. Ces opérations de maintenance doivent toujours être effectuées offline, sans utilisateur actif sur la base.

Les techniques utilisées dépendent du type de segment. Pour une table, vous pouvez effectuer les tâches suivantes :

1. Renommer la table.
2. Créer une nouvelle table à partir de la table renommée.

Par exemple, pour notre table TEST :

```
-- connexion sous l'utilisateur SCOTT
connect scott/tiger

-- renommer la table TEST
rename TEST to TEST_OLD ;

-- créer une nouvelle table avec une clause STORAGE
-- les données provenant de TEST_OLD sont insérées dans TEST.
-- La structure de la table TEST est issue de TEST_OLD.

create table TEST
  storage (initial 800K
           next 100K
           pctincrease 0 )
as select * from TEST_OLD
/
```

Visualisation des extensions sous un compte administrateur :

```
connect system/manager

tttitle center "Place utilisée par les segments de SCOTT " skip -
        center "(tri par type de segment)" skip 2
```

```

set pagesize 150
col owner for a6 heading "Owner"
col tablespace_name for a10 heading "Tablespace"
col table_name for a10 heading "Table"
col segment_name for a8 heading "Segment"
col segment_type for a9 heading "Type Seg"
col extents for 999 heading "Ext"
col max_extents for 999 heading "Mext"

select segment_name,tablespace_name,segment_type,
       extents,max_extents,bytes,owner
  from sys.dba_segments
     where owner = 'SCOTT'
     order by segment_Type desc, segment_name
/

```

Place utilisée par les segments de SCOTT
(tri par type de segment)

Segment	Tablespace	Type Seg	Ext	Mext	BYTES	Owner
...						
TEST	USERS	TABLE	1	121	819200	SCOTT
TEST_OLD	USERS	TABLE	10	121	880640	SCOTT
...						

La table TEST_OLD a toujours dix extensions alors que TEST n'en a qu'une. Visualisons maintenant les caractéristiques de storage de la nouvelle table TEST :

```

tttitle center "Paramètres de storage de SCOTT.TEST" skip 2
col initial_extent for 999999 heading "Initial"
col next_extent   for 999999 heading "Next"
col min_extents   for 999999 heading "Min.Ext"
col max_extents   for 999999 heading "Max.Ext"
col pct_increase  for 999999 heading "Pct.Incr"

select owner, table_name,
       initial_extent, next_extent,
       min_extents, max_extents, pct_increase
  from sys.dba_tables
     where table_name = 'TEST'
     and owner = 'SCOTT'
/

```

Paramètres de storage de SCOTT.TEST

Owner	Table	Initial	Next	Min.Ext	Max.Ext	Pct.Incr
SCOTT	TEST	819200	102400	1	121	0

Cette première méthode est utilisable si la table concernée n'est pas trop volumineuse et si la base est d'une taille suffisante pour héberger à la fois l'original et une copie.

La deuxième technique pour défragmenter une table consiste à recourir à l'Export/Import. Elle est abordée au chapitre 18, *Les outils d'Oracle 10g*.

Défragmenter un index

Pour défragmenter un index, vous pouvez le détruire puis de le créer à nouveau. Comme pour une table, vous pouvez utiliser l'Export/Import, traité au chapitre 18, *Les outils d'Oracle 10g*.

Vous pouvez aussi utiliser des ordres SQL de reconstruction comme ALTER INDEX REBUILD ou ALTER INDEX COALESCE.

Gestion des tablespaces et des fichiers de données

En complément des présentations théoriques du chapitre 9, *Les fichiers d'une base Oracle 10g*, et du paragraphe précédent, nous présentons maintenant les outils de manipulation des tablespaces des fichiers de données associés, utilisés aussi pour dimensionner correctement les différents objets.

Le mode de gestion des extensions LOCAL est proposé par défaut par Oracle. Nous l'utiliserons donc dans tous les exemples de ce paragraphe.

Avant toute action, effectuez une sauvegarde de la base et entraînez-vous sur une base test, sans utilisateur actif.

Création d'un tablespace

La création d'un nouveau tablespace implique d'y associer obligatoirement un fichier de données en précisant sa taille. La commande CREATE TABLESPACE permet d'établir un nouveau tablespace avec ses propres paramètres de stockage.

```
create tablespace user_data
datafile 'c:\oracle\oradata\TEST\user01.dbf' size 50M ,
        'D:\oracle\oradata\TEST\user02.dbf' size 100M
extent management local ;
```

Dans cet exemple, le tablespace est créé avec deux fichiers situés sur des disques différents.

La gestion automatique des fichiers proposée par Oracle 10g limite les informations à donner. Cette possibilité est abordée en fin de chapitre.

Définir un tablespace par défaut

« Ne créez jamais d'objets dans le tablespace SYSTEM ! » C'est l'une des premières recommandations faites à tout administrateur Oracle. Par analogie, il ne faut jamais créer

d'objets « utilisateurs » sous le compte administrateur d'un serveur. Le tablespace SYSTEM contient le dictionnaire de données ainsi que d'autres objets nécessaires à Oracle. Pour éviter de placer malencontreusement des objets dans le tablespace SYSTEM, les administrateurs devaient tout d'abord créer un nouveau tablespace pour les données utilisateurs puis créer les utilisateurs :

```
CREATE USER nouvel_utilisateur  
IDENTIFIED BY son_mot_de_passe  
DEFAULT TABLESPACE tablespace_utilisateur  
QUOTA LIMITED ON tablespace_utilisateur ;
```

Malheureusement, il était trop facile d'oublier d'indiquer un tablespace par défaut lorsqu'on créait un nouvel utilisateur et, en absence d'information, Oracle plaçait par défaut les objets de cet utilisateur dans le tablespace SYSTEM. Oracle 10g apporte la notion de tablespace par défaut pour toute la base. Il suffit de l'indiquer une seule fois au plus haut niveau :

```
ALTER DATABASE  
DEFAULT TABLESPACE tablespace_utilisateur;
```

Tout nouvel utilisateur créé qui n'aura pas de tablespace par défaut utilisera alors celui précisé au plus haut niveau. C'est un avantage qui limite les erreurs lors de la création d'utilisateurs.

Les tablespaces temporaires

Oracle effectue de nombreux tris à la demande des utilisateurs ou pour son fonctionnement interne. Ils sont principalement effectués en mémoire, dans la zone SORT_AREA_SIZE de la SGA. Si cette zone s'avère trop petite, un espace disque est réservé à cet effet : c'est le tablespace temporaire ou TEMPORARY TABLESPACE. De même qu'il existe des segments de données et d'index, le segment de tri est un *sort segment*.

Si un tablespace temporaire n'est pas créé, les tris sont effectués dans le tablespace SYSTEM qui contient le dictionnaire. Cela n'est pas son rôle et il convient de toujours créer un tablespace temporaire.

Une fois créé, un tablespace temporaire ne peut pas être modifié pour contenir des données. Par contre, comme pour tous les autres tablespaces, on peut le mettre en ligne, hors ligne et lui ajouter des fichiers.

Un tablespace temporaire peut être géré en mode local ou dictionnaire. Oracle recommande fortement de le gérer en mode local. La syntaxe utilisée est la suivante :

```
create temporary tablespace temp_data  
tempfile 'c:\oracle\oradata\TEST\temp_data01.dbf' size 50M  
extent management local ;
```

Remarquez qu'un tablespace temporaire n'utilise pas comme fichier un *datafile* mais un *tempfile*.

Une fois le tablespace temporaire créé, Oracle l'utilisera pour tous les tris des utilisateurs, sans qu'il soit nécessaire d'indiquer pour chaque utilisateur quel est son tablespace de tri.

Les vues V\$TEMPFILE et DBA_TEMP_FILES permettent d'accéder à toutes les caractéristiques des tablespaces temporaires.

Avant Oracle9i, pour utiliser un tablespace temporaire, vous deviez le préciser pour chaque utilisateur :

```
CREATE USER nouvel_utilisateur  
IDENTIFIED BY son_mot_de_passe  
TEMPORARY TABLESPACE temp;
```

Si vous oubliez d'indiquer la clause TEMPORARY TABLESPACE lors de la création d'un utilisateur, le tablespace SYSTEM était utilisé pour toutes les actions de tri, de groupement, de jointure et autres activités temporaires sur disque.

Cela occasionnait alors une activité disque supplémentaire sur le tablespace SYSTEM qu'il convenait d'éviter. Ces opérations de tri s'ajoutaient alors à l'activité propre du tablespace SYSTEM et pouvaient ainsi créer des points de contention sur les accès disque. Depuis Oracle9i, vous pouvez indiquer un tablespace par défaut pour toute la base de données :

```
ALTER DATABASE DEFAULT TEMPORARY TABLESPACE temp;
```

Après avoir exécuté cette commande, tous les utilisateurs créés **après** utiliseront le tablespace temporaire indiqué :

```
ALTER USER toto DEFAULT TEMPORARY TABLESPACE temp;
```

Si des utilisateurs ont été créés, vous pouvez corriger cela simplement en créant automatiquement une série d'ordres SQL :

```
select 'ALTER USER '||username||' DEFAULT TEMPORARY TABLESPACE temp;'
from DBA_USERS ;
```

L'accès simultané de tous les utilisateurs à cet espace de tri pouvant créer un point de contention disque, Oracle permet de répartir les accès en utilisant la notion de tablespaces temporaires groupés.

Les tablespaces UNDO ou d'annulation

Un tablespace de type UNDO (Oracle9i) est un nouveau type de tablespaces destiné à remplacer l'ancien mode de gestion des segments d'annulation ou rollback segments. C'est dans cet espace qu'Oracle gère les données en attente de validation ou d'annulation. L'utilisation d'un tablespace UNDO est beaucoup plus simple que l'ancien mode. Ils sont tous deux abordés plus loin dans ce chapitre.

Les tablespaces BIGFILES

Les tablespaces BIGFILES permettent d'utiliser les possibilités des systèmes 64 bits de gérer de très gros fichiers. Un tablespace BIGFILE ne contient qu'un seul fichier qui peut comporter jusqu'à 2^{32} blocs Oracle, soit environ 8 exaoctets ou à peu près 8 millions de teraoctets...

On peut s'interroger sur l'intérêt de créer de tels fichiers. Leur taille gigantesque dépasse les possibilités techniques actuelles. Imaginez-vous comment sauvegarder un fichier d'une taille dépassant celle de tous les médias de sauvegarde ? Ce type d'annonce ressemble à une annonce marketing. La taille d'un fichier : *no limit* !

Modification d'un tablespace

La commande ALTER permet de modifier les caractéristiques d'un tablespace existant.

```
alter tablespace user_data online ;
```

Renommer un tablespace

Vous l'avez longtemps attendu, c'est maintenant chose faite ! Vous pouvez renommer vos tablespaces en une seule commande, à l'exclusion des tablespaces SYSTEM et SYSAUX réservés par Oracle.

```
alter tablespace tbs_test rename tbs_production ;
```

Les règles de nommage de fichiers recommandent que le nom du fichier indique le tablespace auquel il se rattache. En changeant le nom du tablespace, vous perdez cette indication.

C'est très intéressant lors du transport d'un tablespace d'une base à une autre.

Autre point, vous devez sauvegarder votre fichier de contrôle sitôt la modification faite, le nom du tablespace et des fichiers attachés y figurant. C'est important pour la validité de vos sauvegardes.

Ajouter un fichier de données

Il existe deux possibilités : la première consiste à créer un nouveau tablespace et à lui associer le nouveau fichier, la seconde à ajouter un nouveau fichier à un tablespace existant.

```
alter tablespace user_data
  add datafile 'D:\oracle\oradata\TEST\user02.dbf' size 50M;

alter temporary tablespace temp_data
  add tempfile 'c:\oracle\oradata\TEST\temp_data02.dbf' size 50M

alter tablespace undotbs
  add datafile 'c:\oracle\oradata\TEST\undotbs_02.dbf' size 50M
```

Il est possible d'ajouter un fichier à un tablespace mais il est impossible de supprimer un seul fichier d'un tablespace. Il faut alors supprimer le tablespace entier. En effet, dans un tablespace, une table importante peut être contenue dans plusieurs fichiers. Supprimer l'un des fichiers conduirait à un résultat imprévisible.

Compacter et réduire les segments en ligne

Le Segment Advisor (Oracle 10g) indique la liste des segments à réorganiser pour récupérer de la place dans les tablespaces. Il examine la fragmentation des objets ainsi qu'une estimation du taux de remplissage prévu. Il est accessible depuis Oracle Enterprise Manager (OEM) : *Centre de conseil* puis *Fonctions de conseils sur les segments*.

Une fois les segments à réorganiser détectés, la clause SQL SHRINK de récupération d'espace peut être lancée manuellement ou automatiquement depuis OEM. Elle optimise le stockage lorsque l'espace libre dans les blocs est trop important. C'est très utile pour :

- des tables qui font l'objet de beaucoup d'insertions / suppressions de lignes ;
- des tables utilisées temporairement dont l'espace libéré doit être désalloué.

Le compactage d'une table nécessite que le déplacement d'une table soit autorisé :

```
ALTER TABLE nom_table ENABLE ROW MOVEMENT ;
```

La syntaxe SQL utilisée est :

```
ALTER [TABLE | INDEX] nom_table_index SHRINK SPACE [COMPACT][CASCADE]
```

Cette option de récupération d'espace s'effectue en ligne et n'est utilisable que pour des tablespaces dont la gestion d'espace est en mode AUTO.

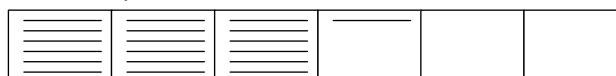
Figure 23-6

Utilisation du compactage d'une table ou d'un index

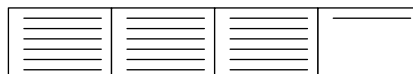
Situation avant SHRINK



Situation après ALTER TABLE ... SHRINK SPACE COMPACT



Situation après ALTER TABLE ... SHRINK SPACE



Sans préciser d'option, les données sont compactées dans les segments des tables et des index et l'espace libre est désalloué.

L'option COMPACT permet de compacter les données dans les segments mais sans désallouer l'espace occupé.

L'option CASCADE permet d'effectuer le compactage pour tous les segments dépendants (index, partitions de tables et d'index, vues matérialisées).

Certaines restrictions existent, comme l'impossibilité d'utiliser le SHRINK pour des tables contenant des données binaires de type Large Objects (LOB), LONG ou LONG ROW.

Extension automatique d'un fichier de données

Auparavant, l'un des problèmes les plus difficiles à résoudre pour les administrateurs Oracle était la gestion de la taille des tablespaces. Que faire lorsqu'un segment (de données, d'index, de tri, de rollback) cherche à s'étendre dans un tablespace et ne dispose plus de place dans les fichiers pour y créer un nouvel extent ? L'utilisateur qui lançait l'opération se heurtait alors à un blocage.

La principale difficulté était de prévoir le segment qui aurait besoin de place, ce qui était pratiquement impossible pour les segments de tri ou de rollback, car on ne peut prévoir a priori la place nécessaire à tous les utilisateurs qui exécutent simultanément des ordres SQL. La seule solution était de « voir large » et de dimensionner les fichiers de sorte que les tablespaces aient toujours beaucoup d'espace disponible. Le blocage était ainsi limité, au prix d'une surconsommation de l'espace disque.

Oracle permet que les fichiers des tablespaces augmentent automatiquement de taille :

```
alter database
  datafile 'c:\oracle\oradata\TEST\user01.dbf'
  autoextend on
  next 25M
  maxsize 2000M;
```

Dans cet exemple, la taille du fichier c:\oracle\oradata\TEST\user01.dbf peut augmenter automatiquement, par palier de 25 Mo jusqu'à la limite maximale de 2 000 Mo. Observez que cette option ne s'applique qu'à des fichiers et non au tablespace auquel appartient ce fichier, ce qui est tout à fait normal, car suivant l'emplacement physique des différents fichiers d'un tablespace, leurs points de montage ne disposent pas tous du même espace.

L'augmentation automatique de la taille des fichiers est possible pour tous les types de tablespaces (données, index, temporaire, UNDO, SYSTEM) qu'ils soient en mode de gestion locale ou dictionnaire.

```
create temporary tablespace temp_data
  tempfile 'c:\oracle\oradata\TEST\temp_data01.dbf' size 50M
  autoextend on
  next 20M
  maxsize 200M
  extent management local ;

create tablespace user_data
  datafile 'c:\oracle\oradata\TEST\user_data01.dbf' size 50M
  autoextend on
  next 100M
  maxsize 2000M
  extent management local ;

create tablespace undotbs_2
  datafile 'D:\oracle\oradata\TEST\undotbs_2_01.dbf' size 50M
  autoextend on
  next 100M
  maxsize 200M
  extent management local ;
```

Il est indispensable de fixer une taille limite pour que les fichiers ne puissent dépasser vos capacités de sauvegarde.

La vue SYS.DBA_DATA_FILES permet de contrôler les caractéristiques d'extension des fichiers.

```
select tablespace_name, file_name, autoextensible,
       maxbytes, increment_by
from dba_data_files
order by tablespace_name;
```

Tablespace	FILE_NAME	AUTOEXTENS	MAXBYTES	INCREMENT_BY
INDX	c:\oradata\TEST\INDX01.dbf	YES	157286400	2560
OEM_REPOSITc:	c:\oradata\TEST\OEMREP01.dbf	NO	157286400	2560
RBS	c:\oradata\TEST\RBS01.dbf	YES	157286400	2560
SYSTEM	c:\oradata\TEST\SYSTEM01.dbf	YES	524288000	5120
TEMP	c:\oradata\TEST\TEMP01.dbf	YES	157286400	2560
USERS	c:\oradata\TEST\USERS01.dbf	YES	157286400	2560

Cet exemple montre que seul le tablespace OEM_REPOSITORY n'est pas en mode auto-extensible.

Les alertes d'utilisation des tablespaces

Vous pouvez maintenant définir des seuils permettant d'être automatiquement alerté en cas de dépassement. Pour les tablespaces, la signification des seuils est différente suivant le type de tablespace :

- UNDO : espace utilisé contenant des données en attente de validation ou d'annulation ;
- temporaire : espace utilisé par les sessions (tris, etc.) ;
- fichiers auto-extensibles : à l'approche de la taille maximale du fichier, dépendant de l'espace disponible sur le disque ou encore du système d'exploitation.

L'interface graphique d'OEM est bien utile pour piloter la syntaxe SQL. Par exemple, pour définir un seuil sur le tablespace utilisateur TBS_USERS, la syntaxe SQL utilisée est :

```
begin
DBMS_SERVER_ALERT.SET_THRESHOLD(9000,4,'90','98',1,1,NULL ,5,'TBS_USERS') ;
end ;
/
```

Estimation de la taille d'un nouveau segment

Avec Oracle Enterprise Manager, lors de la création d'une table ou d'un index, vous disposez d'un utilitaire estimant l'espace nécessaire à partir du nombre de lignes que contiendra la table.

Liens entre fichiers et tablespaces

Pour connaître le lien entre les tablespaces et les fichiers, utilisez la vue SYS.DBA_DATA_FILES du dictionnaire de données :

```
desc sys.dba_data_files
Nom                                Type
-----
FILE_NAME                         VARCHAR2(513)
FILE_ID                           NUMBER
TABLESPACE_NAME                   VARCHAR2(30)
BYTES                             NUMBER
BLOCKS                            NUMBER
STATUS                            VARCHAR2(9)
RELATIVE_FNO                      NUMBER
AUTOEXTENSIBLE                    VARCHAR2(3)
MAXBYTES                          NUMBER
MAXBLOCKS                        NUMBER
INCREMENT_BY                      NUMBER
USER_BYTES                        NUMBER
USER_BLOCKS                      NUMBER

col file_name format a40
select file_name, tablespace_name, bytes, auto-extensible
       from dba_data_files ;
```

FILE_NAME	TABLESPACE_NAME	BYTES	AUT
c:\oracle\oradata\TEST\USERS01.DBF	USERS_DATA	118489088	YES
c:\oracle\oradata\TEST\USERS02.DBF	USERS_DATA	8489088	YES
c:\oracle\oradata\TEST\OEMREP01.DBF	SYS_AUX	5242880	YES
c:\oracle\oradata\TEST\INDX01.DBF	INDX	2097152	YES
c:\oracle\oradata\TEST\RBS01.DBF	RBS	26214400	YES
c:\oracle\oradata\TEST\TEMP01.DBF	TEMP	2097152	YES
c:\oracle\oradata\TEST\SYSTEM01.DBF	SYSTEM	146800640	YES

Dans l'exemple précédent, seul le tablespace USERS_DATA possède deux fichiers. La taille totale de ce tablespace correspond à la somme de ces deux fichiers. La vue SYS.DBA_TABLESPACES permet de connaître la taille totale de tous les tablespaces.

Caractéristiques des tablespaces et fichiers associés

Pour connaître les caractéristiques de chaque tablespace, les vues utiles du dictionnaire de données sont :

- SYS.DBA_DATA_FILES : détail des fichiers de données :
- SYS.DBA_TEMP_FILES : détail des fichiers des tablespaces temporaires :
- SYS.DBA_TABLESPACES : détail des tablespaces :
- SYS.DBA_FREE_SPACE : espace disponible restant dans les tablespaces.

Par exemple, les ordres SQL suivants permettent de connaître la taille totale des tablespaces, puis l'espace disponible restant dans les tablespaces.

```
tttitle center "Taille totale des tablespaces" skip 2
col tablespace_name for a16 heading "Tablespace"
```

```
select tablespace_name, sum(bytes)/1024 "Place totale en Ko"
  from sys.dba_data_files
  group by tablespace_name
  order by tablespace_name
/
```

Taille totale des tablespaces

Tablespace	Place totale en Ko
INDX	2048
OEM_REPOSITORY	5120
RBS	25600
SYSTEM	143360
TEMP	2048
USERS	115712

```
tttitle center "Espace disponible dans les tablespaces" skip 2
col TBS for a15
select tablespace_name, sum(bytes)/1024 "Espace disponible en Ko"
  from sys.dba_free_space
  group by tablespace_name
/
```

Place libre dans les tablespaces

Tablespace	Espace disponible en Ko
INDX	2046
OEM_REPOSITORY	3268
RBS	13698
SYSTEM	5670
TEMP	1976
USERS	114200

L'ordre SQL ci-dessous, plus complexe, reprend les éléments issus des deux ordres SQL précédents. Il permet d'afficher, tablespace par tablespace, la taille totale, l'espace disponible et le pourcentage d'occupation au sein du tablespace. C'est ce dernier ordre qui figure à l'annexe 3, *Procédures pour le DBA*. La syntaxe utilisée pour écrire cet ordre montre que le SQL peut être employé pour des requêtes très évoluées.

Oracle Enterprise Manager vous permet d'accéder à ces informations sans la lourdeur d'une requête SQL dont il est impossible de se souvenir !

```

tttitle center "Taille et occupation des tablespaces" skip 2
col "% occupé" for 99.9

select A.tablespace_name,
       A.total_size "Taille totale (Ko)",
       B.free_size  " Espace disponible(Ko)",
       ((A.total_size - B.free_size)*100 / A.total_size) "% occupé"
from
  (select tablespace_name,
          sum(bytes)/1024 total_size
   from sys.dba_data_files
   group by tablespace_name) A ,
  (select tablespace_name,
          sum(bytes)/1024 free_size
   from sys.dba_free_space
   group by tablespace_name) B
where a.tablespace_name = b.tablespace_name
/

```

Taille et occupation des tablespaces

Tablespace	Taille totale (Ko)	Espace disponible(Ko)	% occupé
INDX	2048	2046	.1
OEM_REPOSITO	5120	3268	36.2
RBS	25600	13698	46.5
SYSTEM	143360	5670	96.0
TEMP	2048	1976	3.5
USERS	115712	114200	1.3

Le dernier ordre SQL est très intéressant ; il permet de visualiser en un seul coup d'œil l'occupation de vos tablespaces.

Positionner un tablespace online/offline

Un tablespace peut être online (actif, accessible à tous les utilisateurs) ou offline (inactif, inaccessible aux utilisateurs, même administrateur).

Mettre un tablespace offline permet de sauvegarder les fichiers qui le composent.

```

alter tablespaces users offline ;

-- sauvegarder les fichiers
-- qui composent le tablespace.

alter tablespaces users online ;

```

Sauf dans le cas d'une base en mode ARCHIVELOG, ne sauvegardez jamais les fichiers lorsque la base est ouverte et les tablespaces online ; votre sauvegarde n'aurait aucune valeur. En effet, entre le début et la fin de votre sauvegarde, le processus Database Writer d'Oracle continue à écrire dans le fichier.

Déplacer/renommer les fichiers d'un tablespace

La marche à suivre est la suivante :

1. Positionnez le tablespace concerné offline ;

```
alter tablespace users offline;
```

Espace de tables (TABLESPACE) modifié.

2. Copiez/renommez sous Windows les anciens fichiers dans leur nouvelle destination.

3. Indiquez au dictionnaire de la base ce changement d'emplacement :

```
alter tablespace users rename datafile  
    'c:\oracle\oradata\TEST\user_old.dbf'  
to 'c:\oracle\oradata\TEST\user_new.dbf' ;
```

4. Mettez le tablespace concerné online :

```
alter tablespace users online;
```

Espace de tables (TABLESPACE) modifié.

5. Vérifiez par la vue DBA_DATA_FILES la prise en compte des modifications, puis supprimez sous Windows l'ancien fichier.

Fragmentation des tablespaces

Les nouvelles options d'Oracle 10g limitent l'intérêt de ce type de mesure. La fragmentation existe, Oracle la gère maintenant très bien !

Les tablespaces contiennent des segments constitués de plusieurs extensions. À ce titre, l'espace disponible au sein d'un tablespace est morcelé. L'ordre SQL suivant permet de mesurer cette fragmentation.

```
tttitle center "Espace disponible en nombre de blocs contigus" skip -  
        center " (1 bloc vaut db_block_size octets)"          skip 2
```

```
select tablespace_name,  
       round(avg(blocks)) "En Moyenne",  
       min(blocks) "Minimum",  
       max(blocks) "Maximum",  
       count(*) "Nombre"  
from sys.dba_free_space  
group by tablespace_name  
/
```

```
Espace disponible en nombre de blocs contigus  
(1 bloc vaut db_block_size octets)
```


TABSPACE_NAME	En Moyenne	Minimum	Maximum	Nombre
INDX	1023	1023	1023	1
OEM_REPOSITORY	1634	1634	1634	1
RBS	381	125	1474	18
SYSTEM	118	5	1485	24
TEMP	1023	1023	1023	1
USERS	18757	400	55440	3

Dans cet exemple, l'espace disponible au sein du tablespace est exprimé en nombre de blocs.

Cet ordre SQL présente la plus importante extension contiguë que votre tablespace acceptera de créer, avant d'augmenter la taille d'un des fichiers qui le compose.

Modifier la taille d'un fichier

Nous avons vu que la taille d'un tablespace pouvait augmenter automatiquement par la clause `AUTOEXTEND ON`. Vous pouvez aussi augmenter manuellement la taille d'un fichier :

```
alter database datafile 'c:\oracle\oradata\TEST\user_01.dbf'  
  resize 100M
```

Suivant la dimension actuelle du fichier, cette opération en augmente ou diminue sa taille. Dans le cas d'une diminution, cette opération est parfois impossible. Dans ce cas, tentez de compacter les tables et index contenus (voir précédemment).

Supprimer un tablespace

C'est une opération simple mais dangereuse ! Avant toute suppression d'un tablespace, assurez-vous de disposer d'une sauvegarde complète de la base. Pour supprimer un tablespace ne contenant aucun segment de données ou d'index, l'ordre `DROP TABLESPACE` suffit :

```
drop tablespace test ;
```

Si le tablespace contient des tables et des index, vous devez le préciser sinon l'ordre SQL précédent échoue.

```
drop tablespace test including contents ;
```

Si des contraintes d'intégrité lient les données de ce tablespace avec d'autres tablespaces, l'ordre `CASCADE CONSTRAINTS` doit être indiqué.

Enfin, vous pouvez lier la suppression des fichiers composant le tablespace à celui du tablespace :

```
drop tablespace test including contents and datafiles ;
```

Les tablespaces transportables

Cette caractéristique permet de transférer un tablespace d'une base de données Oracle 10g vers une autre base de données Oracle 10g. Les utilisations sont multiples : déplacer l'ensemble des données d'une application d'une base à une autre, créer rapidement un environnement de test identique à celui de production, transférer des tablespaces d'une base de production vers une base infocentre. Comme les tablespaces transportables peuvent provenir de bases configurées avec des DB_BLOCK_SIZE différents, Oracle accepte maintenant des tablespaces possédant une valeur différente de celle définie pour son tablespace SYSTEM. L'utilisation des tablespaces transportables est limitée entre bases Oracle 10g.

Comparées aux moyens antérieurs, tel l'Export/Import ou le chargement de données via SQL*Loader, les performances sont exceptionnelles. Lorsque l'on transporte un tablespace, les données et les index sont également déplacés, ce qui évite tout chargement de données et toute construction d'index. Maintenant, il suffit de copier des fichiers et d'intégrer les méta-données du tablespace dans la base cible.

Les tablespaces READ-ONLY

Vous pouvez utiliser des fichiers de données écrits sur des médias non réinscriptibles, comme des disques optiques ou des CD-Rom. Pour cela, vous pouvez utiliser des tablespaces en mode READ-ONLY.

Il est en revanche impossible d'avoir tous les fichiers d'une base de données dans ce format. Le tablespace SYSTEM, les fichiers de contrôle, les fichiers redo-log, les tablespaces de tri doivent être placés sur des disques « normaux ».

Gestion des tablespaces d'annulation

Un tablespace de type UNDO est un nouveau type de tablespace destiné à remplacer l'ancien mode, les rollback segments. C'est dans cet espace que sont gérées par Oracle les données en attente de validation ou d'annulation. L'utilisation d'un tablespace UNDO est beaucoup plus simple que celle des rollback segments. Ils sont tous deux abordés dans ce paragraphe.

Pour la gestion interne du dictionnaire, Oracle utilise toujours un rollback segment SYSTEM, quel que soit le mode de gestion des annulations choisi. Aucune opération n'est nécessaire sur cet objet.

Les nouveautés des tablespaces UNDO

Les tablespaces de type UNDO apportent des fonctionnalités nouvelles très intéressantes :

- la gestion est simplifiée par rapport à celle qu'exigeaient les rollback segments ;
- des erreurs complexes à résoudre concernant la durée de rétention des informations ne sont plus rencontrées ;

- le *Flashback Query* permet d'interroger le contenu de la base de données tel qu'il était plusieurs heures auparavant.

La création d'un tablespace UNDO

Le choix entre un tablespace de type UNDO ou les rollback segments doit être fait lors de la création de la base car il est difficile de passer d'un mode à l'autre. Les deux modes ne peuvent pas fonctionner simultanément.

Un tablespace de type UNDO peut être créé en même temps que la base ou après.

```
CREATE DATABASE TEST
MAXINSTANCES 1
MAXLOGHISTORY 1
MAXLOGFILES 5
MAXLOGMEMBERS 5
MAXDATAFILES 100
DATAFILE 'c:\oracle\oradata\TEST\system01.dbf' SIZE 100M REUSE
        AUTOEXTEND ON
        NEXT 10240K
        MAXSIZE 1000M
UNDO TABLESPACE "UNDOTBS"
DATAFILE 'c:\oracle\oradata\TEST\undotbs01.dbf' SIZE 50M,
        'D:\oracle\oradata\TEST\undotbs02.dbf' SIZE 50M REUSE
        AUTOEXTEND ON
        NEXT 5120K
        MAXSIZE 1000M
CHARACTER SET WE8ISO8859P15
NATIONAL CHARACTER SET AL16UTF16
LOGFILE GROUP 1 ('c:\oracle\oradata\TEST\redo01.log') SIZE 100M,
        GROUP 2 ('c:\oracle\oradata\TEST\redo02.log') SIZE 100M,
        GROUP 3 ('c:\oracle\oradata\TEST\redo03.log') SIZE 100M
;
```

Avant de lancer cette création, le fichier d'initialisation de la base doit préciser le mode de gestion choisi. Le paramètre UNDO_MANAGEMENT accepte les valeurs :

- UNDO_MANAGEMENT=AUTO : la gestion par tablespace UNDO est retenue. Le paramètre UNDO_TABLESPACE doit alors indiquer quel tablespace de type UNDO il doit utiliser ;
- UNDO_MANAGEMENT=MANUAL : la gestion par rollback segment est retenue.

Pour l'exemple précédent, les paramètres figurant dans le fichier d'initialisation persistant ou non seraient :

```
UNDO_MANAGEMENT=AUTO
UNDO_TABLESPACE=UNDOTBS
```

Les paramètres du fichier d'initialisation peuvent être interrogés par la vue V\$PARAMETER.

La période de rétention d'un tablespace UNDO

Le *Flashback Query* permet d'interroger les données dans l'état où elles étaient plusieurs heures auparavant. Pour cela, on indique au tablespace UNDO une durée de rétention des informations. Il est ainsi possible d'interroger les données de la base telles qu'elles étaient 12 heures auparavant. L'utilisation de cette possibilité sur une aussi longue période nécessite de disposer d'espace disque dans le tablespace UNDO car tout l'historique de modification des données est conservé. Les performances de la base sont peu affectées par cette possibilité.

Le package PL/SQL DBMS_FLASHBACK permet d'effectuer des interrogations dans le passé. La durée de rétention est précisée par le paramètre d'initialisation UNDO _RETENTION. Sa valeur par défaut est de 900 secondes mais elle peut être modifiée dynamiquement ou dans le fichier d'initialisation.

Ces données conservées nécessitant de l'espace disque, l'utilitaire Undo Tablespace Advisor d'Oracle Enterprise Manager permet d'en déterminer la taille nécessaire en se basant sur l'historique d'utilisation fourni dans les vues V\$UNDOSTAT.

Actions possibles sur un tablespace UNDO

Comme pour les autres tablespaces, vous pouvez ajouter des fichiers à un tablespace UNDO existant.

```
alter tablespace undotbs
  add datafile 'E:\oracle\oradata\TEST\undotbs03.dbf' SIZE 50M
  autoextend on
  next 10M
  maxsize 100M;
```

Il est possible de changer le tablespace UNDO d'une base. Pour cela :

1. Créez un nouveau tablespace UNDO ;
2. Arrêtez la base et modifiez les paramètres d'initialisation pour choisir le nouveau tablespace UNDO ;
3. Démarrez la base ;
4. Supprimez l'ancien tablespace par la commande DROP UNDO TABLESPACE.

La gestion des rollback segments

La suite de ce paragraphe décrit la gestion des rollback segments. C'est l'ancienne méthode proposée par Oracle pour gérer les segments en attente de validation ou d'annulation. Leur théorie a été abordée au chapitre 11, *Transactions et accès concurrents*.

Oracle recommande fortement d'utiliser la nouvelle méthode de gestion des segments d'annulation dans un tablespace de type UNDO. Ce point est détaillé précédemment.

Création d'un rollback segment

L'ordre SQL CREATE ROLLBACK SEGMENT assure leur création.

```
create public rollback segment rb1
  storage(initial 200K
           next 200K
           optimal 400K)
  tablespace rbs ;
```

Dans cet exemple, la clause STORAGE contient un nouveau paramètre : OPTIMAL. Ce paramètre est exclusivement réservé aux rollback segments. Il détermine la taille optimale que doit posséder le rollback segment. Si un utilisateur lance une transaction lourde, le rollback segment de celle-ci occupera la place nécessaire pour contenir toutes les anciennes données. Cette place peut être importante ; le mécanisme des extensions sera alors mis en œuvre. Dès que la transaction est terminée (COMMIT ou ROLLBACK), la place occupée par les différents segments devient inutile. Le paramètre OPTIMAL précise la taille à laquelle doit revenir le rollback segment.

Dès qu'il est créé, le rollback segment est OFFLINE, c'est-à-dire qu'aucune transaction ne peut l'utiliser. L'ordre SQL suivant le positionne ONLINE.

```
alter rollback segment RB1 online ;
```

Visualiser les rollback segments

L'ordre SQL suivant permet de visualiser la liste des rollback segments ainsi que leur statut.

```
tttitle center "Liste des rollback segments" skip 2
col nomsegment for a10
col iniext for 999999999
col nextex for 999999999
col pctin for 9999
col status for a8
col tablespace_name for a14 heading "Tablespace"

select segment_name "NomSegment", owner, tablespace_name,
       initial_extent "IniExt",
       next_extent "NextEx",pct_increase "PctIn",status
from sys.dba_rollback_segs
/
```

Liste des rollback segments

NomSegment	OWNER	Tablespace	IniExt	NextEx	PctIn	STATUS
SYSTEM	SYS	SYSTEM	51200	51200	0	ONLINE
RB_TEMP	SYS	SYSTEM	102400	102400	0	OFFLINE
RB1	PUBLIC	RBS	102400	256000	0	ONLINE
RB2	PUBLIC	RBS	102400	256000	0	ONLINE
RB3	PUBLIC	RBS	102400	256000	0	ONLINE

Dans cet exemple, les rollback segments SYSTEM et RBS_TEMP sont placés dans le tablespace SYSTEM. Seul le rollback segment SYSTEM est actif. Il est indispensable au fonctionnement de la base.

Suppression d'un rollback segment

Pour pouvoir supprimer un rollback segment, il ne doit pas être utilisé par une transaction. La méthode la plus simple est de le mettre offline puis de le supprimer :

```
alter rollback segment RB1 offline ;  
  
drop rollback segment RB1 ;
```

Nombre de rollback segments à créer

Chaque transaction utilise un rollback segment, mais celui-ci peut l'être simultanément par plusieurs transactions. Pour disposer d'un nombre de rollback segments cohérent par rapport au nombre d'utilisateurs connectés à une base, Oracle conseille la valeur *nombre d'utilisateurs connecté/3*. Cette valeur s'avère très satisfaisante dans la pratique.

Gestion des fichiers redo-log

Les fichiers redo-log assurent deux objectifs : la performance et la sécurité. La performance, car ils permettent à la base de données de traiter plus rapidement un gros volume d'écritures disque. La sécurité, car en cas d'arrêt brutal de votre machine, comme une coupure de courant, les fichiers redo-log permettent de valider les transactions valides et d'annuler celles qui étaient en cours. Ils peuvent aussi être archivés sur le serveur hébergeant la base et sur un serveur distant, grâce au mode ARCHIVELOG. Leur principe a été abordé au chapitre 9, *Les fichiers d'une base Oracle 10g*.

Avant toute action, effectuez une sauvegarde de la base de données et entraînez-vous sur une base test, sans utilisateurs actifs.

Création d'un fichier redo-log

Les fichiers redo-log initiaux sont créés en même temps que la base de données.

```
CREATE DATABASE TEST  
MAXINSTANCES 1  
MAXLOGHISTORY 1  
MAXLOGFILES 5  
MAXLOGMEMBERS 5  
MAXDATAFILES 100  
DATAFILE 'c:\oracle\oradata\TEST\system01.dbf' SIZE 100M REUSE  
        AUTOEXTEND ON  
        NEXT 10240K  
        MAXSIZE 1000M
```

```

UNDO TABLESPACE "UNDOTBS"
DATAFILE 'c:\oracle\oradata\TEST\undotbs01.dbf' SIZE 50M,
        'D:\oracle\oradata\TEST\undotbs02.dbf' SIZE 50M REUSE
        AUTOEXTEND ON
        NEXT 5120K
        MAXSIZE 1000M
CHARACTER SET WE8ISO8859P15
NATIONAL CHARACTER SET AL16UTF16
LOGFILE GROUP 1 ('c:\oracle\oradata\TEST\redo01.log') SIZE 100M,
        GROUP 2 ('c:\oracle\oradata\TEST\redo02.log') SIZE 100M,
        GROUP 3 ('c:\oracle\oradata\TEST\redo03.log') SIZE 100M
;

```

Caractéristiques des fichiers redo-log

Les vues V\$LOGFILE et V\$LOG détaillent les caractéristiques des fichiers redo-log. La vue V\$LOGFILE fournit la liste des fichiers redo-log de la base.

```

select * from v$logfile ;

```

Group#	STATUS	MEMBER
1		c:\oracle\oradata\TEST\redo01.log
2		c:\oracle\oradata\TEST\redo02.log
3		c:\oracle\oradata\TEST\redo03.log

La vue V\$LOG fournit le détail de ces fichiers et précise le fichier redo-log actif, c'est-à-dire celui dans lequel le processus Log Writer (DBWR) écrit.

```

select group#, sequence#, bytes, archived, status
from v$log ;

```

GROUP#	SEQUENCE#	BYTES	ARC	STATUS
1	269	1048576	NO	INACTIVE
2	270	1048576	NO	INACTIVE
3	271	1048576	NO	CURRENT

Actuellement, le groupe numéro 3 (c'est-à-dire le fichier c:\oracle\oradata\TEST\redo03.log) est actif. Aucun fichier n'a été archivé.

Basculer la base en mode ARCHIVELOG

Nous avons abordé le principe et les contraintes du mode ARCHIVELOG au chapitre 9, *Les fichiers d'une base Oracle 10g*.

Pour mettre la base en mode ARCHIVELOG :

1. Éditez le fichier d'initialisation *initSID.ora* pour y préciser les paramètres suivants :

```

...
log_archive_start = true

```

```
log_archive_dest = 'C:\oracle\admin\TEST\arch'
log_archive_format=ARC%S_%T.arc
...
```

Le paramètre LOG_ARCHIVE_DEST peut pointer vers un système de fichiers local ou vers un répertoire d'une machine distante. Sa syntaxe sera alors :

```
log_archive_dest='LOCATION=\repertoire\du\serveur\local'
log_archive_dest='SERVICE=label_du_service_a_utiliser'
```

2. Arrêtez la base. Le processus Log Writer (LGWR) sera lancé aux prochains démarrages.
3. Démarrez la base en mode exclusive avec SQL*Plus et positionnez-la en mode archivelog :

```
sqlplus /nolog
connect sys/change_on_install as SYSDBA
startup mount exclusive;

alter database archivelog ;

shutdown immediate ;
startup ;
```

4. Vérifiez :

```
select name, log_mode from v$database ;
```

NAME	LOG_MODE
TEST	ARCHIVELOG

5. Identifiez le fichier redo-log actif :

```
select group#, sequence#, bytes, archived, status
from v$log ;
```

GROUP#	SEQUENCE#	BYTES	ARC	STATUS
1	269	1048576	NO	INACTIVE
2	270	1048576	NO	INACTIVE
3	271	1048576	NO	CURRENT

6. Forcez la création d'un fichier log archivé :

```
alter system switch logfile;

-- vérification
select group#, sequence#, bytes, archived, status
from v$log ;
```

GROUP#	SEQUENCE#	BYTES	ARC	STATUS
1	269	1048576	NO	CURRENT
2	270	1048576	NO	INACTIVE
3	271	1048576	NO	INACTIVE

7. Vérifiez que le fichier d'archive a été créé au moment du « switch logfile » en regardant, sous Windows, le répertoire pointé par le paramètre LOG_ARCHIVE_DEST.

```
C:\> cd c:\oracle\admin\TEST\arch
C:\> dir A*
C:\> ARC001_272.arc
```

8. Fermez votre base et effectuez une sauvegarde complète. Elle servira de base sur laquelle tous les fichiers redo-log archivés seront appliqués en cas de restauration. N'oubliez pas de surveiller le remplissage de votre système de fichier dans lequel vous placez les fichiers redo-log archivés.

Basculer la base en mode NOARCHIVELOG

La démarche est similaire, il suffit de modifier la commande précédente.

```
alter database noarchivelog ;
```

Il est conseillé de ne pas mettre la base en mode ARCHIVELOG lorsque vous effectuez des opérations de maintenance comme des Export/Import ou des programmes batch lourds, car dans ce cas le volume des fichiers d'archives créés est directement lié au volume des données manipulées.

Si vous passez du mode ARCHIVELOG au mode NOARCHIVELOG, soyez toujours certain du mode dans lequel vous êtes. Rien n'est pire que de penser avoir un système de haute disponibilité opérationnel alors qu'il n'est pas actif...

Sauvegarde base ouverte, utilisateurs actifs

Cette option est très intéressante en cas de haute disponibilité. La base de données est alors nécessairement en mode ARCHIVELOG.

La sauvegarde base ouverte est traitée au chapitre 26, *La sauvegarde d'une base Oracle 10g*.

Informations sur le mode ARCHIVELOG

Les vues du dictionnaire de données qui contiennent des informations sur le mode ARCHIVELOG sont :

- V\$DATABASE : identifie si la base de données est en mode ARCHIVELOG ou NOARCHIVELOG ;
- V\$ARCHIVED_LOG : affiche des informations issues du fichier de contrôle concernant les fichiers redo-log archivés ;
- V\$ARCHIVE_DEST : décrit la destination des fichiers archivés ;
- V\$LOG : décrit les fichiers redo-log et indique celui qui est actif ;

- V\$LOG_HISTORY : contient l'historique de tous les passages d'un fichier redo-log à un autre.

La vue V\$LOG_HISTORY, couplée à l'étude du fichier d'alerte de votre base, permet de vous assurer que les permutations de fichiers redo-log ne sont pas trop fréquentes. Dans le cas contraire, il faut augmenter la taille de ces fichiers. C'est une opération de tuning qui donne souvent d'excellents résultats.

Ajouter un fichier redo-log

Comme pour la création des fichiers redo-log initiaux, cette opération s'effectue au niveau de la base de données.

```
alter database add logfile ('c:\oracle\oradata\TEST\redo04.log' ,
                           'c:\oracle\oradata\TEST\redo05.log' ,
                           'c:\oracle\oradata\TEST\redo06.log' )
size 30M ;
```

Cette opération ajoute trois fichiers redo-log, de 30 Mo chacun, à la base de données.

Supprimer un fichier redo-log

Le plus simple est de supprimer un fichier redo-log inactif. Identifiez d'abord le fichier redo-log actif.

```
select group#, sequence#, bytes, archived, status
from v$log ;
```

GROUP#	SEQUENCE#	BYTES	ARC	STATUS
1	269	1048576	NO	CURRENT
2	270	1048576	NO	INACTIVE
3	271	1048576	NO	INACTIVE
4	271	3145728	NO	INACTIVE
5	271	3145728	NO	INACTIVE
6	271	3145728	NO	INACTIVE

Placez le fichier redo-log actif sur un fichier que vous ne voulez pas supprimer.

```
alter system switch logfile ;
```

Supprimez les fichiers redo-log inactifs.

```
alter database drop logfile
member ('c:\oracle\oradata\TEST\redo01.log',
        'c:\oracle\oradata\TEST\redo02.log',
        'c:\oracle\oradata\TEST\redo03.log' );
```

Vérification :

```
select group#, sequence#, bytes, archived, status
from v$log ;
```

GROUP#	SEQUENCE#	BYTES	ARC	STATUS
4	271	3145728	NO	ACTIVE
5	271	3145728	NO	INACTIVE
6	271	3145728	NO	INACTIVE

N'oubliez pas que le fichier « disparaît » du dictionnaire Oracle mais qu'il existe encore sur le système de fichiers Windows. Il faut supprimer les anciens fichiers manuellement.

Déplacer un fichier redo-log

La procédure à suivre est identique à la création/suppression des fichiers redo-log.

Performances et redo-log

L'impact des fichiers redo-log sur les performances est importante. S'ils sont sous-dimensionnés, ils vont se remplir rapidement puis basculer de l'un vers l'autre, ce qui entraîne une attente le temps que toutes les écritures présentes dans le premier fichier soient écrites dans les fichiers de la base.

Ce goulot d'étranglement étant fréquemment rencontré et facile à éviter, Oracle impose maintenant des fichiers redo-log d'une taille minimale de 4 Mo.

Pour tout savoir de l'historique des mouvements entre fichiers redo-log sur une période d'activité, consultez la vue V\$LOG_HISTORY.

Oracle conseille une durée minimale de 20 min pour remplir un fichier redo-log. Cela évite des bascules de fichier à fichier trop fréquentes.

Cas d'usage de Log Miner

Oracle propose un outil d'analyse du contenu des fichiers redo-log d'une base : le Log Miner. Cet utilitaire analyse de manière séquentielle l'ensemble des opérations réalisées sur une base. Vous pouvez ainsi répondre à des questions telles que :

- Comment visualiser sous forme d'ordres SQL le contenu des fichiers redo-log, c'est-à-dire des actions qui ont modifié les données de la base ?
- Quelles sont les modifications de structure qui ont été apportées à une base ?
- Quels sont les derniers ordres SQL enregistrés dans le fichier redo-log juste avant de rencontrer un problème disque ?
- Quelles sont les modifications des données d'une table ?
- Quelles sont les modifications apportées par un utilisateur, sur plusieurs tables ou sur une table particulière ?

On se rend compte que son usage appartient à deux grandes familles : aller le plus loin possible dans la récupération des actions effectuées sur une base suite à un problème technique d'une part et la surveillance et le contrôle d'accès aux données d'autre part.

L'avantage procuré par Log Miner est que rien ne doit être prévu a priori, tout est analysé a posteriori.

Principe d'utilisation de Log Miner

Pour transformer le contenu des fichiers redo-log en une suite d'ordres SQL compréhensibles, le Log Miner s'appuie sur le dictionnaire de données présent dans la base ou stocké sous forme d'un fichier plat ou de redo-log à l'extérieur de la base. Ce sont ensuite les procédures PL/SQL des packages DBMS_LOGMNR et DBMS_LOGMNR_D associées aux vues V\$LOGMNR_CONTENTS et V\$LOGxx du dictionnaire qui permettent l'analyse complète.

On peut ainsi analyser tous les ordres SQL effectués sur une base en ayant à sa disposition les fichiers redo-log à analyser, le dictionnaire de données dans un fichier ASCII et une base de données. Log Miner peut être utilisé sur une machine différente de celle sur laquelle fonctionne la base.

C'est un moyen de surveillance très puissant et très discret qui ne manquera pas d'être mis en œuvre dans des environnements où la traçabilité et l'accès à l'information sont sensibles. Bienvenue dans le monde de Big Brother !

Enfin, une interface graphique intégrée à Oracle Enterprise Manager permet d'accéder à quelques informations issues de Log Miner. Les cas de figures à étudier étant très variés, un outil graphique ne peut pas en couvrir tous les aspects.

Gestion des fichiers de contrôle

Ce paragraphe précise comment gérer les fichiers de contrôle. Ce sont les fichiers Oracle les plus simples et les plus rapides à administrer. En revanche, leur perte peut s'avérer désastreuse. Nous abordons des opérations de base comme ajouter, renommer, déplacer et supprimer des fichiers de contrôle.

Les fichiers de contrôle

Les fichiers de contrôle contiennent principalement :

- Le nom de la base ;
- Le nom et l'emplacement des fichiers liés à la base et aux redo-log ;
- La séquence identifiant le redo-log actif ;
- Des informations internes concernant l'état de la base, son statut lors du dernier arrêt, etc.

Ces informations sont essentielles pour démarrer et restaurer une base.

Dans le fichier d'initialisation d'une instance, qu'il soit sous la forme d'un fichier `initSID.ora` ou d'un fichier persistant `SPFILE`, le paramètre `CONTROL_FILES` donne le nom et l'emplacement de tous les fichiers de contrôle de la base de données. Durant le démarrage de l'instance Oracle, la lecture de ce paramètre fournit à la base la liste de tous les fichiers de contrôle. Seul le premier d'entre eux sera lu. En revanche, dès qu'un changement intervient, les mises à jour ont lieu simultanément dans tous les fichiers de contrôle.

Dans le fichier `initSID.ora`, ce paramètre ressemblera à la syntaxe suivante :

```
...
control_files = ('c:\oracle\oradata\TEST\control01.ctl',
                 'D:\oracle\oradata\TEST\control02.ctl')
...
```

Pour connaître dynamiquement cette valeur, vous pouvez interroger la base à partir de `SQL*Plus` :

```
show parameters control_files
```

Si vous utilisez la gestion automatique des fichiers par Oracle, cette valeur ne figurera pas dans le fichier d'initialisation (voir plus loin dans ce chapitre).

Visualiser les fichiers de contrôle existants

La vue `V$CONTROLFILE` permet de connaître la liste des fichiers de contrôle d'une instance en cours de fonctionnement.

```
select * from v$controlfile ;
```

STATUS	NAME
-----	-----
	c:\oracle\oradata\TEST\control01.ctl
	D:\oracle\oradata\TEST\control02.ctl

Dans notre cas, la base de données comporte deux fichiers de contrôle, situés sur deux disques différents. C'est le minimum de sécurité en cas de perte de disque.

Il faut toujours disposer de plusieurs fichiers de contrôle, leur perte pouvant engendrer de nombreuses difficultés. Oracle 10g permet de créer un fichier de contrôle même si tous les autres ont disparu. C'est une opération délicate qui nécessite de connaître parfaitement les caractéristiques de la base.

Ajouter un fichier de contrôle

La seule contrainte est de fermer la base de données. Vous devez ensuite effectuer les actions suivantes :

Arrêter, par un `SHUTDOWN NORMAL` ou `IMMEDIATE`, la base de données.

Sous Windows, copiez un fichier de contrôle existant sous un autre nom :

```
copy c:\oracle\oradata\TEST\control01.ctl E:\oracle\oradata\TEST\control03.ctl
```

Dans le fichier *initSID.ora*, indiquez ce nouveau fichier :

```
...
control_files = ('c:\oracle\oradata\TEST\control01.ct1',
                 'D:\oracle\oradata\TEST\control02.ct1',
                 'E:\oracle\oradata\TEST\control03.ct1' )
...
```

Démarrez l'instance et contrôlez le résultat :

```
select * from v$controlfile ;
```

STATUS	NAME
-----	-----
	c:\oracle\oradata\TEST\control01.ct1
	D:\oracle\oradata\TEST\control02.ct1
	E:\oracle\oradata\TEST\control03.ct1

Sauvegarder un fichier de contrôle base « en marche »

La solution précédente implique la fermeture de la base pour ajouter un fichier de contrôle. Heureusement, cela n'est pas nécessaire pour en sauvegarder un exemplaire base ouverte.

```
alter database backup controlfile to 'c:\temp\backup.ct1' ;
```

Tous les fichiers de contrôle étant identiques, il n'est pas nécessaire de préciser celui à sauvegarder.

Pour sauvegarder un fichier de contrôle base arrêtée, il suffit de le copier.

Déplacer et supprimer un fichier de contrôle

Pour supprimer un fichier de contrôle, il faut procéder de la même façon : le retirer du fichier *initSID.ora*, fermer puis redémarrer la base et supprimer physiquement le fichier sous Windows.

Déplacer un fichier de contrôle est une opération identique à sa création.

Que faire lorsque tous les fichiers de contrôle sont perdus ?

Jusqu'à la version Oracle8, la perte de tous les fichiers de contrôle entraînait celle de la base et il fallait repartir d'une sauvegarde. Maintenant, la commande `CREATE CONTROLFILE` permet de créer un fichier de contrôle mais cela nécessite de lui fournir l'ensemble des renseignements qu'il contient.

Cette opération est très périlleuse : si elle échoue, vous risquez d'obtenir un fichier de contrôle inopérant et les modifications apportées à la base la rendront inexploitable. Une sauvegarde sérieuse s'impose avant tout essai.

L'ensemble des informations nécessaires à un CREATE CONTROLFILE peut être généré automatiquement par l'ordre :

```
alter database backup controlfile to trace ;
```

Cela ne peut s'effectuer que si la base est démarrée. Le fichier trace généré contiendra alors l'ensemble des informations nécessaires à la création d'un nouveau fichier de contrôle.

Gestion automatique des fichiers par Oracle

La gestion des fichiers proposée par Oracle offre de multiples options pour en faciliter l'administration : accroissement automatique de leur taille, gestion interne, etc.

Oracle 10g va plus loin en vous proposant d'assurer à 100 % la gestion de l'ensemble des fichiers composant une base, y compris leur création, suppression, nom, etc. Les fichiers concernés par cette nouveauté sont :

- les fichiers liés aux tablespaces ;
- les fichiers redo-log ;
- les fichiers de contrôle.

Les autres fichiers de trace, audit, alerte, etc. ne sont pas concernés.

En utilisant cette possibilité, on peut arriver à l'ordre de création d'une base suivant :

```
CREATE DATABASE TEST ;
```

La syntaxe est alors très simple, surtout si on la compare à celle utilisée au chapitre 14, *La création d'une base Oracle 10g*. Mais attention, tous les fichiers qui composent la base existent toujours : leurs nom, dimensionnement, gestion, création, suppression, emplacement, etc., sont entièrement délégués à Oracle 10g.

Pourquoi utiliser la gestion automatique des fichiers ?

La gestion automatique des fichiers peut s'avérer très pratique pour ces deux cas :

- une base de test à construire rapidement ;
- pour une Standby Database.

L'intérêt pour une base de test est évident : cela permet de se focaliser sur les tests à effectuer en limitant au maximum la gestion de la base.

Une Standby Database est une base de secours, placée sur un serveur distinct de celui hébergeant la base de production. Régulièrement, les fichiers redo-log sont transmis et appliqués à la Standby Database. Elle est ainsi prête à remplacer la base de production si celle-ci s'avère défaillante.

Lorsqu'une opération impacte un fichier sur la base d'origine, le fichier redo-log répercute cette action à la Standby Database. Cette action nécessite une intervention humaine sauf si la base est en gestion automatique des fichiers. Cela limite fortement la surveillance à apporter à ce type de base.

Quels inconvénients présente la gestion automatique des fichiers ?

Pour l'administrateur Oracle, le plus grand inconvénient provient d'une perte de contrôle sur les fichiers et le paramétrage de la base. Voici quelques éléments de réflexion :

- L'espace disque peut-il être mis à disposition d'un dispositif automatique ?
- Êtes-vous certain de disposer de suffisamment de place disque pour permettre la création automatique de plusieurs fichiers de 100 Mo ?
- Acceptez-vous que la taille de certains fichiers puisse dépasser celle de vos médias de sauvegarde ?
- Pour créer les fichiers, Oracle utilise des options par défaut. Connaissez-vous ces options ? Vous conviennent-elles ?
- La mise en place de cette fonctionnalité vous apporte-t-elle des avantages ?

Vous l'avez compris, nous déconseillons cette option pour une base de production.

Pourtant, pour une société éditrice de logiciels, l'avantage est certain : la base s'administre toute seule et n'engendre aucun appel de la part de ses clients, même si tout cela peut conduire à un blocage grave. D'un autre côté, pour un client final ne disposant pas de compétences informatiques, il est difficile de rester bloqué pour le simple ajout d'un fichier à un tablespace.

Dernier point, il est possible de mélanger création manuelle et automatique de fichiers. La simple présence de deux paramètres dans le fichier d'initialisation de la base décide de l'utilisation ou non de la gestion automatique des fichiers.

Les paramètres d'initialisation

Les paramètres d'initialisation concernés sont :

- DB_CREATE_FILE_DEST ;
- DB_CREATE_ONLINE_LOG_DEST_n.

DB_CREATE_FILE_DEST précise le répertoire où Oracle 10g créera les fichiers de tablespaces temporaires ou de données.

DB_CREATE_ONLINE_LOG_DEST_n précise les répertoires où Oracle 10g créera les fichiers de contrôle et redo-log. Comme il est recommandé de multiplexer ces fichiers, il est possible d'indiquer plusieurs répertoires.

Par exemple, on peut avoir les valeurs suivantes dans le fichier d'initialisation :

```
db_create_file_dest='c:\oracle\oradata\AUTO'  
db_create_online_log_dest_1='c:\oracle\oradata\AUTO'  
db_create_online_log_dest_2='d:\oracle\oradata\AUTO'  
db_create_online_log_dest_3='e:\oracle\oradata\AUTO'
```

Attention, le simple fait de préciser l'une de ces valeurs dans le fichier d'initialisation implique le transfert de la gestion automatique des fichiers à Oracle !

Les options utilisées par défaut

La gestion automatique obéit néanmoins à des règles. Elles sont très nombreuses car les cas de figure ne manquent pas : création d'une base, passage d'une base existante en mode automatique, cas des fichiers de données, des fichiers temporaires, de contrôle, redo-log, etc.

Pour simplifier, le fichier d'un tablespace temporaire ou de données créé automatiquement aura les caractéristiques suivantes : taille de 100 Mo, fichier *auto-extensible* et taille maximale illimitée.

Pour un fichier redo-log, sa taille sera de 100 Mo. Aucune caractéristique particulière n'est précisée pour un fichier de contrôle car sa gestion est dans tous les cas assurée automatiquement par la base.

Toute la difficulté consiste à déléguer la gestion des fichiers à Oracle tout en connaissant les avantages, limites et risques qu'elle apporte.

Exemple d'une base créée avec ces options

Dans l'exemple suivant, la base AUTO sera créée le plus simplement possible.

1. Création des répertoires OFA destinés à recevoir les fichiers de données, de contrôle de redo-log ainsi que les fichiers d'administration de la base. L'utilisateur Windows doit posséder les droits de lecture/écriture sur ces répertoires :

```
# Répertoires pour les fichiers de la base  
c:\oracle\oradata\AUTO  
d:\oracle\oradata\AUTO  
e:\oracle\oradata\AUTO  
  
# Répertoires pour les fichiers d'administration  
c:\oracle\admin\AUTO\udump  
c:\oracle\admin\AUTO\cdump  
c:\oracle\admin\AUTO\bdump  
c:\oracle\admin\AUTO\exp  
c:\oracle\admin\AUTO\scripts
```

2. Création d'un fichier d'initialisation initAUTO.ora. Les paramètres db_create_file_dest et db_create_online_log_dest sont renseignés :

```
# Fichier initAUTO.ora

db_block_size=8192
db_cache_size=15428800
open_cursors=300

db_create_file_dest='c:\oracle\oradata\AUTO'
db_create_online_log_dest_1='c:\oracle\oradata\AUTO'
db_create_online_log_dest_2='d:\oracle\oradata\AUTO'
db_create_online_log_dest_3='e:\oracle\oradata\AUTO'

#####
# Diagnostics et statistiques
#####
background_dump_dest=c:\oracle\admin\AUTO\bdump
core_dump_dest=c:\oracle\admin\AUTO\cdump
user_dump_dest=c:\oracle\admin\AUTO\udump
timed_statistics=TRUE

#####
# Distribué, réplication et cliché
#####
db_domain=gilles.fr
compatible=10.0.0
db_name=AUTO
instance_name=AUTO

#####
# Fichier de journalisation et récupération
#####
fast_start_mttr_target=300

#####
# Pools
#####

large_pool_size=1048576
shared_pool_size=10428800

#####
# Resource Manager
#####
resource_manager_plan=SYSTEM_PLAN

#####
# Segments d'annulation (Undo et Rollback) gérés par le système
#####
undo_management=AUTO
```

```
undo_tablespace=UNDOTBS

#####
# Tri, jointures par hachage, index bitmap
#####
sort_area_size=524288
```

3. Création de la base de données. L'ordre CREATE DATABASE a été simplifié au maximum.

```
connect SYS/change_on_install as SYSDBA

startup nomount pfile="c:\oracle\ora10\database\initAUTO.ora"

CREATE DATABASE AUTO;
```

4. Les fichiers créés automatiquement par Oracle sont :

```
Fichiers des tablespaces
-----
c:\oracle\oradata\AUTO\ora_system_xqc29gob.dbf  SYSTEM    127  AVAILABLE
c:\oracle\oradata\AUTO\ora_sys_undo_xqc2bpnc.dbf SYS_UNDOTS  129  AVAILABLE

Fichiers redo-log
-----
c:\oracle\oradata\AUTO\ora_1_xqc28jss.log
d:\oracle\oradata\AUTO\ora_1_xqc28r1p.log
e:\oracle\oradata\AUTO\ora_1_xqc28mqp.log
c:\oracle\oradata\AUTO\ora_2_xqc28zo2.log
d:\oracle\oradata\AUTO\ora_2_xqc2968d.log
e:\oracle\oradata\AUTO\ora_1_xqc28erd.log

Fichiers de controle
-----
c:\oracle\oradata\AUTO\ora_xqc28hym.ctl
d:\oracle\oradata\AUTO\ora_xqc28jcy.ctl
e:\oracle\oradata\AUTO\ora_xqc28jms.ctl
```

La base créée est minimale. Elle contient le tablespace SYS_UNDOTS qui est de type UNDO, mais pas de tablespace temporaire de type TEMPORARY. De plus, aucun tablespace supplémentaire destiné à héberger les schémas des applications n'existe et aucun catalogue n'est créé. Vous devez lancer les catalogues :

```
connect SYS/change_on_install as SYSDBA
@c:\oracle\ora92\rdbms\admin\catalog.sql;
@c:\oracle\ora92\rdbms\admin\catexp7.sql;
@c:\oracle\ora92\rdbms\admin\catblock.sql;
@c:\oracle\ora92\rdbms\admin\catproc.sql;
@c:\oracle\ora92\rdbms\admin\catoc tk.sql;
@c:\oracle\ora92\rdbms\admin\catobtk.sql;
@c:\oracle\ora92\rdbms\admin\caths.sql;
@c:\oracle\ora92\rdbms\admin\owminst.plb;
connect SYSTEM/manager
@c:\oracle\ora92\sqlplus\admin\pupbld.sql;
```

```
connect SYSTEM/manager
@c:\oracle\ora92\sqlplus\admin\help\hlpbld.sql helpus.sql;
exit;

# Lancez maintenant la création des tablespaces applicatifs
CREATE TABLESPACE TBS_1 ;
# les fichiers seront créés par Oracle
```

En fournissant plus de renseignements lors du CREATE DATABASE, il est possible de créer une base minimale plus aboutie. Si vous fournissez trop de paramètres lors de la création de la base, revenez à la méthode utilisant l'assistant DBCA décrite au chapitre 14, *Création d'une base Oracle 10g*.

Conclusion sur la gestion automatique des fichiers

La gestion automatique des fichiers est une avancée proposée par Oracle. Si les avantages sont certains à court terme, des inconvénients majeurs peuvent se présenter à moyen ou long terme.

Oracle possède déjà de nombreuses options facilitant la gestion de l'espace disque : fichiers de type *auto-extensible*, taille maximale d'un fichier, tablespaces TEMPORARY, UNDO, etc. Ces paramètres permettent déjà de limiter très fortement les interventions d'un administrateur Oracle et nous vous conseillons de les utiliser.

Toute la difficulté de cette gestion automatique consiste à l'utiliser intelligemment en connaissant les avantages, limites et risques qu'elle apporte.

Gestion du fichier d'initialisation

Historiquement, toutes les versions d'Oracle ont utilisé un simple fichier texte pour y conserver les paramètres d'initialisation. Avec Oracle 10g, ces valeurs peuvent maintenant être gérées automatiquement dans un fichier binaire persistant.

Comme de plus en plus de paramètres d'initialisation sont modifiables dynamiquement, cela permet de conserver les valeurs modifiées et de les réutiliser au prochain démarrage de la base. Comparé à la gestion manuelle, cela évite de devoir modifier le fichier d'initialisation pour prendre en compte des modifications effectuées base lancée par la commande ALTER SYSTEM.

Le fichier persistant ou SPFILE (*System Parameter File*) est créé par la commande CREATE SPFILE.

Démarrage d'une base

Même si un fichier d'initialisation persistant existe, vous pouvez démarrer une base en indiquant un fichier texte d'initialisation.

```
sqlplus /nolog
connect system/manager as sysdba
startup pfile=c:\oracle\oradata\TEST\scripts\init.ora
```

Avec un fichier persistant, vous n'avez aucun fichier à indiquer.

```
sqlplus /nolog
connect system/manager as sysdba
startup
```

L'ordre CREATE DATABASE fait toujours référence à un fichier d'initialisation texte. Ensuite, le fichier SPFILE est créé.

Création d'un fichier persistant

Le fichier d'initialisation persistant est créé par la commande CREATE SPFILE d'après un fichier texte d'initialisation. Il s'applique à une base précise et se place sur le serveur en ORACLE_HOME\database. Cela permet à Oracle de le situer, surtout si la commande est effectuée à distance.

```
# l'utilisateur doit être connecté sous SYSDBA
set ORACLE_SID=TEST
sqlplus /nolog
connect SYS/change_on_install as SYSDBA

create spfile='c:\oracle\ora92\database\spfileTEST.ora'
FROM pfile='c:\oracle\admin\TEST\pfile\init.ora';
```

Le nom donné au fichier SPFILE est important, il doit respecter la syntaxe *spfileORACLE_SID.ora*, par exemple *spfileTEST.ora* dans l'exemple précédent.

Pour identifier l'emplacement d'un fichier SPFILE existant, utilisez la commande :

```
sqlplus /nolog
connect system/manager

show parameters spfile
```

Créer un fichier d'initialisation

L'opération normale consiste à créer un fichier SPFILE d'après un fichier texte, mais l'inverse est possible par la commande CREATE PFILE.

```
sqlplus /nolog
connect system/manager as SYSDBA

create pfile='c:\oracle\admin\TEST\pfile\init.ora'
FROM spfile='c:\oracle\ora92\database\spfileTEST.ora';
```

Modifier des valeurs par l'ordre ALTER SYSTEM

Pour que la modification d'un paramètre d'initialisation soit prise en compte, il fallait arrêter la base puis la redémarrer. Cela forçait Oracle à prendre en compte la nouvelle valeur. C'était une contrainte majeure pour les bases nécessitant une haute disponibilité.

Sont alors apparus les paramètres dynamiques, qui peuvent être modifiés par l'ordre ALTER SYSTEM sans qu'il soit nécessaire d'arrêter puis relancer la base. De nombreux paramètres sont devenus dynamiques avec Oracle 10g.

La commande ALTER SYSTEM permet de modifier les paramètres en mémoire, dans le fichier SPFILE ou simultanément en mémoire et dans le SPFILE.

La modification uniquement en mémoire (MEMORY) est utile si vous voulez tester un paramétrage sans le rendre permanent dans le SPFILE ou si vous n'utilisez pas de fichier SPFILE.

La modification uniquement dans le fichier persistant (SPFILE) est utile pour les commandes qui ne sont pas modifiables dynamiquement mais qui seront prises en compte au prochain démarrage. Enfin, les deux actions peuvent être simultanées (BOTH).

La syntaxe est la suivante :

```
# le paramètre SCOPE
# peut prendre les valeurs MEMORY, SPFILE, BOTH

ALTER SYSTEM SET nom_du_paramètre = valeur
                COMMENT='commentaire libre'
                SCOPE=MEMORY;
```

Par exemple, pour modifier le paramètre DB_CACHE_SIZE :

```
ALTER SYSTEM SET db_cache_size=15M
                COMMENT='Passage de 20M à 15M pour diminuer le swap'
                SCOPE=BOTH;
```

Pour forcer une valeur à reprendre sa valeur par défaut, affectez-lui une valeur nulle :

```
ALTER SYSTEM SET nom_du_paramètre='' ;
```

Visualiser les valeurs d'initialisation

Les ordres SQL suivants permettent de visualiser les caractéristiques des paramètres d'initialisation :

```
sqlplus /nolog
connect system/manager

# Liste des paramètres
show parameters

# vue accédant l'ensemble des paramètres
select * from v$parameter ;

# vues complémentaires
select * from v$parameter2 ;
select * from v$spparameter ;
```

Outils pour modifier les paramètres d'initialisation

Pour modifier l'ensemble des paramètres d'initialisations, les deux outils possibles sont SQL*Plus pour passer les commandes en mode ligne et Oracle Enterprise Manager qui habille graphiquement ces commandes.

Résumé

Ce chapitre a abordé la façon dont Oracle 10g gère l'espace disque alloué au cours des opérations de création, d'insertion et de modification opérées sur les données. La gestion des différents fichiers qui composent une base Oracle 10g a été abordée. Partant d'éléments théoriques comme le tablespace ou la gestion de l'espace au sein de la base de données, de nombreux exemples mettent en pratique ces notions.

Une nouvelle fois, n'oubliez pas de vous entraîner sur une base de test avec Oracle Enterprise Manager avant d'effectuer ces opérations sur une base de production.

Stratégie de sécurité sous Windows

Dans ce chapitre :

- les points sensibles de sécurité ;
- les privilèges SYSOPER et SYSDBA ;
- démarrage/arrêt d'une base Oracle 10g sans mot de passe ;
- gestion et test de la sécurité d'accès à vos bases Oracle 10g.

D'une manière générale, la gestion de la sécurité est le parent pauvre des administrateurs de système. Dès lors qu'un serveur héberge les données de l'entreprise, ce n'est plus une machine de test, sa sécurité doit être renforcée et faire l'objet d'une attention pointilleuse. Les sauvegardes qui font l'objet du chapitre 26, *La sauvegarde d'une base Oracle 10g* sont, elles aussi, indispensables.

Ce chapitre aborde le thème de la sécurité, les différentes techniques pour démarrer/arrêter une base Oracle 10g et s'y connecter avec ou sans mot de passe. Ces techniques sont intéressantes, car elles permettent de réaliser des scripts de commande, sans qu'apparaisse un mot de passe : la sécurité du système n'est pas compromise. Il est ainsi possible de changer tous les mots de passe sans risque de rendre des scripts inopérants.

Nous traitons ici de la sécurité d'accès à Oracle 10g sous Windows, pas de la sécurité de Windows elle-même. Tout comme les sauvegardes du système, ce point important dépasse le cadre de cet ouvrage.

Sécurité : les points sensibles

La solidité d'une chaîne vaut celle de son maillon le plus faible. Il en est de même pour tous les systèmes informatiques. La sûreté d'accès à Oracle 10g ne doit pas compromettre celle de Windows, et inversement ! C'est pourquoi la politique d'accès à tout système Windows et les droits octroyés doivent être particulièrement surveillés. Nous n'allons pas décrire la mise en place d'un serveur Windows sécurisé, mais les points importants qui concernent Oracle 10g.

Lorsque plusieurs utilisateurs ont accès au même serveur Windows, c'est à l'administrateur de définir leurs droits sur les répertoires utilisés par Oracle 10g. Omettre de gérer ces accès expose le système aux risques suivants :

- autoriser, sans contrôle, l'accès physique aux fichiers d'Oracle 10g et à ceux des bases de données ;
- accès libre aux comptes utilisateur des bases Oracle 10g.

Nous vous proposons dans ce chapitre plusieurs méthodes pour contrer ces deux risques majeurs. Étudions au préalable les conditions qui mettent en danger la sécurité d'un système.

Le danger vient de l'intérieur

Toutes les enquêtes concernant la sécurité des systèmes informatiques arrivent aux mêmes conclusions : les plus grandes menaces se situent au sein même de votre entreprise. Qu'elles soient accidentelles ou volontaires, les attaques trouvent leur origine à plus de 90 % dans l'entourage immédiat. Toutes les raisons, même les plus extravagantes, sont valables : jalousie, volonté de nuire, maladresse, envie de montrer sa supériorité technique, d'accéder à des données confidentielles, etc. Soyez particulièrement vigilant !

Passons en revue les éléments qui fragilisent le plus un système informatique.

Des droits trop étendus

L'accès à des fichiers en lecture, modification ou suppression, qu'il soit accidentel ou volontaire, ne peut s'effectuer que si l'utilisateur en possède les droits d'accès. C'est pourquoi l'installation d'Oracle 10g s'effectue sous le compte Windows Administrateur. Il est toutefois possible de créer un compte spécialisé pour installer Oracle et d'autres pour effectuer les opérations d'administration courantes. Encore faut-il que votre organisation s'adapte à l'utilisation de ces droits ! C'est pourquoi, sous Windows, nous considérons que l'administrateur du système est aussi celui d'Oracle. Ces points sont détaillés au chapitre 13, *L'installation d'Oracle 10g sous Windows* et 14, *Création d'une base Oracle 10g*.

Dans ce dernier chapitre, nous précisons les droits d'accès qu'Oracle octroie aux fichiers et répertoires qu'il crée.

Divulgaration des mots de passe

La méthode la plus simple pour accéder à un système, c'est d'en connaître le mot de passe. C'est une évidence souvent négligée. On rencontre même de très nombreux comptes Windows sans aucun mot de passe ! Assurez-vous que tous vos accès sont protégés.

Les mots de passe doivent être significatifs. Évitez d'utiliser des couples utilisateur/mot de passe comme *administrateur/admin* ou *oracle/oracle*... Dans ce cas, la protection est inexistante.

Oracle a renforcé la sécurité des mots de passe. Depuis Oracle 10g, pendant la création de la base, vous avez l'obligation de changer les principaux mots de passe et vous ne pouvez pas utiliser les anciens mots de passe par défaut. Les couples SYS/change_on_install (explicite !) et SYSTEM/manager sont maintenant interdits. Tous les comptes Oracle dont les mots de passe n'ont pas été changés sont désactivés. Il convient de les réactiver à partir d'Enterprise Manager ou par un ordre SQL. De bons points pour la sécurité !

Une fois les accès protégés, communiquez les mots de passe uniquement à des tiers de confiance et efforcez-vous de les changer régulièrement. Évitez de les inscrire sur des Post-it collés sur l'écran.

Une enveloppe de sécurité scellée contenant les mots de passe des systèmes sera maintenue à jour et accessible aux personnes accréditées. L'accès à cette enveloppe devra être lui aussi sécurisé. Cette technique simple permet d'éviter d'être dérangé durant ses congés !

Dès qu'un mot de passe a été utilisé ou est connu d'une personne extérieure, changez-le immédiatement. Pour les utilisateurs occasionnels, créez des accès temporaires que vous fermerez après leur départ. Oracle et Windows le permettent : vous n'avez plus aucune excuse...

Les raisons empêchant de changer de mot de passe

Changer de mot de passe ne présente que des intérêts en termes de sécurité. Quelles raisons techniques pourraient constituer un obstacle à leur modification ?

La raison la plus classique est le mot de passe figurant en clair dans des scripts de commandes. Outre le fait que cela constitue une très importante faille de sécurité, il est difficile de repérer de façon exhaustive les lieux où il est mentionné. Face à cet inconvénient, les utilisateurs sont enclins à ne plus le changer.

Dans ce chapitre, nous exposons plusieurs méthodes pour accéder à Oracle 10g à partir de scripts de commandes sans exposer de mot de passe.

Les objectifs

Les objectifs que nous nous fixons dans ce chapitre doivent respecter les règles suivantes :

- seul l'utilisateur qui a installé les logiciels Oracle et créé les bases de données Oracle 10g doit pouvoir accéder aux fichiers Oracle ;

- aucun mot de passe ne doit figurer dans les scripts d'administration. Ne pas respecter cette consigne entraîne deux risques majeurs : les mots de passe sont susceptibles d'être divulgués et leur présence dans des scripts interdit leur changement ;
- les scripts d'administration doivent être protégés. S'ils ne le sont pas, ils peuvent être modifiés. Assurez-vous que tous les scripts utilisés pour administrer la base sont la propriété exclusive de l'administrateur Oracle ou de *Administrateur* ;
- lorsque vous exécutez des programmes à partir d'un ordonnanceur ou d'autres commandes dans les scripts, indiquez toujours le chemin d'accès complet à vos programmes. Se fier aux variables d'environnement peut être source d'erreurs. Mais cela permet surtout à des commandes pirate du même nom que la vôtre d'être exécutées, si elles sont placées dans un chemin d'accès plus proche de la racine ;
- ces mêmes recommandations sont valables pour toutes les stations d'administration connectées en client-serveur sur vos bases. N'autorisez aucun accès (ni en lecture, ni en écriture) aux comptes autres que celui de l'administrateur Oracle ;
- permettre à d'autres personnes que les administrateurs Oracle 10g d'effectuer des tâches d'administration sans qu'elles disposent pour autant d'un accès complet à la base ;
- autoriser certains utilisateurs, déjà connectés à Windows, à se connecter à la base sans nécessairement taper de mot de passe.

La sécurité d'accès physique

L'accès physique aux programmes et exécutables d'Oracle 10g ainsi qu'aux fichiers qui composent votre base doit être protégé. Sans détailler inutilement tous les risques, nous en exposons ici les principaux.

L'accès à votre machine

L'accès au serveur est-il sécurisé ? Nous savons tous que le vol de matériel est une menace. Le serveur Windows peut-il être volé ? Ne souriez pas, ce cas s'est déjà produit...

De même, il doit être impossible de « cambrioler » une base de données. Assurez-vous que l'accès aux fichiers de données est verrouillé et qu'il en est de même de leurs sauvegardes. À quoi bon assurer une confidentialité maximale, si la cartouche de sauvegarde est posée négligemment à côté du serveur. Il est bien plus simple de voler une sauvegarde que de tenter une intrusion dans le serveur ! Si les fichiers qui composent une base nécessitent des compétences techniques et un environnement de travail identique à l'environnement d'origine pour réutiliser la base, les exportations peuvent être importées dans n'importe quelle base Oracle, soyez donc vigilant.

L'accès réseau doit être l'objet de toute votre attention. Assurez-vous que tous les accès au serveur soient sécurisés.

Les comptes Windows

Sous Windows, deux types de comptes utilisateur nous intéressent : *Administrateur* et les utilisateurs de vos bases qui possèdent un compte Windows.

Le compte *Administrateur* doit faire l'objet de toutes les attentions d'un administrateur Windows. Confiez-vous votre numéro de carte bancaire à tout le monde ? Il doit en être de même pour l'accès *Administrateur*. Sur le système, le compte *Administrateur* est responsable de la sécurité d'accès aux fichiers, principalement aux fichiers système. On rencontre trop de sites où, pour plus de simplicité, tous les droits (lecture, écriture, exécution) sont attribués à tous les fichiers. Ne transformez pas votre système Windows en passoire ! De même, changez très régulièrement le mot de passe *Administrateur* qui doit impérativement figurer dans l'enveloppe de secours.

Le compte Windows *Administrateur* sert aussi à installer les logiciels Oracle 10g et accéder physiquement aux fichiers qui composent les bases. Il est possible de créer un compte spécial à cet effet.

Souvent, les utilisateurs Windows doivent s'identifier auprès de Windows avant de se connecter à Oracle. Pourquoi alors saisir une nouvelle identification pour accéder à Oracle ? Ne peut-on pas synchroniser l'identification Windows et celle d'Oracle ? Oracle permet de récupérer l'authentification Windows pour autoriser l'accès à une base. L'utilitaire *Oracle Administrative Assistant for Windows* vous permet d'effectuer simplement cette configuration. Nous ne décrivons pas dans le détail cet assistant, car nous préférons vous présenter les « fondamentaux » Oracle, l'assistant n'étant que l'automatisation de certaines de ces actions.

Les utilisateurs Oracle

Sauf cas particulier évoqué dans ce chapitre, vous devez toujours saisir un nom d'utilisateur et un mot de passe pour accéder aux bases Oracle 10g. La politique de définition et de changement de mot de passe doit être respectée pour les comptes définis dans les bases Oracle 10g.

Deux comptes doivent être particulièrement surveillés, car ils sont créés avec un mot de passe défini lors de la création de la base de données : le compte SYS, propriétaire du dictionnaire et le compte SYSTEM, premier administrateur de la base. Comme nous l'avons évoqué, depuis Oracle 10g, les mots de passe doivent obligatoirement être changés et les anciens mots de passe par défaut sont interdits.

Pour les versions antérieures à Oracle 10g, il faut obligatoirement changer les mots de passe de SYS et SYSTEM. Plus de 60 % des bases Oracle conservent les mots de passe par défaut. Toutes les intrusions sont alors possibles.

Comme celui de *Administrateur*, les mots de passe SYS et SYSTEM doivent figurer dans une enveloppe scellée, car ce sont les seuls accès pour administrer les bases.

Les besoins d'accès de l'administrateur Oracle 10g

Comme pour les autres systèmes d'exploitation, l'administrateur de données Oracle 10g sur un serveur Windows doit disposer de l'accès *Administrateur* pour effectuer son travail quotidien. Dans des situations extrêmes de « crash » ou de dysfonctionnement, ces droits sont indispensables pour restaurer rapidement l'environnement de production. Ces accès sont utiles pour les tâches suivantes :

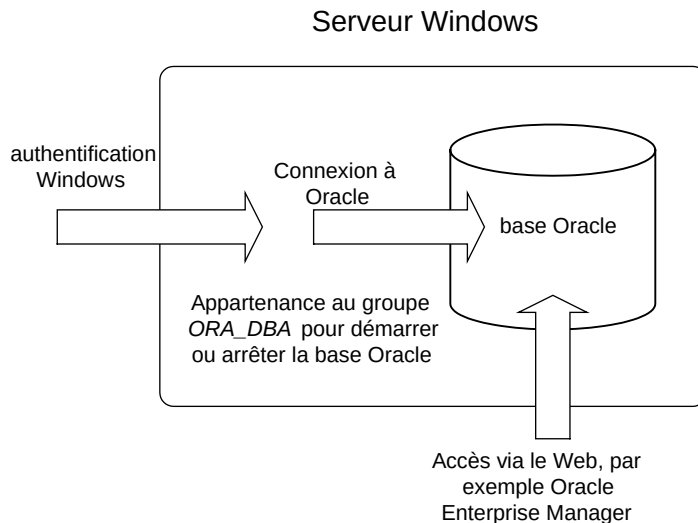
- installer/supprimer/modifier les logiciels Oracle ;
- démarrer/arrêter/modifier les bases Oracle 10g ;
- planifier des tâches système ;
- arrêter le serveur Windows à distance ;
- effectuer des sauvegardes/restaurations ;
- modifier des paramètres système, des droits d'accès, etc.

La connexion « standard » à Oracle 10g

Pour se connecter à une base Oracle 10g sur un serveur, il y a deux frontières à franchir : la connexion au système d'exploitation, puis à la base de données.

Figure 24-1

*Appartenance au groupe
ORA_DBA pour démarrer
Oracle 10g*



Le nom d'utilisateur et le mot de passe Oracle 10g donnent les droits nécessaires pour se connecter à la base Oracle 10g en mode client-serveur ou via le Web. Pour démarrer/arrêter la base Oracle 10g, il faut disposer de droits plus étendus.

Droits étendus pour démarrer/arrêter Oracle 10g

La création, le démarrage, l'arrêt ou la restauration d'une base sont des actions très importantes. Par souci de sécurité, elles nécessitent que les utilisateurs qui les effectuent possèdent des privilèges spéciaux. Comme le démarrage d'une base s'effectue base fermée, il n'est pas possible de conserver ces droits dans la base ; c'est pourquoi ils reposent sur des droits vérifiés et stockés hors de la base de données.

C'est là qu'interviennent les groupes : lors de l'installation d'Oracle, un groupe Windows spécifique à Oracle sera créé. Ce groupe sera aussi affecté à des utilisateurs Windows particuliers devant démarrer ou arrêter Oracle. C'est donc au niveau du système d'exploitation que sont gérés des droits essentiels, comme ceux qui permettent à un utilisateur Windows d'installer les logiciels ou de créer une base de données.

Une autre possibilité consiste à s'appuyer sur un fichier mot de passe extérieur à la base de données. Ce point est étudié dans ce chapitre.

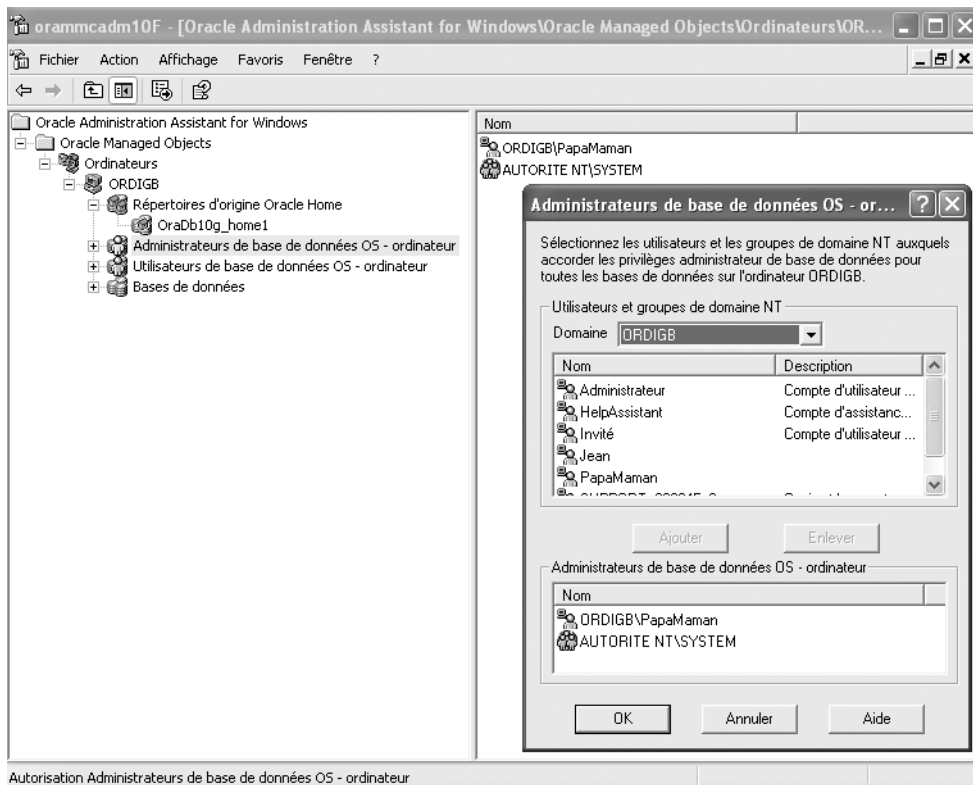


Figure 24-2

Bénéficiaires du droit

Pour autoriser un utilisateur Windows à effectuer pour toutes les bases sur un serveur Windows des opérations telles que leur création, leur démarrage, leur arrêt, etc., le plus simple est d'utiliser l'assistant Oracle prévu à cet effet :

1. lancez l'assistant : Démarrer > Programmes > Oracle - OraDb10g_home1 > Configuration and Migration Tools > *Administration Assistant for Windows*.
2. effectuez un clic droit sur l'option **Administrateur de bases de données OS – ordinateur**.
3. ajoutez / retirez les utilisateurs devant posséder ce droit (voir figure 24-2).

Le droit est automatiquement créé durant l'installation d'Oracle. Les utilisateurs appartenant à ce groupe reçoivent de la base de données le privilège SYSDBA qui regroupe tous les privilèges de la base de données.

Dans l'exemple précédent, seuls les utilisateurs ORDIGB\PapaMaman (et oui, ordinateur familial oblige !) et AUTORITE NT\SYSTEM possèdent le droit d'agir sur une base. Le premier car c'est le compte possédant les droits administrateur Windows qui a servi à installer Oracle, le second car c'est le compte sous lequel s'exécutent les *services* Windows. Dans cet exemple, le compte Administrateur Windows ne pourra ni arrêter, ni relancer aucune des deux bases Oracle créées sur ce serveur.

L'assistant permet aussi d'attribuer le privilège SYSOPER qui est un sous-ensemble de SYSDBA. Pour les utilisateurs qui le possèdent, il permet des opérations comme le démarrage et l'arrêt de la base, mais pas sa création ni la visualisation de son contenu, réservés aux utilisateurs Windows possédant le droit SYSDBA. Pour donner le droit SYSOPER :

1. lancez l'assistant : Démarrer > Programmes > Oracle - OraDb10g_home1 > Configuration and Migration Tools > *Administration Assistant for Windows*.
2. effectuez un clic droit sur l'option **Utilisateur de bases de données OS – ordinateur**.
3. ajoutez / retirez les utilisateurs devant posséder ce droit.

Le terme **Utilisateur** est troublant. On ne parle pas à ce niveau d'utilisateurs de la base Oracle pour lesquels aucune définition de droit Windows n'est nécessaire mais de l'utilisateur SYSOPER qui permet d'arrêter / relancer une base sans en être administrateur et ayant la capacité d'en visualiser le contenu.

En jouant sur ces privilèges, on peut ainsi offrir un contrôle total ou limité aux administrateurs Oracle :

- un administrateur Oracle possèdera le privilège SYSDBA ;
- un opérateur en charge de la maintenance aura uniquement le privilège SYSOPER.

Pour assurer cette vérification extérieure à la base, il existe sous Windows deux possibilités :

- hériter d'un droit spécial attribué par Windows. C'est l'option par défaut qui vient de vous être présentée ;
- attribuer le privilège SYSDBA ou SYSOPER à des utilisateurs de la base Oracle (les mots de passe sont alors cachés dans un fichier).

Ce sont les programmes utilisés pour démarrer une base, SQL*Plus ou Enterprise Manager, qui vérifient ce droit et permettent de réaliser un `CONNECT AS SYSDBA` ou `CONNECT AS SYSOPER` pour démarrer/arrêter une base.

Si l'utilisateur Windows a reçu ce droit, lorsqu'il exécute un `CONNECT AS SYSDBA` ou `CONNECT AS SYSOPER`, l'outil constate qu'il dispose de droits étendus et le connecte sans demander de mot de passe supplémentaire.

Cette méthode n'est valable que si une session Windows est ouverte sur le serveur hébergeant la base. Elle ne permet pas de démarrer une base à distance en client-serveur ou par le Web.

Autre méthode, si vous disposez d'un « fichier mot de passe », lorsque vous exécutez un `CONNECT AS SYSDBA` ou `SYSOPER`, un mot de passe est exigé puis vérifié dans le fichier caché. Cette méthode assure une connexion sécurisée pour démarrer/arrêter des bases à distance en client-serveur ou via le Web.

L'appartenance au groupe `ORA_DBA` permet aux utilisateurs Windows de démarrer et arrêter les bases présentes sur leur serveur. Les seuls utilisateurs appartenant à ce groupe doivent être *Oracle* et les utilisateurs Windows autorisés à démarrer/arrêter les bases.

Configuration standard s'appuyant sur le groupe Windows `ORA_DBA`

Vous devez suivre cette méthode pour autoriser un utilisateur Windows « quelconque » à démarrer/arrêter les bases Oracle.

1. Connectez-vous sur votre serveur Windows. Assurez-vous que l'utilisateur *Administrateur* possède le groupe `ORA_DBA` (voir figure précédente).
2. Dans les paramètres d'initialisation de la base, vérifiez que le paramètre `REMOTE_LOGIN_PASSWORDFILE` possède la valeur `NONE`.

Si vous utilisez un fichier d'initialisation `initSID.ora` :

```
# fichier initORCL.ora
...
REMOTE_LOGIN_PASSWORDFILE = NONE
...
```

Si vous utilisez un fichier persistant `spfileSID.ora` :

```
# connectez-vous sous SQL*Plus
sqlplus system/manager
show parameters remote_login_passwordfile
```

3. Démarrez la base de données sans avoir besoin de fournir un mot de passe :

```
sqlplus /nolog
connect / as SYSDBA
startup

connect / as SYSOPER
```

Le groupe `ORA_DBA` est-il satisfaisant ?

L'utilisation du groupe `ORA_DBA` est très satisfaisante en termes de souplesse d'utilisation et de sécurité. Elle permet, sous couvert d'une connexion Windows, de démarrer/arrêter les bases Oracle 10g sans fournir de mot de passe supplémentaire. C'est la technique à employer dans les scripts pour démarrer/arrêter les bases Oracle 10g.

L'exemple suivant est utilisé dans un fichier de commande du chapitre 26, *La sauvegarde d'une base Oracle 10g*. Les commandes transmises à SQL*Plus effectuent un `CONNECT AS SYSDBA`, arrêtent puis redémarrent la base et quittent l'outil SQL*Plus :

```
# connexion sous Administrateur
# l'utilisateur appartient au groupe Windows ORA_DBA

C:\> sqlplus /nolog @fichierstart.sql

Le fichier start.sql contient :

# connexion SYSDBA sans fournir de mot de passe
connect / as SYSDBA
shutdown immediate;
startup pfile=C:\oracle\admin\TEST\pfile\initTEST.ora
exit;
```

Cette méthode est la plus simple pour écrire des scripts SQL sans exposer de mot de passe.

Les privilèges `SYSDBA` et `SYSOPER`

Pour accomplir des tâches d'administration sur une base Oracle 10g, différents accès sont possibles :

- se connecter à `AS SYSOPER` ;
- se connecter à `AS SYSDBA`.

`SYSOPER` et `SYSDBA` sont des privilèges Oracle 10g particuliers qui permettent à un utilisateur Oracle 10g d'acquérir plus de pouvoirs.

`SYSOPER` est le privilège qui permet à un utilisateur Oracle 10g de démarrer/arrêter une base sans être DBA (c'est-à-dire sans détenir les droits d'un DBA Oracle 10g).

`SYSDBA` est le privilège qui permet à un utilisateur Oracle 10g de démarrer/arrêter une base en tant que rôle DBA. Le privilège `SYSDBA` est le plus puissant qui puisse exister. Il correspond aux droits de l'utilisateur `SYS`, propriétaire du dictionnaire de la base Oracle.

L'utilisation des privilèges `SYSDBA` et `SYSOPER` est maintenant une obligation car l'ancienne syntaxe `CONNECT INTERNAL` est maintenant impossible. Oracle a communiqué sur ce point depuis plusieurs années.

Attribuer les privilèges SYSDBA ou SYSOPER à un utilisateur particulier

Pour cet exemple, nous allons attribuer les privilèges SYSDBA et SYSOPER à l'utilisateur SCOTT. Cet utilisateur a déjà été créé dans la base Oracle.

Ces privilèges doivent être attribués à l'utilisateur par le compte Oracle 10g SYS ou un utilisateur Oracle possédant déjà les privilèges SYSDBA ou SYSOPER.

Pour attribuer les droits, connectez-vous à l'utilisateur sys avec l'outil SQL*Plus :

```
SQL> connect sys/change_on_install # mot de passe inchangé !
SQL> grant sysoper to scott ;
SQL> grant sysdba to scott ;
SQL> exit ;
```

L'utilisateur SCOTT possède maintenant les privilèges SYSOPER et SYSDBA.

Vérification

Connectez-vous à l'outil SQL*Plus :

```
# Vérifiez que l'utilisateur Windows ne possède pas le droit ORA_DBA
# Lancement de SQL*Plus
c:\> sqlplus /nolog

SQL> connect scott/tiger as SYSOPER
SQL> connecté
SQL> shutdown immediate
...
base de donnée arrêtée          → SYSOPER peut arrêter
...
SQL> startup
...
base de donnée démarrée         → SYSOPER peut démarrer
...
SQL> select * from dba_users ;
la table ou la vue n'existe pas  → SYSOPER n'a pas le droit DBA

SQL> connect scott/tiger as SYSDBA
SQL> connecté

SQL> shutdown immediate
...
base de donnée arrêtée          → SYSDBA peut arrêter
...
SQL> startup
...
base de donnée démarrée         → SYSDBA peut démarrer
...
SQL> select * from dba_users ;
vous avez accès aux données      → SYSDBA donne les droits DBA
SQL> exit
```

Les privilèges SYSOPER et SYSDBA permettent à n'importe quel utilisateur d'arrêter et de démarrer une base Oracle 10g. Notre exemple ne comporte pas de fichier mot de passe externe, c'est le mot de passe interne à la base qui est utilisé.

Quels sont les utilisateurs qui possèdent les privilèges SYSOPER et SYSDBA ?

Les utilisateurs Oracle ayant reçu les privilèges SYSDBA et SYSOPER peuvent être identifiés en interrogeant la vue V\$PWFILE_USERS :

```
■ select username, sysdba, sysoper from V$PWFILE_USERS ;
```

Schémas des utilisateurs connectés sous SYSOPER et SYSDBA

Lorsque vous vous connectez sous SYSDBA ou SYSOPER en utilisant le nom d'un utilisateur et son mot de passe, vous êtes connecté à un schéma par défaut et non au schéma de l'utilisateur.

Si vous êtes connecté sous SYSDBA, le schéma est celui de SYS, si vous êtes connecté sous SYSOPER, le schéma est celui de PUBLIC.

```
SQL> connect scott/tiger as SYSOPER
```

```
SQL> show user  
USER is SYS
```

```
SQL> connect scott/tiger as SYSDBA
```

```
SQL> show user  
USER is PUBLIC
```

Ce point est important car il peut entraîner des confusions. C'est ce que nous présentons dans l'exemple suivant :

```
■ # connectez-vous sous un utilisateur et créez un objet  
connect scott/tiger  
create table TEST (coltest varchar2(10)) ;  
# la table TEST est propriété du schéma SCOTT
```

```
# connectez-vous avec le privilège SYSDBA  
connect scott/tiger AS SYSDBA
```

```
# l'ordre suivant retourne une erreur  
select * from TEST ;
```

```
# l'ordre suivant est correct  
select * from scott.TEST ;
```

Pour éviter toute confusion, ne créez jamais d'objet lorsque vous êtes connecté avec les privilèges SYSDBA ou SYSOPER. Ils seraient créés sous un autre schéma que celui sous lequel vous pensez être connecté : SYS si vous êtes connecté en SYSDBA, PUBLIC si vous êtes connecté en SYSOPER.

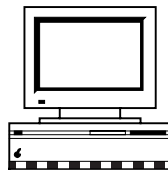
Utilisation d'un fichier mot de passe

Le système de fichier mot de passe a été conçu par Oracle pour permettre le démarrage et l'arrêt des bases de données à partir de postes distants.

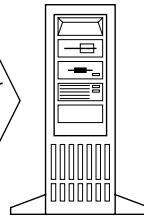
Si ces postes ne sont pas connectés au serveur Windows, ils ne peuvent hériter du groupe *ORA_DBA* et c'est au fichier mot de passe d'assurer l'autorisation d'accès à la base lorsqu'elle est fermée.

Dans ce cas, le fichier mot de passe contient les informations concernant les utilisateurs possédant les privilèges SYSDBA et SYSOPER.

Machine cliente



Accès client serveur
ou Web



Machine serveur

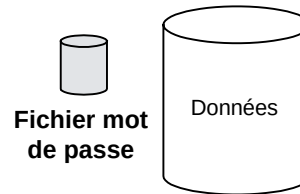


Figure 24-3

Droits nécessaires pour démarrer une base par le Web

Si le mode client-serveur est effectivement obligatoire pour démarrer à distance une base Oracle 10g sur un serveur Windows, ce n'est pas le cas pour Oracle 10g sous Windows. En effet, Windows 2000 Server permet de se connecter à distance par une session *TSE* (*Terminal Server Edition*) et d'administrer aussi bien Windows qu'Oracle 10g. Les opérations d'administration ne nécessitent plus de se déplacer jusqu'à la console du serveur pour être effectuées : c'est une fonctionnalité attendue depuis longtemps !

Créer un fichier mot de passe

Principe

La commande *orapwd* crée le fichier mot de passe. Les paramètres attendus sont les suivants :

```
orapwd file=<fichier> password=<mot de passe> entries=<nombre>
avec :
file : nom du fichier contenant les mots de passe
password : mot de passe actuel de l'utilisateur SYS
entries : nombre maximum de DBA déclarés dans ce fichier
```

Il faut ensuite indiquer à la base de données qu'elle doit utiliser un fichier mot de passe et non le groupe *ORA_DBA*. C'est la fonction du paramètre *REMOTE_LOGIN_PASSWORDFILE* présent dans le fichier d'initialisation (permanent ou non) *initSID.ora* de la base. Ce paramètre accepte trois valeurs :

- **NONE** : indique que la possibilité de démarrer/arrêter Oracle 10g n'est offerte qu'aux utilisateurs appartenant au groupe Windows *ORA_DBA*. C'est la valeur par défaut ;
- **EXCLUSIVE** : valeur qui indique que l'on utilise le fichier mot de passe ;
- **SHARED** : valeur réservée aux configurations de type cluster. Lorsque plusieurs instances partagent un fichier mot de passe commun.

Pour prendre en compte le changement de cette valeur, arrêtez puis démarrez la base de données. Vous pouvez ensuite ajouter de nouveaux administrateurs dont les mots de passe seront inscrits à la fois dans le dictionnaire de données de la base et dans le fichier mot de passe.

Une fois créé, le fichier mot de passe est caché. Modifiez le paramétrage du navigateur pour le visualiser, comme décrit au chapitre 4, *Environnement Windows : l'indispensable*.

Démarche à suivre pour utiliser un fichier mot de passe

1. Connectez-vous sous l'utilisateur Windows *Administrateur*.
2. Fixez la variable *ORACLE_SID* pour indiquer la base pour laquelle vous créez un fichier mot de passe.

```
Set ORACLE_SID=ORCL
```

3. Créer un fichier mot de passe à l'aide de l'utilitaire *orapwd*. Le mot de passe correspond à celui de l'utilisateur *SYS*.

```
# attention, le nom du fichier est important !  
# il doit respecter la syntaxe pwdSID.ora(SID=ORCL dans cet exemple)
```

```
orapwd file= C:\oracle\product\10.1.0\db_1\database\PWDORCL.ORA  
password=mot_de_passe_SYS entries=10
```

4. Dans le fichier d'initialisation de la base, attribuez la valeur **EXCLUSIVE** au paramètre *REMOTE_LOGIN_PASSWORDFILE* :

```
# fichier initORCL.ora  
  
...  
REMOTE_LOGIN_PASSWORDFILE = EXCLUSIVE  
...  
# ou modifiez le fichier persistant spfileORCL.ora  
# connectez-vous à SQL*Plus  
connect system/manager  
alter system set remote_login_passwordfile=EXCLUSIVE  
scope=SPFILE ;
```

5. Arrêtez puis relancez la base de données en fournissant le mot de passe défini précédemment :

```
sqlplus /nolog
connect SYS/mot_de_passe_SYS as SYSDBA
startup
exit
```

6. Essais de démarrage/arrêt en utilisant le fichier mot de passe :

```
# la connexion directe n'utilise pas le fichier mot de passe
# les deux syntaxes suivantes sont possibles
SQL> connect / as sysdba
Connecté.
SQL> connect system/mot_de_passe as sysdba
Connecté.
```

```
# la connexion client-serveur utilise le fichier mot de passe
SQL> connect sys/mot_de_passe@alias as sysdba
Connecté.
```

```
# vérification des utilisateurs inscrits
# dans le fichier mot de passe
SQL> select * from v$pwfile_users;
```

USERNAME	SYSDBA	SYSOPER
INTERNAL	TRUE	TRUE
SYS	TRUE	TRUE

```
# vérification avec SYSTEM qui ne figure pas dans
# le fichier mot de passe
SQL> connect system/mot_de_passe@alias as sysdba
ERROR:
ORA-01031: insufficient privileges
```

```
SQL> connect system/mot_de_passe as sysdba
Connecté.
```

```
SQL> grant sysdba to system;
```

Autorisation de privilèges (GRANT) acceptée.

```
# Vérification de l'ajout de SYSTEM dans le fichier mot de passe
SQL> select * from v$pwfile_users;
```

USERNAME	SYSDBA	SYSOPER
INTERNAL	TRUE	TRUE
SYS	TRUE	TRUE
SYSTEM	TRUE	FALSE

```
SQL> connect system/mot_de_passe@alias as sysdba
Connecté.
```

7. Autorisez l'utilisateur SCOTT à démarrer/arrêter la base :

```
sqlplus system/mot_de_passe

grant SYSDBA to SCOTT ;

select * from v$pwfile_users;

  USERNAME                SYSDBA SYSOPER
  -----
INTERNAL                  TRUE  TRUE
SYS                      TRUE  TRUE
SYSTEM                   TRUE FALSE
SCOTT                   TRUE FALSE

Exit
```

8. Essai de démarrage/arrêt sous l'utilisateur SCOTT :

```
sqlplus /nolog

connect scott/tiger as SYSDBA ;
shutdown immediate
startup
exit
```

9. Essai de démarrage/arrêt sous l'utilisateur SCOTT en client-serveur :

```
sqlplus scott/tiger@alias AS SYSDBA
shutdown immediate
startup
```

Vous avez démarré la base de données en fournissant un mot de passe. Utilisez cette méthode pour démarrer les bases à distance en saisissant de manière interactive le mot de passe. Ne l'utilisez pas dans les scripts car dans ce cas les mots de passe sont exposés !

Une base utilise-t-elle un fichier mot de passe ?

Pour vérifier si une base utilise un fichier mot de passe, vérifiez sa présence en ORACLE_HOME/database et son utilisation avec l'ordre SQL :

```
sqlplus
SQL> show parameters

# consultez la valeur du paramètre remote_login_passwordfile
```

Faut-il sauvegarder le fichier mot de passe ?

Chaque fois qu'un utilisateur recevra le privilège SYSDBA ou SYSOPER, le fichier mot de passe sera mis à jour. Si vous utilisez cette possibilité, il faut donc intégrer le fichier mot de passe dans vos sauvegardes. Le fichier est situé en ORACLE_HOME/database.

Comment tester votre sécurité ?

Vous avez suivi nos conseils et désirez tester rapidement la sécurité de la configuration Oracle 10g sur Windows. Vérifiez :

- si, dans les fichiers de démarrage et d'arrêt d'Oracle 10g, des mots de passe sont exposés ;
- si, dans les fichiers d'administration (sauvegarde, batch...) d'Oracle 10g, des mots de passe sont exposés ;
- si seul l'utilisateur Windows *Administrateur* possède les droits de lecture / écriture sur tous les fichiers de l'installation Oracle 10g ;
- que le mot de passe de l'utilisateur *system* ne soit pas *manager* ;
- que le mot de passe de l'utilisateur *sys* ne soit pas *change_on_install* ;
- que les utilisateurs Oracle 10g n'aient pas des mots de passe trop simples, par exemple *toto/toto* ;
- que vous pouvez modifier les mots de passe Windows et Oracle 10g sans contrainte.

Enfin, de très nombreux utilitaires, comme *Satan*, permettent de tester la sécurité d'accès au serveur Windows. Ce sont des outils conçus à la fois pour pénétrer dans le système et pour en tester la sûreté. Ils vous seront très utiles.

Résumé

La sécurité est primordiale pour une base de données. Ce chapitre fournit quelques conseils pratiques pour établir, selon vos besoins, la meilleure sécurité possible.

Il est possible d'aller plus loin en ce sens et de montrer comment les rôles d'Oracle 10g s'intègrent dans la gestion de groupes Windows, mais nous avons préféré insister sur les principes fondamentaux. Une fois ces bases assimilées, l'intégration d'Oracle 10g à d'autres options de sécurité de Windows ne vous posera pas de difficultés.

Oracle Enterprise Manager

Dans ce chapitre :

- les objectifs d'Oracle Enterprise Manager (OEM) ;
- l'architecture des différentes versions d'Oracle Enterprise Manager
- la présentation des grandes fonctions d'OEM au travers des onglets *Page d'Accueil*, *Performance*, *Administration* et *Maintenance* ;
- les faiblesses d'Oracle Enterprise Manager.

Oracle Enterprise Manager est un environnement d'administration et de supervision de nouvelle génération, développé d'une manière modulaire et ouverte. Il est très complet et permet d'envisager des tâches difficiles en toute sérénité. Nous décrivons les différents composants, ainsi que leur mise en œuvre et leur utilisation.

Les réserves concernant les précédentes versions d'OEM étaient nombreuses. Son évolution étant particulièrement significative, nous vous conseillons très fortement de le découvrir et de l'utiliser.

Objectifs d'Oracle Enterprise Manager

Il y a quelques années, les administrateurs de bases de données Oracle détestaient Oracle Enterprise Manager (OEM), qui était perçu comme un *truc* tout juste utile pour les débutants incapables de maîtriser la syntaxe Oracle. Les administrateurs expérimentés préféraient la facilité et la pertinence d'ordres SQL lancés à partir de SQL*Plus. Chacun d'eux se constituait sa propre bibliothèque d'ordres SQL d'administration, de tuning, ainsi que

des scripts de surveillance et d'administration. Mes ouvrages en sont le parfait exemple : depuis la première version, publiée en 1998, chacun de mes cinq livres était accompagné en annexe d'une bibliothèque de scripts. Ceci est du passé.

Avec Oracle 10g, OEM a franchi une étape majeure. Entre ses humbles débuts où il n'était qu'un simple habillage graphique d'ordres SQL, sans réelle valeur ajoutée, Oracle a très fortement investi pour en faire un produit ambitieux et robuste.

En accompagnant les perfectionnements du matériel, les bases de données évoluent selon plusieurs axes. Les entreprises sont de plus en plus demandeuses de bases plus nombreuses, d'une taille plus importante ou encore plus distribuées, parfois les trois en même temps. Les contraintes de disponibilité et de performance augmentent elles- aussi.

Oracle Enterprise Manager (OEM) est un système d'administration de nouvelle génération qui propose une réponse intégrée à toutes ces exigences. Il est développé d'une manière modulaire et ouverte. Une console d'administration Web constitue une « tour de contrôle » à partir de laquelle vous pouvez lancer des utilitaires spécialisés dans les diverses tâches à accomplir.

Pour les nouveaux produits Oracle, des utilitaires sont progressivement intégrés à Oracle Enterprise Manager pour les administrer dans un environnement cohérent et évolutif. D'autres logiciels sont ainsi administrables à travers cette interface modulaire : le serveur LDAP Oracle, les serveurs d'application d'Oracle (Oracle iAS), les accès sécurisés par clé de cryptage, etc.

C'est un outil majeur dont l'envergure dépasse celle des produits proposés par les concurrents d'Oracle. Il est conçu pour gérer les grands systèmes aussi bien que les petits.

Dernier avantage et non des moindres, cet outil est gratuit, livré en standard avec les versions d'Oracle 10g et fonctionne dans de nombreux environnements. Seuls des modules optionnels sont facturés.

Depuis la console Web d'Enterprise Manager, un gestionnaire de système peut :

- accéder à distance de façon sécurisée à toutes les bases, à d'autres logiciels Oracle ainsi qu'aux serveurs ;
- administrer, gérer la performance et superviser plusieurs bases et serveurs ;
- surveiller certains points et être prévenu lorsque des anomalies surviennent ;
- avant l'apparition des packages Oracle DBMS_JOB, vous deviez utiliser un outil de planification du système d'exploitation (*at* sous Windows, *cron* sous Linux) ou un outil tiers pour planifier et suivre l'exécution de tâches sur des serveurs. OEM propose une interface unique pour contrôler les scripts, tant pour la base que pour le serveur ;
- depuis sa première apparition avec Oracle 8i, STATSPACK a été considérablement amélioré. OEM s'interface avec le Workload Repository History d'Oracle 10g, ce qui permet de disposer instantanément d'indicateurs graphiques de performances sans qu'il soit nécessaire de lancer des ordres SQL complexes ;

- Oracle Automatic Workload Repository (AWR) analyse la base et vous propose des axes d'amélioration des performances, que vous pouvez accepter. OEM les mettra en œuvre pour vous ;
- depuis un assistant personnel (PDA), EM2go vous permet d'administrer les bases, ainsi que de recevoir des alertes. La mobilité est enfin possible pour les administrateurs Oracle ;
- sous le nom incroyable de *fetchlets*, OEM permet d'exécuter toutes sortes de scripts (SQL, Linux, Windows) ou n'importe quel programme du serveur ;
- de nombreux utilitaires présentés dans cet ouvrage (Import, Export, Data Pump, SQL*Loader) bénéficient d'interfaces graphiques conviviales ;
- découverte de l'environnement : OEM détecte automatiquement une nouvelle ressource, un nouveau serveur et le place dans un groupe de ressources disponibles.

Le concept de Grid apparu avec Oracle 10g est perceptible au travers d'OEM. Vous gérez un ensemble de machines, de bases de données, vous distribuez les bases sur un cluster, affectez la puissance et la disponibilité là où elles sont nécessaires, le tout depuis une interface unique. Vous êtes enfin capable de gérer et d'administrer un ensemble puissant construit à partir de composants multiples d'un faible coût unitaire.

Vous constaterez qu'étant 100 % Web, vous n'avez plus besoin d'installer de client lourd sur le poste client pour administrer les bases Oracle à distance.

L'avantage majeur d'Oracle Enterprise Manager est de transformer une administration traditionnellement réactive en une administration proactive.

Dans ce chapitre, nous expliquerons les concepts d'Oracle Enterprise Manager, ses composants, ainsi que la façon de procéder pour mettre en œuvre un environnement d'administration.

La documentation d'Oracle Enterprise Manager est très bien faite. Pour en découvrir l'utilisation courante, je vous conseille le document « 2 days DBA » qui vous présente les tâches courantes d'administration qui peuvent toutes être effectuées depuis Oracle Enterprise Manager. Il est accessible depuis le menu Démarrer > Programmes > Oracle - OraDb10g_home1 > Release Documentation.

Architecture d'Oracle Enterprise Manager

La nouvelle version d'Oracle Enterprise Manager est maintenant 100 % Web, accessible depuis un simple navigateur, sans installation longue et fastidieuse sur le poste client. Si vous désirez être encore plus mobile, l'accès par un assistant personnel à EM2Go est disponible. Désormais, vous pouvez vous trouver à l'autre bout du monde et administrer une base Oracle.

Les différentes versions d'Oracle Enterprise Manager

Fidèle à ses habitudes, Oracle propose plusieurs configurations pour ses logiciels. Cela permet, dans la version de base, d'installer et de configurer rapidement OEM et, dans la

version plus puissante, au prix d'une configuration spécifique, d'accéder simultanément à plusieurs bases.

Dans les deux cas, des options supplémentaires facturées sont proposées.

Les versions d'Oracle Enterprise Manager sont :

- Oracle Enterprise Manager **Database Control** ;
- Oracle Enterprise Manager **Database Grid Control** ;
- Il existe d'autres versions d'OEM, par exemple pour administrer les serveurs d'application Oracle (Internet Application Server iAS).

Oracle Enterprise Manager **Database Control** est la version de base d'OEM qui ne permet d'accéder qu'à une seule base de données ou aux deux bases d'un cluster. Si vous avez deux bases de données sur un serveur, vous aurez deux versions du Database Control qui seront lancées simultanément.

Oracle Enterprise Manager **Database Grid Control** est une version qui nécessite une architecture spécifique qu'il convient de mettre en place. Elle permet d'accéder simultanément à toutes les bases Oracle, y compris celles depuis la version 8i.

Oracle Enterprise Manager Database Grid Control est disponible sur un CD-Rom différent du CD-Rom unique qui permet d'installer la base. Vous pouvez le télécharger depuis <http://otn.oracle.com>.

Détaillons maintenant ces deux options.

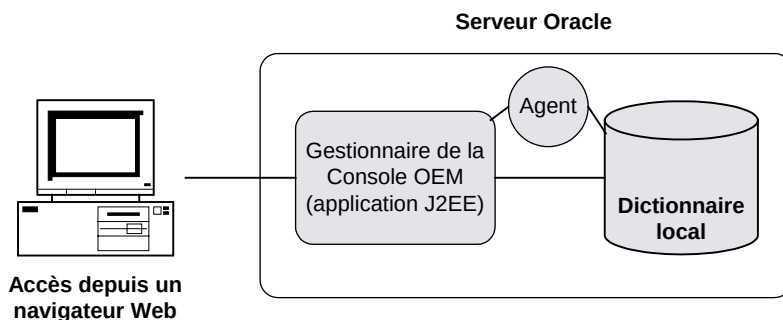
Oracle Enterprise Manager Database Control

OEM Database Control nécessite plusieurs composants pour administrer une base :

- le gestionnaire de la console OEM ;
- un agent chargé de remonter des informations du serveur et d'exécuter des tâches ;
- un dictionnaire local, chargé de conserver des informations relatives à la base accédée.

Figure 25-1

Architecture d'OEM
Database Control



Si deux bases sont présentes sur le même serveur, cette architecture sera doublée. Dans ce cas, vous aurez intérêt à ne démarrer les composants *gestionnaire de la console* et *agent* que si la base correspondante est démarrée, afin de limiter la consommation de ressources inutiles.

Dans l'exemple suivant sous Windows (le fonctionnement sous Linux est comparable), seule la base PROD est lancée, grâce aux services Windows OracleServicePROD, le service Agent OracleJobSchedulerPROD ainsi que la console OracleDBConsolePROD. Les services similaires correspondants à la base TEST sont arrêtés.

 OracleDBConsolePROD		Démarré	Automatique
 OracleDBConsoleTEST			Manuel
 OracleJobSchedulerPROD		Démarré	Automatique
 OracleJobSchedulerTEST			Manuel
 OracleOraDb10g_home1ISQL*Plus	iSQL*Plus ...	Démarré	Automatique
 OracleOraDb10g_home1SNMPPeerEncapsulator			Manuel
 OracleOraDb10g_home1SNMPPeerMasterAgent			Manuel
 OracleOraDb10g_home1TNSListener		Démarré	Automatique
 OracleServicePROD		Démarré	Automatique
 OracleServiceTEST			Manuel

Figure 25-2
Services d'OEM sous Windows

Le gestionnaire de la console OEM

Le gestionnaire de la console OEM est une application J2EE (Java to Enterprise Edition) qui assure le lien entre la base Oracle, l'agent, et réalise toute la présentation au format HTML pour le client.

Cette architecture n'utilise par de serveur Web (Apache ou Microsoft IIS). Le gestionnaire de la console OEM dialogue directement avec le client par un port réseau spécifique. Vous devrez indiquer ce port dans l'URL d'accès. Par exemple : <http://monserveur:5500/em>. Si deux bases sont présentes sur un même serveur, le port d'écoute sera différent, par exemple <http://monserveur:5501/em> pour la seconde base.

Si une seule base est installée par serveur, se souvenir de l'URL d'accès à OEM est relativement simple : <http://monserveur:5500/em>. Mais comme le numéro de port (5500 par défaut) peut être modifié et que plusieurs bases peuvent être présentes sur un serveur, ces contraintes poussent rapidement à mettre en place l'architecture d'OEM Database Grid Control, beaucoup plus souple pour naviguer de base à base et de serveur à serveur.

Oracle Enterprise Manager Database Grid Control

Le Database Grid Control permet l'accès simultané à plusieurs serveurs et bases de données, pour en assurer l'administration et le suivi depuis un point unique. Cela rend aussi possible la centralisation des tâches qui seront lancées sur les différentes bases et serveurs distants.

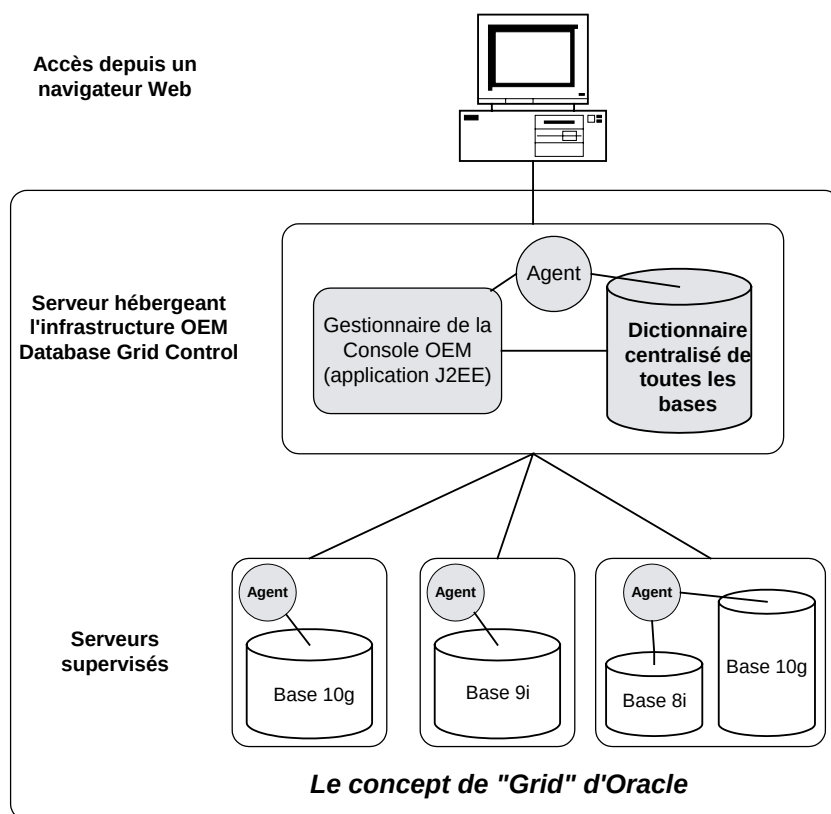


Figure 25-3

Architecture d'OEM Database Grid Control

Le concept de Grid annoncé par Oracle est perceptible dans cette architecture : gérer simplement une architecture puissante composée d'une multitude de serveurs de faible coût unitaire.

Vous retrouvez les composants qui vous ont été présentés précédemment. Une différence importante : une base Oracle centralise maintenant toutes les informations de l'infrastructure supervisée. Par exemple, l'historique des performances, les connexions aux serveurs et bases distantes sont conservées dans le dictionnaire centralisé d'OEM.

Sur les serveurs supervisés, le gestionnaire de la console OEM n'est plus nécessaire dans cette architecture. Il peut toujours être présent, au cas où vous voudriez à la fois profiter d'OEM Database Grid Control et, dans certains cas, d'un accès direct via OEM Database Control. Par contre, l'agent est nécessaire pour tirer pleinement partie de cette architecture.

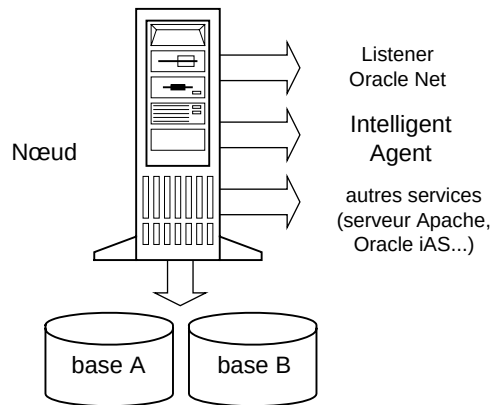
Cette architecture est la seule qui permette de superviser les précédentes versions d'Oracle (8, 8i, 9i) à partir d'OEM 10g.

Les nœuds administrés

Le dernier tiers de l'architecture OEM est constitué des nœuds administrés : leurs *Listener*, bases, Agent, ainsi que d'autres services comme le serveur Apache livré par Oracle. Les caractéristiques et performances du serveur peuvent elles aussi être suivies.

Figure 25-4

Composants OEM d'un nœud



Un nœud possède généralement une ou plusieurs bases et un Listener Oracle Net. La présence de l'Intelligent Agent est facultative. Dans ce cas, certaines possibilités seront limitées.

Les agents intelligents

Pour effectuer une découverte automatique ou lancer des *jobs* et des *events* sur des machines distantes, Oracle utilise des agents de communication. Ils doivent être présents et lancés sur toutes les machines distantes.

Le Management Server envoie les ordres aux agents distants **responsables de l'exécution des jobs et de la surveillance des événements demandés**. L'agent est, de plus, chargé de remonter vers le serveur toutes les informations sur le déroulement des *jobs* et des *events*.

Note : L'agent intelligent est indispensable dans plusieurs cas dont la phase de découverte des machines. Après la découverte, vous pouvez décider d'arrêter les agents intelligents sur certaines machines. La base sera toujours accessible mais de nombreuses possibilités ne seront pas disponibles.

Le système de jobs

Enterprise Manager permet de définir des jobs ou « travaux » à accomplir sur des sites distants. Par exemple, un administrateur peut planifier l'exécution simultanée sur plusieurs sites d'une sauvegarde des bases de données, à une fréquence déterminée. Enterprise Manager se charge de transmettre les travaux à effectuer aux agents intelligents des machines

distantes, de conserver l'historique des jobs, de visualiser ceux en cours, de contrôler leur déroulement et leur accomplissement.

Les events ou événements

Enterprise Manager permet de surveiller des points précis du système et d'être prévenu automatiquement si l'un de ces événements se produit. Ce sont les *events* ou événements. De plus, on peut associer à un événement un *fixit job* ou job correctif qui se déclenchera automatiquement dès que l'événement se produira.

Par exemple, il est possible de surveiller l'espace disque libre sur des machines et de prévenir automatiquement l'administrateur pour lui signifier qu'il va falloir ajouter un disque ou réorganiser ses données.

La remontée d'alertes

Vous pouvez paramétrer OEM pour remonter toutes sortes d'alertes par e-mail ou encore par « pager ». Il est même possible d'indiquer des fenêtres temporelles pendant lesquelles vous ne voulez pas que les alertes soient produites. Par exemple, il ne sert à rien d'être alerté chaque week-end, lorsque la base est fermée pour sa sauvegarde.

Par exemple, voici le résultat de deux alertes envoyées automatiquement par e-mail :

```
Nom=LISTENER_ORDIGB
Type=Listener
Hôte=ORDIGB
Mesure=Status
Horodatage=13 mai 2005 20 h 23 CEST
Gravité=Critique
Message=The listener is down: TNS-12541: TNS : aucun processus d'écoute .
Nom de la règle=Listener Availability
Propriétaire de la règle=SYSMAN

Nom=ORDIGB
Type=Host
Hôte=ORDIGB
Mesure=CPU Utilization (%)
Horodatage=24 avr. 2005 11 h 46 CEST
Gravité=Critique
Message=CPU Utilization is 99.98%
Nom de la règle=Host Availability and Critical States
Propriétaire de la règle=SYSMAN
```

La découverte de services

Oracle Enterprise Manager permet d'administrer les logiciels Oracle. Pour faire face à leur diversité, une découverte automatique des « services » est proposée. Il suffit de donner le nom d'une machine cible pour que l'Agent Intelligent distant explore et retourne toutes les informations nécessaires à la console sur les produits installés, bases Oracle, Listener, produits Internet, Oracle Application, etc.

Le repository ou référentiel

Le référentiel est un jeu de tables situées dans une base Oracle 10g. Le OEM Database Control ou Database Grid Control utilise ce référentiel local ou centralisé pour conserver toutes ses informations de configuration, d'envoi et de remontées d'informations des nœuds.

Les applications standard

De nombreuses applications sont livrées avec Enterprise Manager. Elles sont toutes accessibles depuis la console. Elles ne peuvent pas toutes être énumérées, leur nombre étant en constante augmentation de version en version. Certaines d'entre elles sont abordées dans ce chapitre.

Les applications optionnelles

Des applications optionnelles, c'est-à-dire facturées, existent autour d'Oracle Enterprise Manager.

Options facturées d'OEM Database Control et Database Grid Control

Dans chaque version d'OEM, toutes les options facturées d'Oracle Enterprise Manager sont installées en standard. Oracle vous indique dans un panneau d'accueil quelles sont ces options :

Database Diagnostics Pack

- Suivi des performances (base de données et serveur) ;
- ADDM (*Automated Database Diagnostic Monitor*) ;
- Suivi des performances (base de données et serveur) ;
- Automatic Workload Repository ;
- Notification d'événements: méthodes, règles et planification ;
- Historique et mesure des événements (base de données et serveur) ;
- Indisponibilités.

Database Tuning Pack

- SQL Access Advisor ;
- SQL Tuning Advisor ;
- SQL Tuning Sets ;
- Réorganisation des objets.

Configuration Management Pack

- Configuration de la base de données et du serveur ;
- Déploiements ;

- Téléchargement des patchs dans un cache et installation ;
- Clonage d'une base ;
- Clonage d'une installation Oracle (Oracle Home) ;
- Recherche dans la base ;
- Comparaison et différences de bases.

On peut regretter qu'Oracle mette sciemment ses clients dans l'illégalité : les options sont toutes installées et il est quasi impossible d'identifier si l'on utilise une option standard ou une option supplémentaire facturée.

Les éditions

La précédente version d'Oracle Enterprise Manager disposait d'une interface rigide et permettait d'appeler quelques éditions au format HTML. Tous les écrans étant maintenant au format HTML, c'est l'ensemble d'OEM qui peut être édité.

Les Groups ou groupes

Dans la version Database Grid Control, vous pouvez réunir plusieurs bases de données au sein d'un groupe. Toutes les actions de paramétrage, droits d'accès, etc., affectées en une seule opération au groupe, sont répercutées à l'ensemble des bases membres du groupe. Un *job* ou un *event* peut notamment être affecté à un groupe.

Cette caractéristique est l'une des plus puissantes et des plus productives d'Oracle Enterprise Manager.

L'installation d'Oracle Enterprise Manager

L'installation d'Oracle Enterprise Manager doit être adaptée à vos besoins. En standard, la version Database Control est installée. Pour bénéficier de la version multibase, y compris pour gérer de précédentes versions d'Oracle, installez Database Grid Control.

Par souci de simplicité, seule l'installation et l'utilisation d'OEM Database Control seront décrites dans le reste de ce chapitre. Cela ne constitue pas un handicap, car tous ces conseils s'appliquent à la version OEM Database Grid Control. La plus grande difficulté consistant à bien comprendre les différences d'architecture, qui vous ont été présentées.

Installer OEM Database Control

Tous les paramétrages nécessaires à OEM Database Control sont effectués automatiquement par Oracle lorsque :

- la création d'une base s'effectue en même temps que l'installation des logiciels ;
- vous utilisez DBCA pour créer une nouvelle base.

Ces points sont détaillés aux chapitres 13, *L'installation d'Oracle 10g sous Windows* et 14, *Création d'une base Oracle 10g*.

Si vous créez une base manuellement, sans utiliser ces assistants, l'environnement OEM Database Control ne sera pas installé. Pour l'installer, vous devrez utiliser un script *Perl* nommé *DBConsole.pm*, c'est une option complexe que je vous déconseille d'utiliser. Préférez la simplicité des assistants.

Paramétrer OEM pour utiliser les Jobs sous Windows

Les systèmes Windows (2000, XP, 2003) nécessitent des droits particuliers pour que les Jobs d'OEM puissent s'exécuter. Par défaut, l'agent est installé sous le compte *Utilisateur Local du Système*. Pour qu'il puisse exécuter les Jobs, les utilisateurs qui les soumettent doivent posséder le droit *Ouvrir une session en tant que tâche*.

Pour donner ce droit à l'utilisateur PapaMaman (et oui, toujours mon ordinateur familial !) :

1. menu démarrer > Paramètres > Panneau de configuration ;
2. sélectionnez *Outils d'administration* puis *Stratégie de sécurité locale* ;
3. dans l'option *Stratégies Locales* choisissez *Attribution des droits utilisateurs* ;
4. Ajoutez le nom utilisateur PapaMaman (par exemple) à la stratégie *Ouvrir une session en tant que tâche*.

Vérifier le fonctionnement d'Oracle Enterprise Manager

Ces vérifications ne vous seront peut-être pas utiles, mais elles seront d'un appui précieux si vous n'arrivez pas à accéder à OEM !

Pour assurer le fonctionnement d'OEM Database Control, les services Windows suivants doivent être lancés :

- OracleService<SID> : le service qui lance la base de données <SID> ;
- Oracle<Oracle_Home>TNSListener : le Listener Oracle ;
- OracleJobScheduler<SID> : l'agent d'OEM de la base <SID> ;
- OracleDBConsole<SID> : le gestionnaire de la console OEM pour la base <SID>.

Par exemple, pour la base PROD dont l'Oracle Home est *OracleOraDb10g_home1*, les services nécessaires sont : OracleServicePROD, OracleOraDb10g_home1TNSListener, OracleJobSchedulerPROD, OracleDBConsolePROD.

Pour économiser les ressources système, positionnez le démarrage automatique de ces services uniquement pour les bases dont vous avez besoin. Pour les autres, plus occasionnelles, positionnez le service en démarrage manuel.

Vérifiez localement avec SQL*Plus le fonctionnement correct :

```
connect system/mot_de_passe@PROD # PROD est un alias exemple
```

Sur ma machine de test, le Listener devait démarrer automatiquement pendant le démarrage de la base. Cela engendrait des difficultés de reconnaissance automatique de cette base par le Listener. Il m'a suffi d'arrêter puis de démarrer le Listener, en vérifiant après environ une minute de fonctionnement par la commande *lsnrctl status* que la base démarrée était bien reconnue.

Si plusieurs bases de données sont installées sur un serveur, les fichiers de configuration respectifs de chacune d'entre elles sont situés en `ORACLE_HOME\<nom de l'ordinateur>\<SID>`.

Figure 25-5

Fichiers de configuration



Pour lancer OEM Database Control, vous devez utiliser une URL :

`http://<nom_ordinateur>:<numéro_de_port>/em`

- Nom_ordinateur : le nom ou l'adresse IP de l'ordinateur ;
- Numéro_port : le port réseau sur lequel écoute le gestionnaire de la console, généralement le 5500 ;
- em : mnémonique d'Enterprise Manager.

Exemple : `http://ORDIGB:5500/em`

Si plusieurs bases sont installées, pour identifier avec certitude le numéro de port utilisé, consultez le fichier `ORACLE_HOME\install\portlist.ini`.

Par exemple :

```

Numéro de port HTTP Ultra Search =5620
Enterprise Manager Agent Port =
Numéro de port HTTP iSQL*Plus =5560
Enterprise Manager Console HTTP Port (TEST) = 5500
Enterprise Manager Agent Port (TEST) = 1830
Enterprise Manager Console HTTP Port (PROD) = 5501
Enterprise Manager Agent Port (PROD) = 1831

```

Dans l'exemple précédent, la base TEST utilise le port 5500 et la base PROD le port 5501. C'est l'assistant DBCA, lors de la création de la base PROD, qui a automatiquement configuré et incrémenté ce numéro de port. Oracle prend soin de nous !

Assurez-vous auprès de votre administrateur réseau que les ports indiqués soient bien « ouverts » sur le serveur cible. Sinon, la demande d'accès à l'URL échouera.

Pour tester le gestionnaire de la console : lancez la commande suivante depuis une invite système (cmd sous Windows, fenêtre Shell sous Linux).

```

# définissez le SID cible
set ORACLE_SID=PROD      # sous Windows
ORACLE_SID=PROD          # sous Linux

# lancez la commande
emctl status dbconsole

# la commande affiche le commentaire :
Oracle Enterprise Manager 10g Database Control Release 10.1.0.2.0
Copyright (c) 1996, 2004 Oracle Corporation. All rights reserved.
http://ORDIGB:5501/em/console/aboutApplication
Oracle Enterprise Manager 10g is running.
-----
Logs are generated in directory C:\oracle\product\10.1.0\db_1\ORDIGB_PROD\sysman\log

```

Pour autoriser l'utilisateur SYSTEM (ou tout autre utilisateur) à démarrer ou arrêter Oracle, affectez-lui les droits SYSOPER et SYSDBA, décrits au chapitre 24, *Stratégie de sécurité sous Windows* :

```

connect sys/mot_de_passe@PROD      # PROD est l'alias exemple

SQL> select * from V$PWFILE_USERS;

USERNAME                                SYSDB SYSOP
-----                                -
SYS                                     TRUE  TRUE

# SYSTEM ne figure pas dans la liste,

```

```
# affecter à SYSTEM les droits SYSOPER et SYSDBA
```

```
grant SYSOPER to SYSTEM;  
grant SYSDBA to SYSTEM;
```

```
# vérification
```

```
SQL> select * from V$PWFILE_USERS;
```

USERNAME	SYSDB	SYSOP
SYS	TRUE	TRUE
SYSTEM	TRUE	TRUE

Si vous utilisez une machine non connectée au réseau ou à Internet, vous devrez certainement modifier les options d'accès au réseau du navigateur. En effet, certaines pages d'OEM, notamment celles de l'aide, essayent d'accéder au site <http://otn.oracle.com> pour y puiser de l'information. Vous ne pourrez donc pas y accéder sans réseau.

Enfin, n'oubliez pas que le propriétaire de l'installation OEM est l'utilisateur SYSMAN. Son mot de passe a été saisi lors de la création de chaque base Oracle.

Présentation d'Oracle Enterprise Manager

Vous connaissez maintenant les grands principes du fonctionnement d'Oracle Enterprise Manager, ainsi que les différences entre le Database Control et le Database Grid Control. À partir de cette première version, étudions maintenant les principales caractéristiques d'Oracle Enterprise Manager.

Comment naviguer dans Oracle Enterprise Manager ?

La console d'Oracle Enterprise Manager est le point de contrôle unique de l'administrateur. Elle permet d'accéder à toutes les ressources Oracle 10g grâce à des assistants dédiés, le tout au travers d'une interface Web.

La console des précédentes versions d'OEM était sous forme d'un « client lourd », écrit en Java, l'accès Web n'étant qu'une option aux possibilités réduites. Avec Oracle 10g, le client lourd disparaît au profit d'un accès 100 % Web très complet.

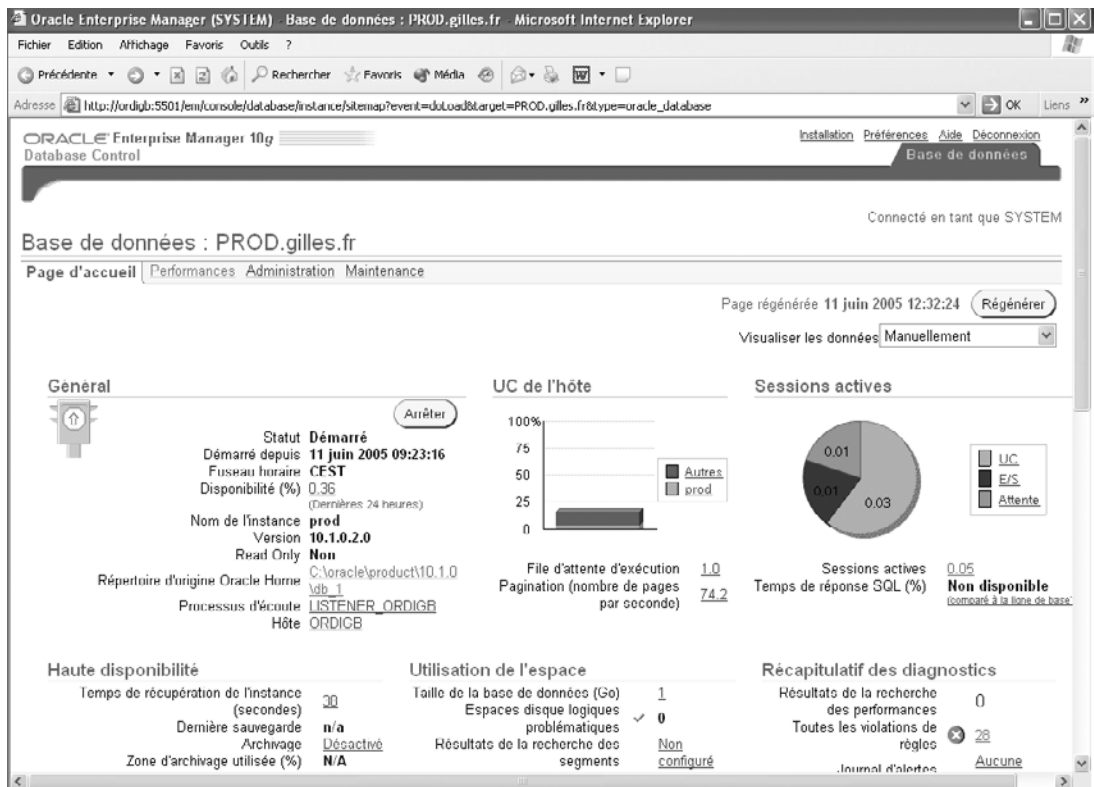


Figure 25-6
Console Web d'Oracle Enterprise Manager

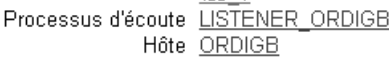
Cette console Web permet une navigation aisée. Tout d'abord, accédez par un onglet à la page qui vous intéresse. Nous présentons dans ce chapitre le contenu de chacun des quatre onglets : *Page d'Accueil*, *Performances*, *Administration* et *Maintenance*.

Figure 25-7
Navigation dans les pages
par onglet



Ensuite, dans la page choisie, vous accédez au détail d'un élément en cliquant sur son lien. Par exemple, cliquez sur [LISTENER_ORDIGB](#) ou sur [ORDIGB](#) pour accéder aux pages respectives du Listener ou du serveur.

Figure 25-8
Détail d'un élément



Enfin, en pas de page figurent le rappel des onglets ainsi que de nombreux liens utiles accessibles depuis toutes les pages d'OEM.



Figure 25-9

Bas de page de la console Web

La connexion à Oracle Enterprise Manager et la page d'accueil

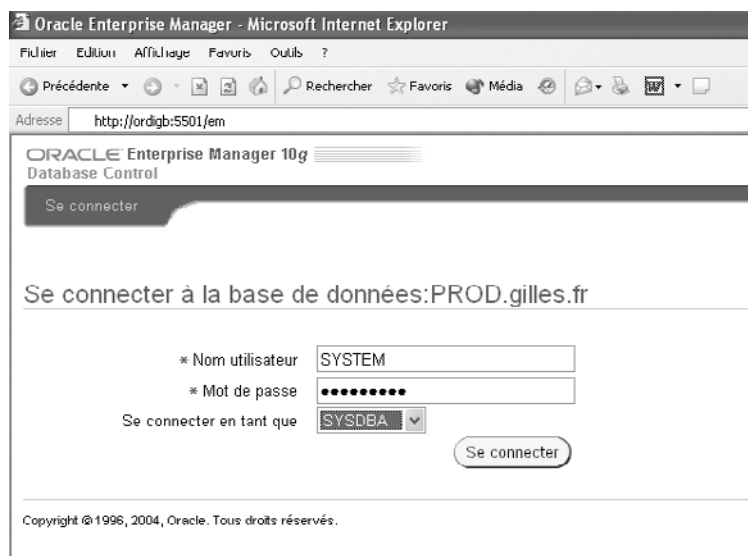
Testez tout d'abord le fonctionnement en local, sur le serveur où tous les composants fonctionnent. Essayez ensuite depuis un poste distant sur le même réseau, puis depuis un poste distant extérieur au réseau, devant traverser un dispositif de sécurité.

La connaissance des ports utilisés est primordiale pour indiquer à votre administrateur de sécurité les ports à ouvrir pour permettre l'accès à OEM au travers d'un firewall.

Si vous avez suivi le paragraphe précédent, vous pourrez vous connecter sous l'utilisateur SYSTEM, en tant que SYSDBA. Par défaut l'URL est `http://<serveur>:5500/em`.

Figure 25-10

Page de connexion à OEM



La connexion réalisée, vous accédez à la page d'accueil.

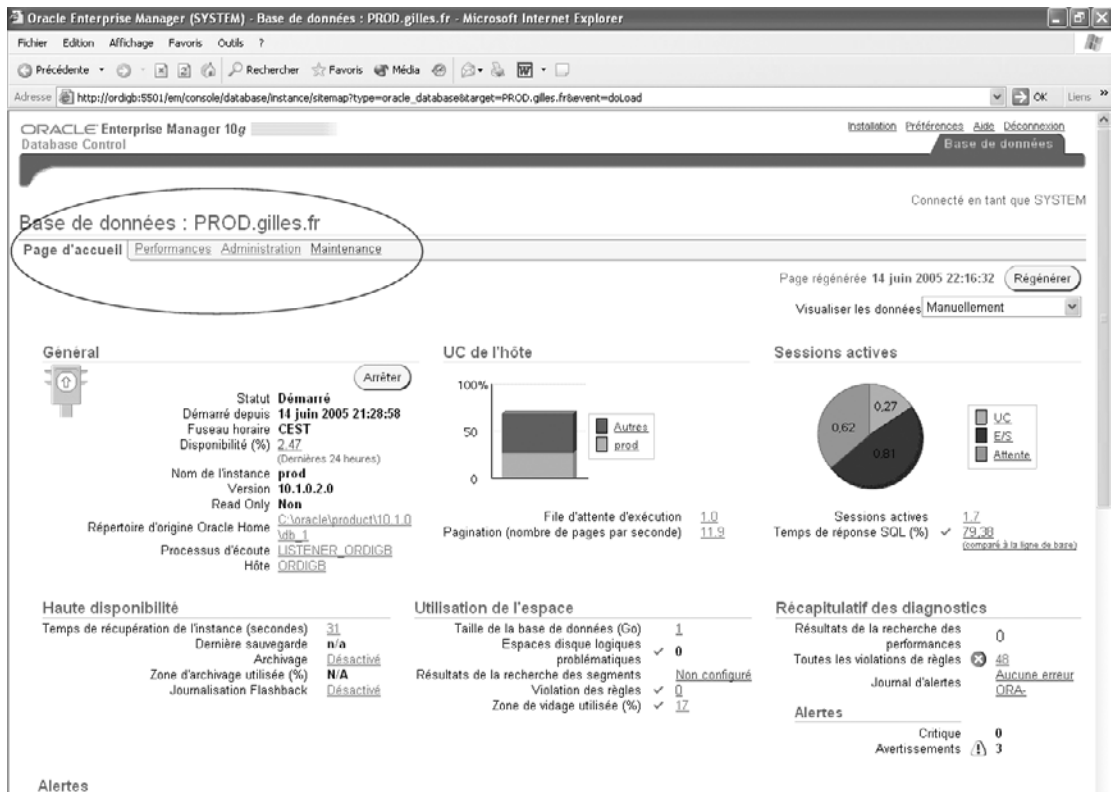


Figure 25-11
Page d'accueil d'OEM

La page d'accueil vous donne des informations généralistes concernant les performances, le système, les utilisateurs, etc. C'est une synthèse des autres pages. Elle vous permet en outre de démarrer/arrêter la base (si vous vous êtes connecté avec le privilège SYSOPER ou SYSDBA).

Par exemple, vous pouvez suivre la consommation globale mémoire et CPU du serveur : de quoi faire pâlir d'envie un administrateur système dans un environnement multi OS !

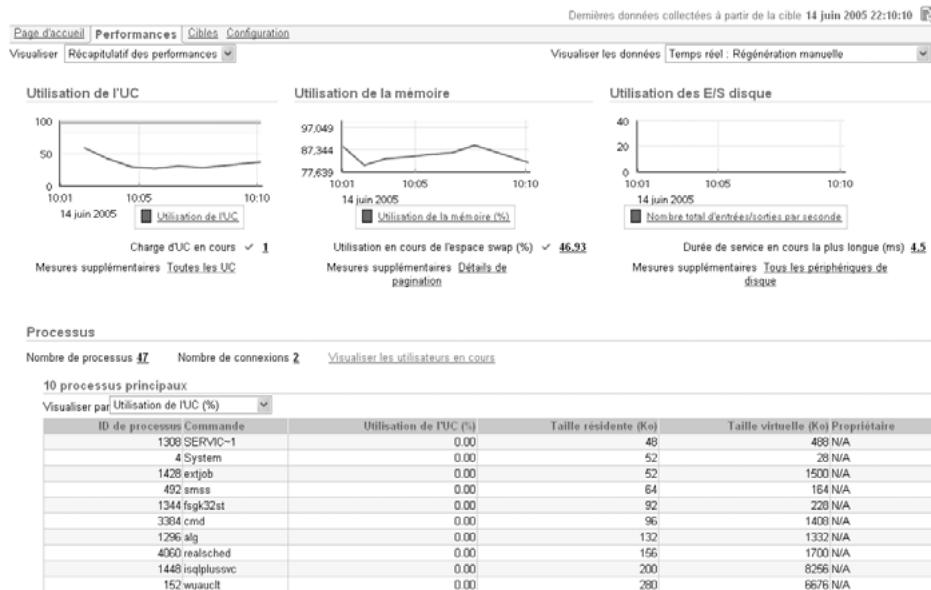


Figure 25-12

Vision globale des performances d'un serveur

En bas de chaque page, les options de navigation sont précieuses :



Figure 25-13

Les options de bas de page

Tout d'abord, sous le rappel des onglets, vous retrouvez dans chaque page les *Liens Associés*. Ce sont des raccourcis de navigation utiles. Ensuite, tout en bas de page, vous retrouverez d'autres options de navigation :

- **Base de données :** la bouée de sauvetage, retour à la page d'accueil ;
- **Installation :** permet de définir les administrateurs accédant à OEM, les méthodes de notification (adresses d'envoi d'e-mails), la méthode pour télécharger automatiquement les patches depuis le support technique Oracle ;

- **Préférences** : indique qui doit recevoir quel type de message e-mail. Par exemple, vous souhaitez être informé du titre et du détail d'une alerte, alors qu'un de vos collègues n'aura que le titre du message, sans son détail. Chacun reçoit les messages au format souhaité.
- **Aide** : l'aide en ligne est riche, très bien conçue et majoritairement traduite en français. Abusez-en !
- **Déconnexion** : vos données sont sensibles, évitez de laisser la console ouverte à des yeux indiscrets.

Je vous engage à paramétrer l'environnement à partir de ces écrans.

N'hésitez pas à naviguer à partir de la page d'accueil pour découvrir toute la richesse des informations proposées et vous habituer à naviguer efficacement dans OEM. Voyons maintenant le contenu de l'onglet *Performances*.

L'onglet Performances

Tout comme la page d'accueil, la page *Performances* présente des mesures globales, puis vous accédez au détail en cliquant sur les liens et les images.

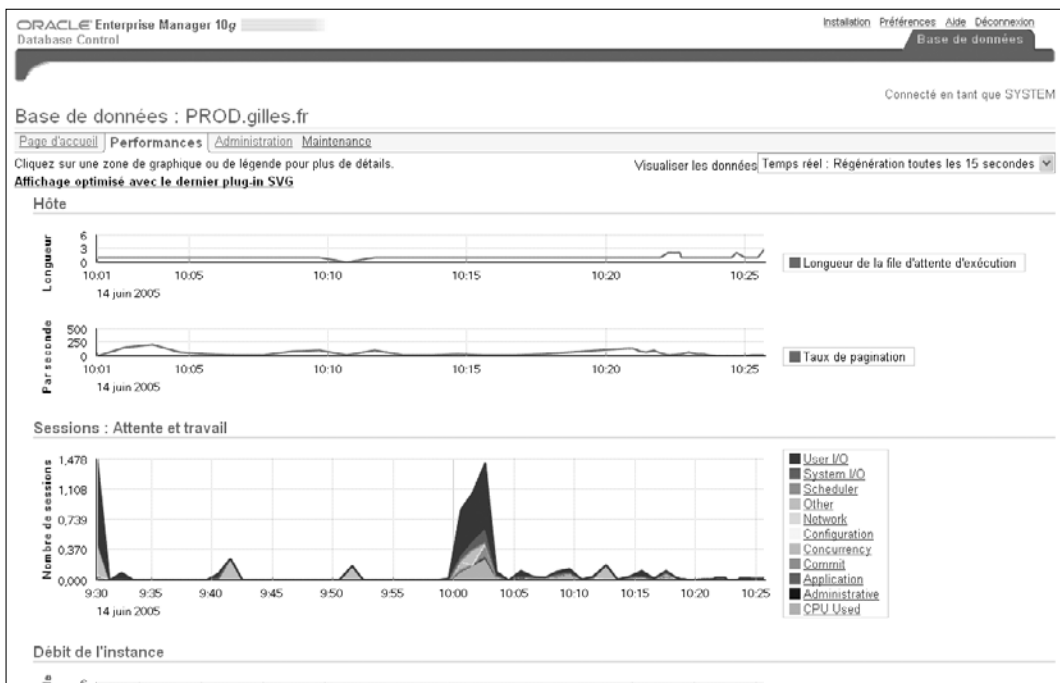


Figure 25-14

L'onglet Performances

Vous accédez ainsi à des mesures pointues. Toutes ces données étaient disponibles depuis des années mais seuls des outils complémentaires coûteux permettaient de les exploiter visuellement. Par exemple, avez-vous déjà visualisé les entrées-sorties logiques, c'est-à-dire mémoire, d'une base ? C'est maintenant simple comme un clic.



Figure 25-15

Les entrées-sorties logiques

En bas de page, avant le rappel de l'onglet *Performances*, vous disposerez dans tous les écrans de *Performances* des *Liens de surveillance supplémentaires* utiles pour naviguer efficacement au sein de l'onglet *Performances*.



Figure 25-16

Liens de surveillance supplémentaire

Visualisons maintenant le contenu de l'onglet *Administration*.

L'onglet Administration

Nous sommes maintenant au cœur de l'administration d'une base Oracle.

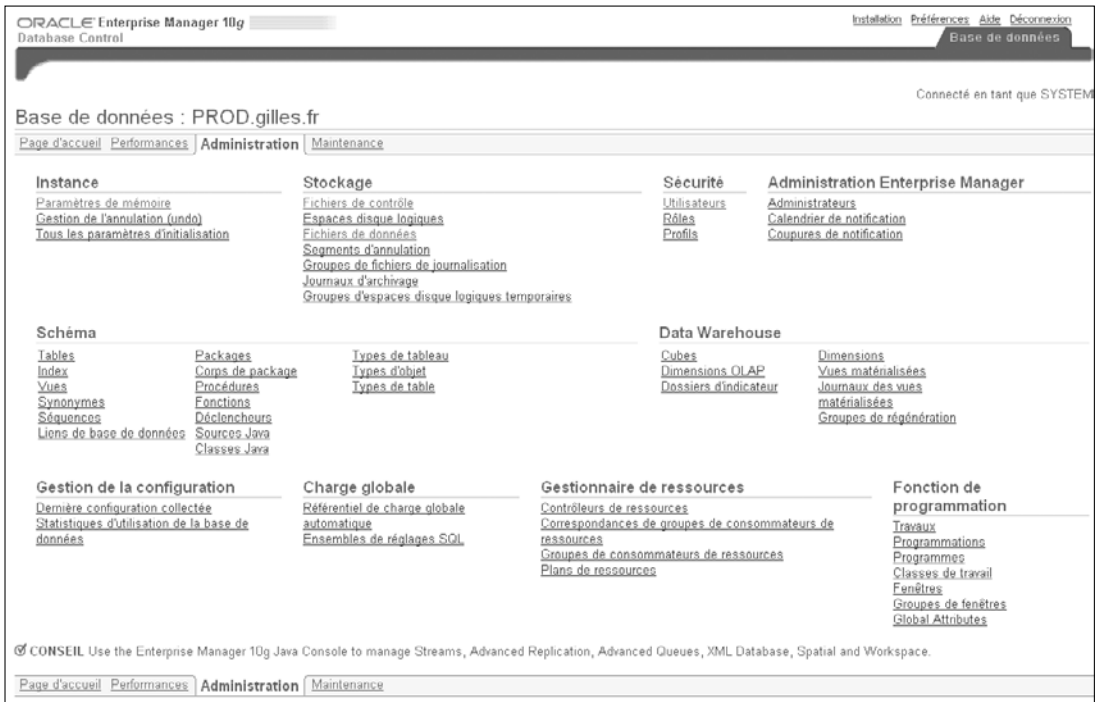


Figure 25-17
L'onglet Administration

- Toutes les interfaces graphiques indispensables à la gestion d'une base Oracle sont regroupées à ce niveau :
- Instance ;
 - Stockage ;
 - Sécurité ;
 - Administration OEM ;
 - Schéma ;
 - Data Warehouse ;
 - Gestion de la configuration ;
 - Charge globale ;
 - Gestionnaire des ressources ;
 - Fonctions de programmation.

Le détail et la richesse des informations accessibles nécessiteraient un ouvrage entier à eux seuls.

Par exemple, la figure suivante détaille l'espace d'une table propriété de SYS. Figurent sa taille totale, l'espace utilisé et donc l'espace « perdu ». Pour accompagner cet état, un graphe sur une période de temps de votre choix vous permet d'étudier *a posteriori* l'évolution de la taille de la table. Superbe !

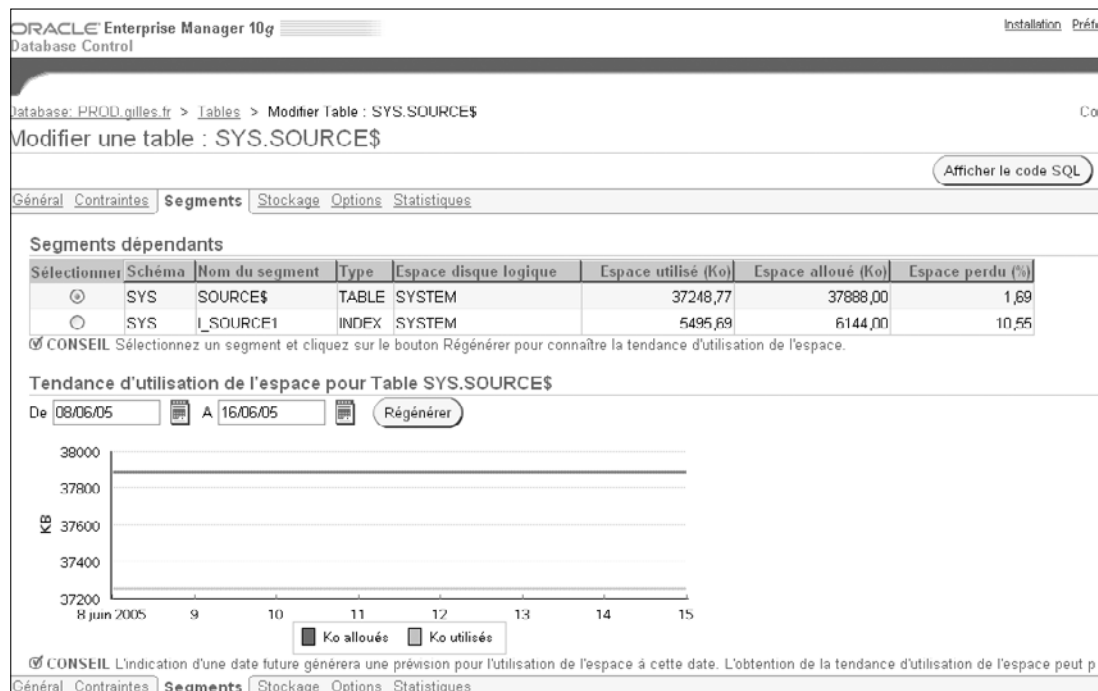


Figure 25-18

Détail des caractéristiques d'une table

Nous ne rentrerons pas plus dans le détail des options proposées, la découverte personnelle d'OEM sera votre meilleur guide.

L'onglet Maintenance

Cet écran regroupe trois familles :

- **Utilitaires** : un habillage graphique pour Export, Import, SQL*Loader, etc. ;
- **Sauvegarde/Récupération** : gérez vos sauvegardes depuis OEM ;
- **Déploiements** : appliquer des patches, cloner une base, etc.



Figure 25-19

L'onglet Maintenance

L'habillage d'utilitaires comme Export, Import ou SQL*Loader ne m'ont pas convaincu. Il vous sera plus rapide de l'effectuer manuellement, comme décrit au chapitre 18, *Les outils d'Oracle 10g*. Pour autant, les autres options sont très intéressantes et méritent que vous les étudiez.

Les faiblesses d'Oracle Enterprise Manager

Force est de constater que, par rapport aux précédentes versions, Oracle Enterprise Manager a très favorablement évolué. J'ai émis de nombreuses réserves à propos des précédentes versions :

- installation complexe : OEM Database Control est maintenant installé par l'assistant lors de la création de toute base. Il n'y a plus rien à faire !
- c'est boîte à outils parfois mal assemblée : l'ensemble a été revu de fond en comble, la présentation est agréable et prend beaucoup mieux en compte les besoins réels d'un administrateur Oracle ;
- des lacunes existent : il est toujours possible de mieux faire, mais la version 10g est une étape décisive ;
- les agents sont consommateurs : de gros efforts ont été faits. Les ressources consommées sont maintenant raisonnables. À vérifier toutefois lors d'une montée en charge conséquente ;
- la stabilité est perfectible : le « client lourd » Java était buggué. Le container OC4J J2EE n'est plus installé que sur le serveur (il n'y a plus rien à installer sur un poste client) et je n'ai pas pris sa stabilité en défaut ;

- les événements sont spécialisés Oracle : c'est l'Intelligent Agent présent sur le serveur distant qui a la charge d'exécuter les événements. Ceux qui concernent le matériel ou les systèmes d'exploitation sont obligatoirement « généralistes » : à partir d'un même événement, vous pouvez surveiller un serveur Windows NT ou Linux ! C'est là l'une des limites d'OEM : sa force réside dans la surveillance des bases Oracle et non dans celle du réseau, des systèmes d'exploitation ou du matériel. C'est là une question de fond qui subsiste : à quel outil confier l'ordonnancement de tâches ?

Faut-il utiliser Oracle Enterprise Manager ?

C'est maintenant sans aucune réserve que je vous conseille d'utiliser Oracle Enterprise Manager Database Control ou Database Grid Control dans toutes les tâches quotidiennes.

Résumé

Oracle Enterprise Manager est un produit très sophistiqué, permettant d'apporter des solutions à de très nombreuses questions d'administration.

C'est maintenant un outil mature que tous les administrateurs Oracle, les nouveaux comme les anciens, doivent considérer avec la plus grande attention.

26

La sauvegarde d'une base Oracle 10g

Dans ce chapitre :

- les principes de la sauvegarde d'une base Oracle 10g ;
- la sauvegarde base ouverte ou fermée ;
- un exemple de sauvegarde complète par script de commande Windows.

Une base de données contient souvent des informations vitales pour votre société. La mise en place de sauvegardes efficaces et testées est essentielle à votre sécurité et tranquillité d'esprit. C'est la tâche majeure dévolue à tout administrateur Oracle 10g.

Ce chapitre a pour objectif de vous aider à choisir le meilleur système de sauvegarde. Nous aborderons les différentes techniques de sauvegarde d'une base Oracle 10g et nous évoquerons Enterprise Manager avant de proposer une méthode de scripts pour automatiser les sauvegardes.

Seule la sauvegarde de la base Oracle 10g est abordée. Nous ne traitons ni de la sauvegarde de Windows, ni des mesures à prendre pour démarrer le système à partir de médias de sauvegardes, à la suite d'un dysfonctionnement majeur. Ces mesures doivent impérativement être mises en œuvre. Dans un souci de compatibilité, nous vous conseillons, pour sauvegarder les bases Oracle 10g, d'utiliser les mêmes moyens que ceux ayant permis de sauvegarder l'ensemble du système Windows.

Les sauvegardes sont vitales : les exemples exposés dans ce chapitre doivent être soigneusement testés pour valider leur bon fonctionnement sur les systèmes Windows de production.

Principes de sauvegarde d'une base Oracle 10g

Une sauvegarde consiste à conserver partiellement ou totalement une base de données afin de pouvoir la restaurer en cas d'incident. Par principe, il faut toujours s'attendre au pire.

Le choix du type de sauvegarde dépend bien évidemment de vos contraintes et tout d'abord du niveau de disponibilité requis par votre machine. Il est parfois plus facile de se poser la question inverse : de combien de temps disposez-vous pour remettre en œuvre les bases de données en cas d'incident ? Cela vous permettra de choisir les moyens adéquats.

La sauvegarde peut s'effectuer, base fermée ou ouverte, avec les utilisateurs « en ligne ». Nous traitons les deux cas dans ce chapitre.

Une sauvegarde n'est effective qu'une fois validée, le cas idéal étant de tester le redémarrage de la base de données sur une autre machine Windows. Cela permet de cerner l'ensemble des autres fichiers et actions à mettre en œuvre pour reconstruire l'environnement de production.

Les éléments à sauvegarder

Pour les bases Oracle 10g, il faut sauvegarder l'ensemble des fichiers qui la composent : fichiers de contrôle, redo-log, fichiers de données et fichiers temporaires.

Autres éléments à sauvegarder

D'autres fichiers doivent être sauvegardés :

- les fichiers d'initialisation comme le *initSID.ora*, local ou persistant ;
- les fichiers d'administrations placés en C:\oracle\product\10.1.0\admin ;
- les fichiers listener.ora et tnsnames.ora d'Oracle Net ;
- les fichiers mot de passe externes à la base placés en C:\oracle\product\10.1.0\db_1\database ;
- d'autres fichiers et applications sont également utiles à sauvegarder, de même que le système Windows.

Ils doivent être ajoutés au script proposé.

Les quatre types de sauvegarde d'une base Oracle 10g

Oracle 10g offre deux familles de sauvegarde, *logique* et *physique*. Ce sont les contraintes de disponibilité des bases Oracle 10g, ainsi que le type de sinistre contre lequel on cherche à se prémunir qui orientent vers l'une ou l'autre méthode.

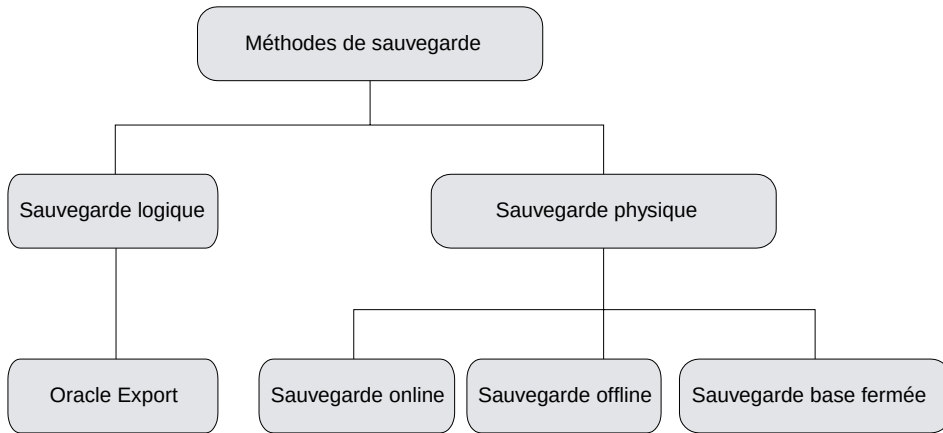


Figure 26-1

Sauvegardes logique et physique

La famille de sauvegarde qui possède le plus de possibilités est la sauvegarde physique. Les caractéristiques de chacune de ces méthodes sont résumées dans le tableau suivant :

Sauvegarde logique	Sauvegarde physique online	Sauvegarde physique offline	Sauvegarde base fermée
• Contient la définition de tous les objets (tables, index, synonymes...). • Exporte les données. • Les tablespaces doivent être ouverts. • Utilisée principalement pour transférer une base de machine à machine, changer de version Oracle, réorganiser des données fragmentées. • Outil Oracle Export/Import.	• Sauvegarde physique de tous les fichiers qui composent la base. • S'effectue tablespace par tablespace. • La base doit être en mode ARCHIVELOG. • La sauvegarde des fichiers a lieu sous Windows. • Les utilisateurs continuent à travailler normalement durant l'opération.	• Sauvegarde physique de tous les fichiers qui composent la base. • S'effectue tablespace par tablespace. • La base peut être en mode NOARCHIVELOG. • La sauvegarde des fichiers s'effectue sous Windows. • Les utilisateurs ne peuvent accéder au tablespace en cours de sauvegarde.	• Sauvegarde physique de tous les fichiers qui composent la base. • La sauvegarde des fichiers est effectuée sous Windows. • S'effectue base fermée. • Les utilisateurs ne peuvent accéder à la base.

La sauvegarde logique par Export/Import

Les utilitaires d'Export/Import des données sont très utilisés avec Oracle 10g. L'Export permet d'extraire tout ou partie de la base et la conserve dans un fichier. Ce fichier peut être transféré vers une base Oracle sur n'importe quel système d'exploitation cible, puis importé.

L'Export/Import est avant tout un outil de migration, de défragmentation et de transfert de données d'un environnement vers un autre.

Il permet par exemple d'importer une table qui aurait été détruite par mégarde, sans entreprendre la restauration d'un tablespace et donc de toutes les tables qu'il contient.

L'Export/Import est un moyen de sauvegarde supplémentaire à une sauvegarde physique complète de la base de données. Le chapitre 18, *Les outils d'Oracle 10g* détaillant ces aspects, nous n'y reviendrons pas dans ce chapitre.

Avantages et inconvénients des sauvegardes physiques

Ce type de sauvegarde permet le redémarrage uniquement sur un serveur de même type. Une base Oracle 10g pour Windows ne pourra être relancée que sur un serveur Windows où une version identique d'Oracle aura été installée. N'oubliez pas que le service Windows OracleServiceSID doit être préalablement créé à l'aide de l'utilitaire ORADIM. Ce point est détaillé au chapitre 5, *Fonctionnement d'Oracle 10g sous Windows*.

Une sauvegarde logique comme l'Export permet de contourner cet obstacle.

Trois méthodes sont abordées :

- la plus simple permet la sauvegarde d'une base Oracle 10g fermée ;
- la seconde, concerne la sauvegarde d'une base Oracle 10g tablespace par tablespace, chaque tablespace étant fermé avant sa sauvegarde ;
- la troisième, effectue la sauvegarde base ouverte, les utilisateurs continuant de travailler normalement.

Chacune de ces méthodes peut utiliser des scripts destinés à être lancés en tâche de fond par la commande *Tâche planifiée* de Windows.

La première méthode est étudiée dans le détail, car c'est celle que vous devez mettre en œuvre prioritairement. Une fois cette sauvegarde effectuée, elle doit vous permettre de redémarrer sur une autre machine ayant un système d'exploitation identique. Pour redémarrer sur un système d'exploitation différent, seul l'Export logique convient. Dans ce cas, une base de données cible ayant les mêmes caractéristiques que celle d'origine devra être créée au préalable.

Les deuxième et troisième méthodes sont d'une mise en œuvre plus complexe, surtout dans les phases de restauration. Nous décrivons succinctement leur principe de fonctionnement.

Sauvegarde et restauration

Ces deux actions sont les plus importantes pour un administrateur Oracle 10g. Soyez extrêmement vigilant. N'hésitez pas à faire valider vos sauvegardes par une autre personne. Enfin, entraînez-vous pour être en mesure de réagir positivement, sans succomber à la panique et au stress imposé par des utilisateurs encombrants lorsqu'un dysfonctionnement survient.

Nous vous conseillons fortement de conserver par écrit, dans un cahier de suivi ou dans un fichier, toutes les actions effectuées sur votre machine lors de sa configuration et de la mise en œuvre des différentes bases de données. C'est très utile en cas de reconstruction d'un environnement identique. Les scripts SQL fournis à l'annexe 3, *Procédures pour le DBA*, vous seront d'une aide précieuse en cas de problème.

Enfin, avant toute opération de correction effectuée dans le feu de l'action, il est indispensable de réaliser une (ou plusieurs !) sauvegarde(s). Cela vous permet de prendre du recul, d'examiner la situation, de pouvoir revenir plusieurs fois au niveau de l'incident et de ne pas le transformer en catastrophe.

Existe-t-il d'autres options de sauvegarde ?

Oracle 10g a introduit de nouvelles options de sauvegarde extrêmement poussées et performantes. Notre but est de vous fournir les éléments fondamentaux de toute sauvegarde.

Un volume entier de la documentation Oracle est consacré aux possibilités de sauvegardes et restaurations d'Oracle 10g.

Les sauvegardes avec Oracle Enterprise Manager

Parmi les utilitaires standard d'Oracle Enterprise Manager, RMAN (*Recovery Manager*) est conçu pour automatiser les sauvegardes. Son principe repose sur des jobs lancés à des périodes déterminées.

Toutes les caractéristiques des sauvegardes sont indiquées et transmises à l'agent d'Oracle Enterprise Manager situé sur la machine Windows hébergeant la base. Celui-ci est responsable de l'exécution de la sauvegarde et de son compte rendu.

Cette méthode nécessite toute une infrastructure préalable dont une base tierce contenant le référentiel d'Oracle Enterprise Manager. Cette possibilité de sauvegarde n'est pas abordée dans ce chapitre qui se limite à effectuer des sauvegardes complètes les plus simples possibles, sans recours à un outil externe.

Même confiées à Oracle Enterprise Manager, n'oubliez pas de tester vos sauvegardes avant de les mettre en production.

Sauvegarde base fermée ou base ouverte

Sauvegarde base fermée

Lorsque la base est fermée, l'opération de sauvegarde est simple : elle consiste à copier tous les fichiers de la base sur un support fiable.

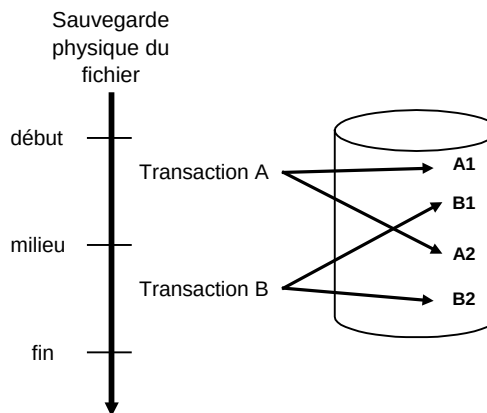
Cela se complique avec les options d'administration des tablespaces. L'ensemble des fichiers des différents tablespaces regroupe la totalité des fichiers base de données. Pour plus de souplesse, Oracle 10g permet à un tablespace plein d'augmenter sa taille ou de s'allouer automatiquement un nouveau fichier sur disque. Il est impératif que vos sauvegardes construisent dynamiquement la liste des fichiers à sauvegarder, en interrogeant le dictionnaire de la base de données. L'oubli d'un fichier (données, temporaire, contrôle, redo-log) rend impossible la reconstitution de la base en cas de restauration.

Pourquoi ne pas copier les fichiers base ouverte ?

Il convient de ne jamais sauvegarder les fichiers d'une base Oracle 10g sans mettre en place des méthodes spéciales. La figure suivante en schématise la raison :

Figure 26-2

*Sauvegarde des fichiers
base ouverte*



Dans la figure précédente, l'on effectue une copie d'un des fichiers de la base par la commande Windows copy, alors que celle-ci est ouverte et que les utilisateurs travaillent. Durant la copie, la transaction A est réalisée au début de la copie du fichier et la transaction B vers la fin de la copie. Si toutes les modifications de la transaction A figurent dans le fichier sauvegardé, seule une partie des écritures de la transaction B sont prises en compte dans la copie du fichier. Des informations sont perdues et la sauvegarde incohérente.

Les mécanismes mis en place dans Oracle 10g utilisent deux techniques : l'interdiction temporaire d'écrire dans le fichier à sauvegarder ou l'utilisation conjointe des segments d'annulation et des redo-log pour retarder l'écriture des modifications. Dans les deux cas, un « top départ » sera à envoyer à la base Oracle 10g pour lui signifier le début de la

sauvegarde et un « stop » pour lui indiquer la fin. Ensuite, Oracle 10g utilise les informations conservées dans les segments d'annulation et les redo-log pour écrire les données dans les fichiers.

Sauvegarde base ouverte

Lorsque la base est ouverte, c'est-à-dire en production et accessible au public, l'administrateur a la possibilité de réaliser une sauvegarde des fichiers d'un tablespace en service (*on line*) ou hors service (*off line*).

Dans le cas de la sauvegarde d'un tablespace en service, l'administrateur va devoir copier les fichiers de la base de données alors que les utilisateurs continuent d'y accéder. Ceci n'est possible que si le processus d'archivage des fichiers redo-log est actif et que la base est en mode ARCHIVELOG.

Lors de la sauvegarde base ouverte, les fichiers de la base sont accédés par le processus `oracle.exe` et la commande `copy` de Windows refuse alors de fonctionner pour sauvegarder les fichiers. Pour contourner ce problème, Oracle livre le logiciel `ocopy.exe` :

```
Usage:
  ocopy from_file [to_file [a | size_1 [size_n]]]
  ocopy -b from_file to_drive
  ocopy -r from_drive to_dir
```

Pour démarrer le processus d'archivage des redo-log, placez les lignes suivantes dans le fichier d'initialisation *initSID.ora* des bases à sauvegarder :

```
LOG_ARCHIVE_START=TRUE
LOG_ARCHIVE_DEST= répertoire_cible_pour_redo_log_archivés
```

Pour mettre votre base de données en mode ARCHIVELOG (c'est-à-dire pour qu'elle copie les redo-log à l'endroit précisé par `LOG_ARCHIVE_DEST`), vous devez utiliser SQL*Plus :

```
# assurez vous de travailler sur la bonne base !
# Par exemple, la base TEST
set ORACLE_SID=TEST

# lancez SQL*Plus
sqlplus /nolog
SQL> connect / as sysdba
SQL> shutdown immediate ;

# Pour passer en mode ARCHIVELOG il faut être en STARTUP MOUNT
SQL> startup mount ;

SQL> alter database archivelog ;

SQL> shutdown immediate ;

# on redémarre en mode normal
SQL> startup
```

Dans ce cas, pour sauvegarder le tablespace X en ligne, il faut procéder ainsi :

```
SQL> ALTER TABLESPACE x BEGIN BACKUP
Sauvegarde niveau système d'exploitation de tous les fichiers du tablespace
➔ à l'aide de l'outil Oracle OCOPY.EXE
SQL> ALTER TABLESPACE x END BACKUP
```

La sauvegarde de la base entière s'effectue tablespace par tablespace, y compris pour le tablespace SYSTEM.

Dans le cas d'une sauvegarde d'un tablespace hors service, il suffit de le mettre offline de sorte que personne ne puisse l'utiliser, et de copier les fichiers qui le composent.

Pour sauvegarder le tablespace X hors ligne, il faut procéder comme suit :

```
SQL> ALTER TABLESPACE x OFFLINE
Sauvegarde niveau système d'exploitation de tous les fichiers du tablespace
➔ à l'aide de l'outil Oracle OCOPY.EXE
SQL> ALTER TABLESPACE x ONLINE
```

Notez que seul le tablespace SYSTEM ne peut être mis *offline*.

Des commandes SQL fournies en annexe permettent de contrôler les options utilisées par la base de données.

Le fichier d'alerte

Toute opération importante qui influe sur la vie de votre base (démarrage, arrêt, création d'un tablespace, mise *offline* ou *online*) figure dans son fichier d'alerte. D'après les règles OFA, il se situe dans c:\oracle\admin\SID\bdump>alert_SID.log.

Lorsqu'un tablespace est sauvegardé offline, il contient les indications suivantes :

```
...
Je Aou 23 21:49:52 2005
alter database archivelog
Completed: alter database archivelog
Je Aou 23 21:49:52 2005
...
Je Aou 23 21:49:52 2005
alter tablespace users begin backup
Je Aou 23 21:49:52 2005
Completed: alter tablespace users begin backup
Je Aou 23 21:49:52 2005
Load Indicator not supported by OS !
Je Aou 23 21:49:52 2005
....
Copie des fichiers du tablespace Users par des commandes Windows !
....
alter tablespace users end backup
Je Aou 23 21:49:52 2005
Completed: alter tablespace users end backup
Je Aou 23 21:49:52 2005
...
```

Toutes les opérations importantes qui se sont déroulées sur la base, figurent dans ce fichier.

Les sauvegardes par script de commandes

C'est notre mode de sauvegarde préféré, car sa souplesse et son contrôle sont total. Il utilise différentes techniques que nous vous présentons maintenant.

Création dynamique des fichiers à sauvegarder

Les vues V\$DATAFILE, V\$TEMPFILE, V\$LOGFILE, V\$CONTROLFILE permettent de récupérer dynamiquement les fichiers qui composent la base de données. Il suffit d'insérer du texte avant et après pour créer des fichiers de sauvegarde de type :

```
copie fichier1 vers destination  
copie fichier2 vers destination  
etc.
```

Principes de sauvegarde

Lancement des sauvegardes en deux temps : le fichier sauvegarde.cmd contient toutes les commandes pour effectuer la sauvegarde. Il crée deux fichiers intermédiaires, détruits à la fin du script principal : un fichier contenant des ordres SQL pour récupérer dynamiquement, dans la base, la liste des fichiers qui la composent et un fichier de commandes qui duplique les fichiers de la base en ajoutant un .old à leur nom. Cela permet de limiter l'attente due à la copie des fichiers de données sur bande, les fichiers .old étant sauvegardés une fois la base de données relancée. Le fichier sauvegarde.log conserve toutes les opérations effectuées ainsi que les erreurs rencontrées.

Où trouver un exemple plus abouti ?

L'exemple proposé est rudimentaire et le parti pris lors de son écriture a été d'être clair. Vous pouvez réaliser des scripts plus sophistiqués, comme celui proposé sur le site <http://www.backupcentral.com/oraback.html>.

Exécution par la commande de planification *Tâches planifiées*

La commande *Tâches planifiées* de Windows autorise la planification (*scheduling*) de commandes. Une fois les scripts mis au point, validés manuellement, ils peuvent être lancés à votre convenance, sous le compte utilisateur choisi, à fréquence déterminée.

L'assistant Windows est accessible depuis Démarrer > Paramètres > Panneau de Configuration > Tâches planifiées.

Il est particulièrement important que les commandes lancées par l'outil de planification s'exécutent sous le même compte Windows que celui utilisé lors de vos tests de mise au point des scripts.

Remontées d'informations

Une fois la sauvegarde effectuée, vous pouvez étudier vos fichiers résultat. S'il y a des erreurs ou si des mots-clés y figurent, des alertes seront remontées. Ces alertes sont définies

comme vous le souhaitez : dans un fichier particulier, par l'envoi d'un message dans une messagerie, etc.

Lancement des sauvegardes

La sauvegarde présentée se compose de deux scripts : lance.cmd et sauvegarde.cmd. Le premier sert uniquement à lancer le second et à recueillir et analyser tous les messages provenant de sauvegarde.cmd. Si des erreurs sont rencontrées, un message est envoyé dans l'observateur d'événements.

Les scripts ont été paramétrés pour permettre une maintenance facile. Seules les variables en début de chaque script doivent être adaptées.

Ce script doit être lancé sous l'utilisateur Windows *administrateur* (ou tout autre utilisateur appartenant groupe ORA_DBA) pour permettre une connexion CONNECT / AS SYSDBA sans exposer de mot de passe dans le script. Ce point est présenté au chapitre 24, *Stratégie de sécurité sous Windows*.

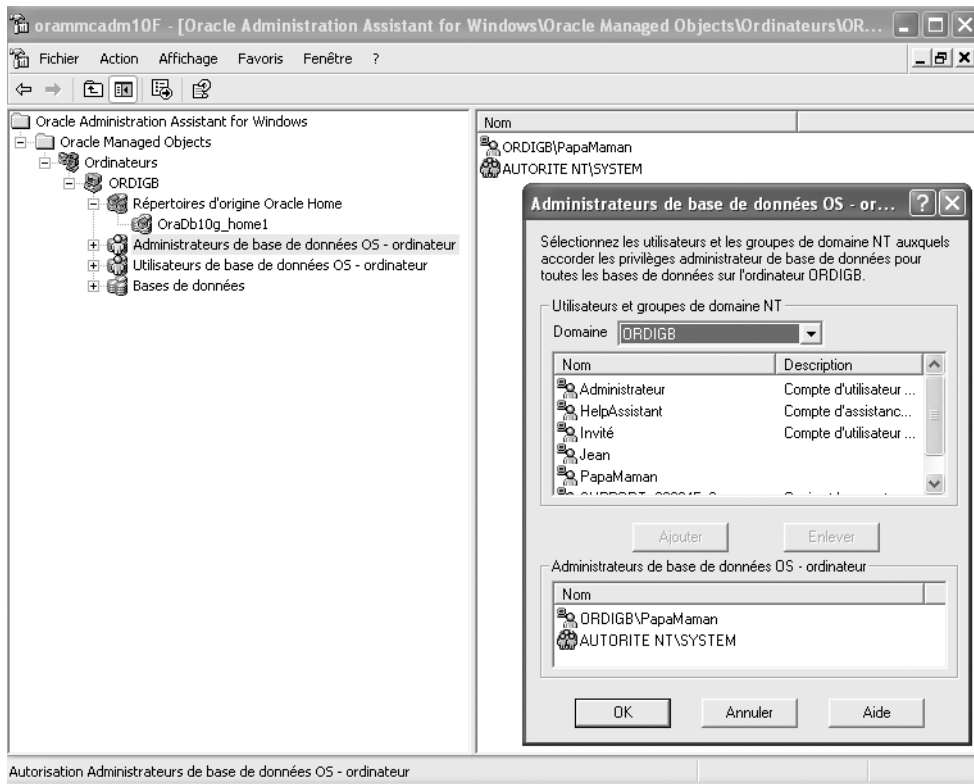


Figure 26-3

Le groupe permettant de démarrer la base

Nous vous conseillons de mettre au point et de valider le script sauvegarde.cmd avant de le lancer avec le script lance.cmd. La dernière étape consistant à confier l'exécution du script lance.cmd à l'assistant *Tâches Planifiées* de Windows.

Le script lance.cmd

```
rem
rem Fichier LANCE.CMD : sert a lancer les scripts de sauvegarde.
rem
rem Il recupere aussi les erreurs rencontrees et vous avertit
rem par le moyen que vous aurez choisi.

rem Adaptez cette valeur a votre environnement
set L_RAPPORT=c:\gilles\backup
set L_SOURCE=c:\gilles\script

rem *****
rem Lance le script de sauvegarde et conserve tous les messages
rem *****

call %L_SOURCE%\sauvegarde.cmd >> %L_RAPPORT%\sauvegarde.log

rem *****
rem Recherche d'erreurs (remontees ici dans l'Observateur d'Evenement)
rem *****
qgrep "error" %L_RAPPORT%\sauvegarde.log >>%L_RAPPORT%\erreur.log
qgrep "ora-" %L_RAPPORT%\sauvegarde.log >>%L_RAPPORT%\erreur.log
qgrep "cannot" %L_RAPPORT%\sauvegarde.log >>%L_RAPPORT%\erreur.log
qgrep "not logged" %L_RAPPORT%\sauvegarde.log >>%L_RAPPORT%\erreur.log
qgrep "failure" %L_RAPPORT%\sauvegarde.log >>%L_RAPPORT%\erreur.log
if exist %L_RAPPORT%\erreur.log c:\ntreskit\logevent -s E "!!! ERREUR DANS VOTRE
➡SAUVEGARDE ORACLE !!! Consultez les fichiers situes en: %L_RAPPORT%"

rem *****
rem Nettoyage des fichiers de commande crees dans SAUVEGARDE.CMD
rem *****

if exist %L_SOURCE%\db_start.sql del %L_SOURCE%\db_start.sql
if exist %L_SOURCE%\db_stop.sql del %L_SOURCE%\db_stop.sql
if exist %L_SOURCE%\db_resteint.sql del %L_SOURCE%\del db_resteint.sql

rem *****
rem Fin du fichier de lancement
rem *****
```

Le script sauvegarde.cmd

```
@echo off
rem Fichier SAUVEGARDE.CMD
ECHO =====
ECHO Script de sauvegarde d'une base Oracle sous Windows.
ECHO =====
```

```

ECHO
setlocal

rem =====
rem Modifiez ces valeurs pour les adapter a votre environnement
rem =====

set ORACLE_SID=PROD
set ORA_HOME=OraDb10g_home1

REM Indiquez le lieu ou sauvegarder les fichiers de la base
set L_SAUVE= c:\gilles\backup

rem Identifiez les executables Oracle utilises
set L_PLUS=C:\oracle\product\10.1.0\db_1\BIN\sqlplus.exe
set L_EXPORT=C:\oracle\product\10.1.0\db_1\BIN\exp.exe
set L_COPY=C:\oracle\product\10.1.0\db_1\BIN\ocopy.exe

ECHO =====
ECHO Base sauvegardee : %ORACLE_SID%
ECHO Serveur          : %COMPUTERNAME%
ECHO Heure debut      :
Date /t
Time /t
ECHO =====
ECHO

rem =====
ECHO -- Lance des programmes utilises dans les sauvegardes
rem =====

rem Si le Listener n'est pas demarre, on le lance
net start | find "Oracle%ORA_HOME%TNSListener" || net start
➔Oracle%ORA_HOME%TNSListener

rem si OracleDBConsole n'est pas lance, on le lance
net start | find "OracleDBConsole%ORACLE_SID%" || net start
➔OracleDBConsole%ORACLE_SID%

rem Si le Service Oracle n'est pas demarre, on le lance
net start | find "OracleService%ORACLE_SID%" || net start OracleService%ORACLE_SID%

rem =====
ECHO -- Debut de la sauvegarde
rem =====
REM Creation du fichier de demarrage de la base : DB_START.SQL
REM On considere que le fichier d'initialisation persistant
REM est utilise, ce qui evite "startup pfile=c:\..."
echo connect / as sysdba      >db_start.sql
echo startup                  >>db_start.sql
echo exit                      >>db_start.sql

REM Creation du fichier d'arret de la base : DB_STOP.SQL

```

```
echo connect / as sysdba      >db_stop.sql
echo shutdown immediate      >>db_stop.sql
echo exit                    >>db_stop.sql

REM Creation du fichier d'arret et de demarrage de la base
REM en mode restreint : DB_RESTEINT.SQL
REM Aucun autre utilisateur supplementaire ne peut se connecter.
echo connect / as sysdba      > db_resteint.sql
echo shutdown immediate      >> db_resteint.sql
echo startup restrict        >> db_resteint.sql
echo exit                    >> db_resteint.sql

ECHO =====
ECHO SAUVEGARDE BASE ARRETEE (SAUVEGARDE A FROID)
ECHO =====

ECHO -- Arret de la base et demarrage en mode restreint
ECHO %L_PLUS% "/nolog" @db_resteint.sql
%L_PLUS% "/nolog" @db_resteint.sql

ECHO -- Creation du script SQL*PLUS pour la sauvegarde a froid
echo connect / as SYSDBA      >%L_SAUVE%\sauve.sql
echo set heading off;        >>%L_SAUVE%\sauve.sql
echo set feedback off;       >>%L_SAUVE%\sauve.sql
echo set linesize 150;       >>%L_SAUVE%\sauve.sql
echo set pagesize 100;       >>%L_SAUVE%\sauve.sql

echo spool %L_SAUVE%\sauve.cmd; >>%L_SAUVE%\sauve.sql

echo select 'copy '^|^|member^|^'|' %L_SAUVE%' from v$logfile; >>%L_SAUVE%\sauve.sql
echo select 'copy '^|^|name^|^'|' %L_SAUVE%' from v$controlfile; >>%L_SAUVE%\sauve.sql
echo select 'copy '^|^|name^|^'|' %L_SAUVE%' from v$datafile; >>%L_SAUVE%\sauve.sql
echo select 'copy '^|^|name^|^'|' %L_SAUVE%' from v$tempfile; >>%L_SAUVE%\sauve.sql
echo spool off; >>%L_SAUVE%\sauve.sql
echo exit; >>%L_SAUVE%\sauve.sql

ECHO -- Lancement du script SQL pour créer le script de sauvegarde
%L_PLUS% "/nolog" @%L_SAUVE%\sauve.sql

ECHO -- Arret de la base de donnees
%L_PLUS% "/nolog" @db_stop.sql

ECHO -- Ajoutez ici les autres fichiers a sauvegarder
echo rem Autres fichiers a sauvegarder... >>%L_SAUVE%\sauve.cmd

ECHO -- Lance le script de sauvegarde
call %L_SAUVE%\sauve.cmd

ECHO -- Optionnel : arrete le service Oracle
rem net stop OracleService%ORACLE_SID%

ECHO -- Demarre le service avec la base de donnees
rem net start OracleService%ORACLE_SID%
```

```

ECHO -- Sauvegarde a froid terminee
endlocal
ECHO =====
ECHO Fin du script
ECHO =====

```

Compte rendu d'exécution

Le fichier *sauvegarde.log* comporte un compte rendu exhaustif de toutes les opérations réalisées durant la sauvegarde. Il serait judicieux de le sauvegarder avec vos données et d'incorporer la date du jour dans son nom de façon qu'il ne soit jamais écrasé par le compte rendu de la sauvegarde suivante.

Dans cet exemple, nous avons grisé les fichiers à sauvegarder issus de l'interrogation SQL de la base. Même si un fichier est ajouté à la base, il sera automatiquement ajouté à la prochaine sauvegarde. C'est un point fort pour les scripts dynamiques !

```

=====
Script de sauvegarde d'une base Oracle sous Windows.
=====
=====
Base sauvegardée : PROD
Serveur          : ORDIGB
Heure début      :
dim. 15/06/2005
11:53
=====
Commande ECHO désactivée.
-- Lance des programmes utilises dans les sauvegardes
  Oracle OraDb10g_home1TNSListener
  OracleServicePROD
-- Debut de la sauvegarde
=====
SAUVEGARDE BASE ARRETEE (SAUVEGARDE A FROID)
=====
-- Arret de la base et demarrage en mode restreint

SQL*Plus: Release 10.1.0.1.0 - Production on Di Jun 15 10:50:08 2005

Copyright (c) 1982, 2005, Oracle Corporation. All rights reserved.

Connecté.
Base de données fermée.
Base de données démontée.
Instance ORACLE arrêtée.
Instance ORACLE lancée.

Total System Global Area 135338868 bytes
Fixed Size      453492 bytes
Variable Size 109051904 bytes
Database Buffers 25165824 bytes

```



```
Redo Buffers      667648 bytes
Base de données montée.
Base de données ouverte.
Déconnecté de Oracle 10g Enterprise Edition Release 10.1.0.1.0 - Production
With the Partitioning, OLAP and Oracle Data Mining options
-- Creation du script SQL*PLUS pour la sauvegarde a froid
-- Lancement du script SQL pour créer le script de sauvegarde
```

SQL*Plus: Release 10.1.0.1.0 - Production on Di Jun 15 10:50:36 2005

Copyright (c) 1982, 2005, Oracle Corporation. All rights reserved.

Connecté.

```
copy C:\ORACLE\ORADATA\PROD\RED001.LOG c:\gilles\backup
copy C:\ORACLE\ORADATA\PROD\RED002.LOG c:\gilles\backup
copy C:\ORACLE\ORADATA\PROD\RED003.LOG c:\gilles\backup
```

```
copy C:\ORACLE\ORADATA\PROD\CONTROL01.CTL c:\gilles\backup
copy C:\ORACLE\ORADATA\PROD\CONTROL02.CTL c:\gilles\backup
copy C:\ORACLE\ORADATA\PROD\CONTROL03.CTL c:\gilles\backup
```

```
copy C:\ORACLE\ORADATA\PROD\SYSTEM01.DBF c:\gilles\backup
copy C:\ORACLE\ORADATA\PROD\UNDOTBS01.DBF c:\gilles\backup
copy C:\ORACLE\ORADATA\PROD\CWMLITE01.DBF c:\gilles\backup
copy C:\ORACLE\ORADATA\PROD\DRSYS01.DBF c:\gilles\backup
copy C:\ORACLE\ORADATA\PROD\EXAMPLE01.DBF c:\gilles\backup
copy C:\ORACLE\ORADATA\PROD\INDEX01.DBF c:\gilles\backup
copy C:\ORACLE\ORADATA\PROD\ODM01.DBF c:\gilles\backup
copy C:\ORACLE\ORADATA\PROD\TOOLS01.DBF c:\gilles\backup
copy C:\ORACLE\ORADATA\PROD\USERS01.DBF c:\gilles\backup
copy C:\ORACLE\ORADATA\PROD\XDB01.DBF c:\gilles\backup
copy C:\ORACLE\ORADATA\PROD\OEM_REPOSITORY.DBF c:\gilles\backup
```

```
copy C:\ORACLE\ORADATA\PROD\TEMP01.DBF c:\gilles\backup
```

```
Déconnecté de Oracle 10g Enterprise Edition Release 10.1.0.1.0 - Production
With the Partitioning, OLAP and Oracle Data Mining options
JServer Release 10.1.0.1.0 - Production
-- Arrêt de la base de données
```

SQL*Plus: Release 10.1.0.1.0 - Production on Di Jun 15 10:50:39 2005

Copyright (c) 1982, 2005, Oracle Corporation. All rights reserved.

Connecté.

```
Base de données fermée.
Base de données démontée.
Instance ORACLE arrêtée.
Déconnecté de Oracle 10g Enterprise Edition Release 10.1.0.1.0 - Production
```

```
-- Ajoutez ici les autres fichiers a sauvegarder
-- Lance le script de sauvegarde
```

```
-- Optionnel : arrete le process Oracle
Le service OracleServicePROD s'arrête...
Le service OracleServicePROD a été arrêté.Le service OracleServicePROD
➡démontre.....
Le service OracleServicePROD a démarré.-- Sauvegarde a froid terminee
=====
Fin du script
Heure de fin :
dim. 15/09/2005
12:02
=====
```

Comme vous le constatez, la syntaxe de ce fichier de sauvegarde est simple et peut aisément être complétée ou modifiée pour s'intégrer à d'autres scripts d'administration ou de sauvegarde. Une première étape peut consister à l'intégrer à vos sauvegardes Windows actuelles.

Résumé

Ce chapitre a abordé, en les commentant, différents types de sauvegardes : au moyen d'utilitaires standards, d'Oracle Enterprise Manager et enfin par des scripts de commandes.

De nombreux conseils sont fournis pour vous aider à choisir parmi ces méthodes et pour les mettre en œuvre.

Configurer les disques pour Oracle

Dans ce chapitre :

- les différents systèmes de fichiers rencontrés sous Windows ;
- les différentes technologies RAID ;
- les interactions entre Oracle 10g et les disques RAID ;
- sur quel type de RAID placer les fichiers Oracle ;
- présentation d'Oracle ASM.

Ce chapitre intéresse à la fois l'administrateur système et celui d'Oracle 10g. Sur les systèmes Windows, il s'agit souvent d'une seule et même personne.

Nous commentons d'abord un aspect important du système : le stockage des données. Windows propose différents systèmes de fichiers dont *FAT16*, *FAT32* et *NTFS* avec lesquels vous pouvez mettre en œuvre des techniques RAID.

Ce chapitre se focalise sur les contraintes spécifiques liées à la mise en œuvre d'une base de données. À ce titre, nous présentons les fichiers de type Raw Devices et surtout la nouveauté d'Oracle 10g, Automatic Storage Management (ASM).

Enfin, nous précisons les conséquences des choix sur les performances dans le contexte d'une base de données.

Les différents systèmes de fichiers

Windows autorise plusieurs systèmes de fichiers. Certains, comme *CDFS* et *UDF*, concernent des médias amovibles CD-Rom et DVD-Rom. Nous ne parlerons ici que des formats *FAT16*, *FAT32* et *NTFS* qui concernent les disques durs.

Le choix du format s'opère au niveau du formatage d'une partition disque et se trouve donc sous la responsabilité de l'administrateur de système Windows. Ce formatage s'effectue aussi bien sur des disques classiques que sur des disques associés en technologie RAID.

Le système de fichiers est d'une grande importance pour une base de données dans la mesure où celle-ci le sollicite fortement.

L'outil privilégié utilisé par l'administrateur Windows est le Gestionnaire de disques.

Les différents systèmes de fichiers Windows

Windows propose quelques systèmes de fichiers. Ils n'ont pas la même utilisation et certains sont plus populaires que d'autres :

- *FAT16* : c'est le premier système de fichiers utilisé par Microsoft sous Windows 3.11. Il présente maintenant de nombreuses limitations et doit être réservé au formatage des disquettes !
- *FAT32* : introduit sous Windows 95 R2 puis Windows 98, *FAT32* permet d'accéder à de plus grands volumes que ceux autorisés par *FAT16* et le stockage des fichiers est plus efficace ;
- *NTFS* : comme le préconise Microsoft : « Si vous recherchez la meilleure sécurité, les performances et l'efficacité, choisissez NTFS ». C'est le système de fichiers à utiliser sous Windows 2000, 2003, XP.

Quel système de fichiers pour Oracle 10g ?

Pour installer et utiliser Oracle 10g, créez des partitions au format *NTFS*. C'est le plus généraliste et ses performances sont bonnes.

N'utilisez pas certaines options de NTFS qui permettent de compresser la partition ou encore d'en crypter les données. Les performances en entrée/sortie seraient alors fortement limitées.

Vous pouvez utiliser d'autres systèmes de fichiers, comme *FAT16* ou *FAT32*, mais ils ne présentent aucun intérêt dans l'utilisation d'une base de données.

Vous avez aussi la possibilité de court-circuiter tout système de fichiers Windows en utilisant des fichiers Raw Devices ou encore des disques ASM.

Les technologies RAID

Le « mirroring », le « duplexing », le « shadowing » ou encore le « striping » sont autant de techniques destinées à améliorer les performances entrées/sorties des disques et à renforcer la sécurité des unités de stockage de masse.

Depuis quelque temps, les performances des processeurs ont, en moyenne, doublé chaque année. Parallèlement, la capacité des périphériques de stockage a, elle aussi, doublé, mais au rythme d'une fois tous les trois ans seulement. Il y a un décalage évident d'évolution des performances entre les disques durs et les processeurs.

Pour pallier ces décalages, nous sommes amenés à mettre en œuvre des techniques capables, malgré la « lenteur » des disques, d'exploiter au mieux la cadence élevée des processeurs tout en satisfaisant les besoins croissants en capacité de stockage des systèmes informatiques actuels.

Un système disque de type RAID (*Redundant Array of Inexpensive Disks*) a pour objectif de répondre aux principaux critères de performance et de protection des données.

Une panne sur l'un des disques du système ne doit ni altérer l'exploitation ni entraîner de perte de données. En cas de défaillance d'un disque RAID, les techniques utilisées sont celles du mirroring (recopie complète d'un ou de plusieurs disques à partir d'un contrôleur), du duplexing (principe très proche, mais via deux contrôleurs distincts et synchronisés) et du striping (combinaison de plusieurs unités physiques en une seule unité logique et distribution des blocs de données sur l'ensemble de ces unités logiques).

Il existe plusieurs niveaux RAID (RAID 0, RAID 1, RAID 2, ..., RAID 5), ce qui peut prêter à confusion. Le niveau RAID 3, par exemple, n'a aucun point commun avec le niveau RAID 2. Il est plus approprié de parler de types RAID.

Le panachage entre les accès écriture et lecture ainsi que la longueur d'un transfert sont des points importants. Les performances varient également avec la longueur (taille) des demandes aux disques et, par conséquent, en fonction de l'application. Une application transactionnelle travaillera plutôt avec de petites requêtes courtes et fréquentes, alors qu'une application décisionnelle (infocentre, datawarehouse), effectuera des requêtes plus longues et des « balayages » séquentiels de disque.

On ne peut pas affirmer que les systèmes RAID apportent une solution sur mesure à tous les problèmes de flux de données. Une solution RAID n'est pas forcément plus rapide que les mêmes disques utilisés de manière conventionnelle.

RAID niveau 0

Le RAID 0 définit une technique d'entrelacement (striping) : cela consiste à assembler plusieurs petites unités de disque pour composer une grande unité logique. Aussi, l'information ne sera pas écrite en contigu sur une seule unité physique, mais sera décomposée et stockée sur plusieurs unités. Pour lire ou écrire une information, plusieurs disques interviennent, ce qui augmente le débit de transfert. Le facteur d'entrelacement indique la taille du fragment stocké sur une seule unité physique. Plus ce facteur est petit (1 octet), plus le débit de transfert moyen s'en trouvera amélioré par un transfert parallèle sur toutes les unités.

Avantages et inconvénients

Dans le cas de défaillance d'un seul des disques de l'ensemble, l'accès à la totalité des données est souvent impossible. Pour cette raison, on dit souvent que le système RAID 0 (stripping) n'est pas un véritable système RAID. Les performances sont excellentes, mais l'insécurité rend la solution inadaptée aux impératifs de l'entreprise.

RAID niveaux 1 et 0+1

Le RAID 1 définit la technique du disque miroir (*mirroring*). Le principe consiste à dupliquer toutes les données d'un disque (ou d'un périphérique RAID appelé RAID 10 ou RAID 0+1) sur un second disque.

Avantages et inconvénients

RAID 1 est une technique simple et facile à mettre en œuvre. Elle est cependant onéreuse car seule la moitié de la capacité totale de stockage est disponible pour vos données. Ses performances (proches du débit maximal d'un disque) ainsi que la sécurité sont excellentes.

RAID niveau 2

L'algorithme de détection et de correction d'erreurs utilisé pour le RAID 2 est l'ECC (*Error Correction Code*). Les données sont stockées suivant le même principe que le RAID 0 (répartition des bits sur plusieurs unités physiques), à la différence qu'on y ajoute des bits de contrôle ECC écrits sur une unité physique supplémentaire. On observe souvent 4 disques de données pour 3 disques ECC.

Avantages et inconvénients

La sécurité est excellente mais les performances sont médiocres.

RAID niveau 3

Par rapport au RAID 2, le niveau 3 remplace les unités ECC par un seul disque destiné au contrôle de parité.

Avantages et inconvénients

Les performances sont excellentes pour la lecture de nombreux blocs mais faibles pour les écritures de petits blocs, tout transfert physique étant précédé de nombreux accès en lecture, afin de localiser les blocs. Les écritures offrent ainsi des performances très moyennes.

RAID niveau 4

Le RAID niveau 4 est identique au RAID 3, excepté la valeur du facteur d'entrelacement. Cette technique présente un avantage considérable : la lecture d'un nombre de blocs réduit n'impose pas un accès physique à l'ensemble des unités du système, mais se limite aux unités où sont stockées les données.

Avantages et inconvénients

La capacité en lecture/écriture du RAID 4 est améliorée pour les petits transferts par rapport au RAID 3. En revanche, pour les gros transferts, le RAID 3 conserve l'avantage.

RAID niveau 5

RAID 3 et RAID 4 doivent accéder au disque de parité, en écriture comme en lecture. Pour RAID 5, les données et les indicateurs de parité sont répartis sur tous les disques du système.

Avantages et inconvénients

La sécurité en cas de défaillance d'un disque est excellente. Souvent, il suffit de changer le disque serveur en marche et les informations contenues sur les autres disques du RAID 5 le reconstruisent dynamiquement.

Les accès en lecture comme en écriture sont améliorés, puisque les accès physiques sont répartis sur plusieurs unités. Si le facteur d'entrelacement est trop faible, beaucoup trop de disques seront accédés même pour de petites requêtes.

En considérant le débit en Mo par seconde, un ensemble RAID 5 est nettement moins performant qu'un disque dur unique. Comme les contrôleurs de disque peuvent écrire de manière asynchrone, ces performances peuvent s'approcher de celles d'un RAID 0.

Les petites lectures sont excellentes à condition que les données soient contenues dans la taille d'un bloc d'entrelacement. Une lecture n'accède qu'à un bloc d'un disque, dans ce cas.

Comparaison des différents niveaux RAID

C'est l'association des deux critères performance et sécurité qui entraîne le choix d'une solution RAID. Si l'on ne recherche que la performance, une solution « classique » sans RAID sera pratiquement toujours préférable.

Bien souvent, le choix d'un système RAID se limite à deux possibilités : RAID 1 ou RAID 5. Ce sont eux, en effet, qui offrent les meilleurs rapports performance/sécurité.

Capacité de stockage

Les différentes manières d'assurer la sécurité influent sur le volume disque mis à votre disposition par rapport au volume total requis.

Pour le RAID, ce niveau se situe entre 80 % et 90 % du volume total, alors que le RAID 1 ne laisse que 50 % du volume total à votre disposition.

Coût d'acquisition

Ce critère est directement lié au précédent.

Sécurité

RAID 1 et RAID 5 sont excellents, l'un comme l'autre. Après un « crash » disque, le RAID 5 immobilise moins votre système, car il propose souvent la possibilité de reconstruire le

disque manquant à partir des informations de parité stockées sur les autres disques. Avec le RAID 1, on opère une copie disque à disque, ce qui mobilise à près de 100 % le disque en reconstruction, le rendant indisponible pour les autres opérations.

Performance

La performance respective des différents systèmes RAID est difficile à évaluer car les besoins sont nombreux, qu'il s'agisse d'opérations de lecture ou d'écriture, le volume des données concernées ou encore la fréquence et la charge transactionnelle. Chaque système présente des points forts et des points faibles en fonction de ces considérations.

Un RAID 1 sera excellent dans le cas d'une grosse charge d'écritures, alors que le RAID 5, qui doit « éclater » toutes les écritures en différents segments plus celui de parité, sera ainsi défavorisé. Ces critères sont inversés dans le cas de lectures massives.

Dans le cas du RAID 5, n'oubliez pas que le facteur d'entrelacement des disques doit être un multiple du paramètre `DB_BLOCK_SIZE` d'Oracle 10g, qui définit la taille des blocs échangés entre les fichiers de la base, la mémoire et les processus. S'il est plus petit, chaque bloc accédé impose l'accès à plusieurs disques. S'il n'en constitue pas un multiple, l'espace disque est gaspillé car un seul bloc Oracle 10g est contenu dans un bloc de stripping et la place libre est inutilisable.

RAID logiciel et RAID matériel

Windows permet de construire des partitions RAID *logicielles*. Toutes les opérations liées à la gestion du RAID seront effectuées par le processeur et la mémoire du serveur. Cela crée alors une surcharge.

En cas de charge disque plus importante, vous pouvez acquérir un RAID *matériel*, dans lequel les disques composant le RAID sont déportés et reliés au système par une interface à haut débit (de type SCSI ou autre).

Dans tous les cas, nous vous conseillons de croiser les informations en provenance de vos fournisseurs de matériel, avec celles en provenance de Microsoft. Et surtout, n'oubliez pas qu'avoir un système de disques performant et sécurisé ne vous dispense pas de réaliser des sauvegardes.

Coexistence RAID 1 et RAID 5

La figure suivante présente une configuration RAID à deux disques. Sur le premier, la partition RAID 1 est mise en « miroir ». Elle se retrouve donc sur le second disque. La partition RAID 5 occupe de l'espace sur les deux disques. Enfin, une partition libre subsiste sur le second disque (figure 27-1).

Plusieurs types de RAID peuvent coexister sur un même disque, mais les performances ne sont pas optimales : le RAID 1 qui est orienté « performance » coexiste alors avec un RAID 5 orienté « sécurité » (figure 27-2).

Dans cet exemple, les disques 1 et 2 sont placés en miroir en RAID 1 et les disques 3, 4 et 5 dans une partition RAID 5.

Figure 27-1

Exemple de configuration RAID

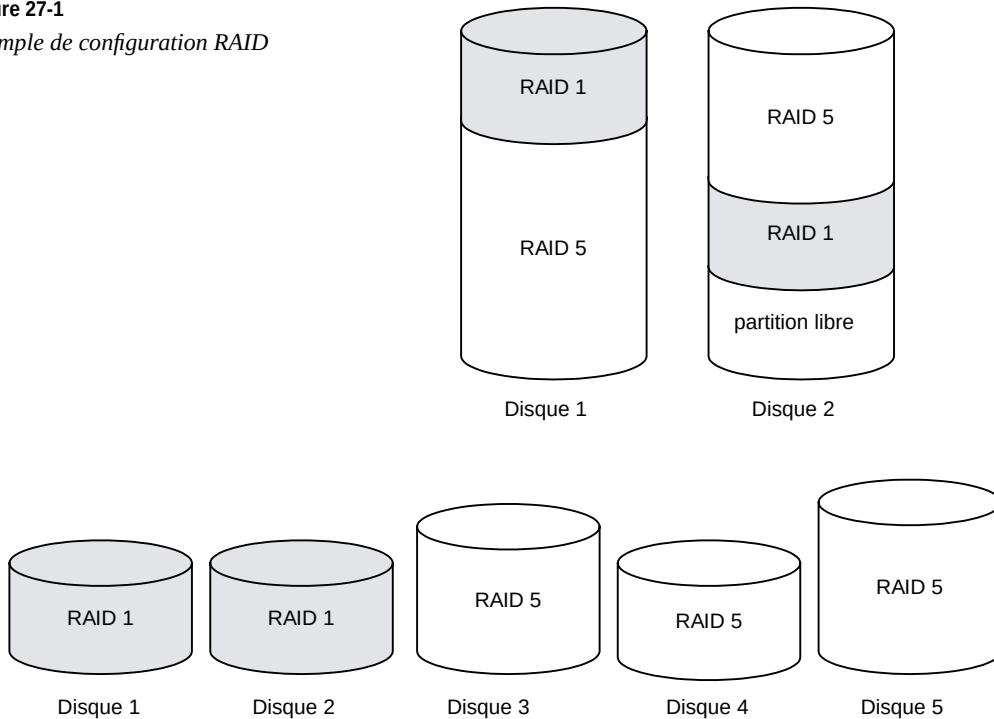


Figure 27-2

Répartition des RAID sur différents disques

Sur quel type de RAID placer les fichiers Oracle ?

Nous préconisons d'utiliser des partitions au format *NTFS*, mais le choix du niveau RAID doit être discuté pour une base de données.

N'oubliez pas que l'achat d'un contrôleur matériel RAID externe est plus performant que le RAID logiciel proposé par Windows, mais qu'il présente un surcoût.

Quelles règles faut-il suivre pour placer les fichiers d'une base Oracle sur ces partitions ? Tout d'abord, nous indiquons le nombre minimal de disques requis pour chaque niveau de RAID.

Niveau RAID	Description	Nombre minimum de disques requis
RAID 0	Entrelacement (striping) disque	2
RAID 1	Mirroring de disques	2
RAID 5	Entrelacement (striping) avec contrôle de parité	3
RAID 10 (1+0)	Entrelacement (striping) disque avec mirroring	4

Ensuite, nous vous proposons plusieurs niveaux RAID pour chaque type de fichier Oracle. **Nous indiquons le niveau RAID par ordre de performance décroissante.**

Type de fichier Oracle	Niveau RAID (performance)	Commentaires
Fichiers redo-log	RAID 0 RAID 1 RAID 5	
Fichiers de données d'une base (tablespace SYSTEM, données, index, rollback segments, temporaire)	RAID 1+0 RAID 1 RAID 5 RAID 0	Privilégie les écritures. Pour écritures moyennes. Écritures moyennes, lectures intensives. Pas de tolérance aux pannes, écritures intensives.
Fichiers de contrôle	Pas d'impact	La sécurité doit être maximale.
Logiciel Oracle et fichiers de configuration	RAID 1	Lecture excellente, sécurité assurée.

Enfin, en croisant les impératifs de performance et de sécurité, voici le choix que nous vous proposons.

RAID 1	Placer les logiciels Oracle (la « distribution ». Le mirroring RAID 1 protège les logiciels installés et les quelques fichiers d'initialisation (<i>pf1le</i> , fichier mot de passe, <i>listener.ora...</i>) qui sont « mélangés » avec les logiciels.
RAID 1	La vitesse d'écriture doit être privilégiée pour les fichiers redo-log . Comme ils peuvent être multiplexés dans des « group », la sécurité est assurée et le mirroring RAID 1 convient.
RAID 5	La disponibilité des fichiers devant être optimale, placer tous les autres fichiers d'une base Oracle en RAID 5 : fichiers de données , et fichiers de contrôle . L'idéal côté performance serait de placer tous ces fichiers en RAID 1, mais il faudrait alors acheter un volume disque double de celui des données à héberger. RAID 5 offre plus de sécurité et constitue une bonne alternative.

Ces recommandations concernent aussi le chapitre 28, *Optimisation et performances*.

Les partitions de type Raw Devices

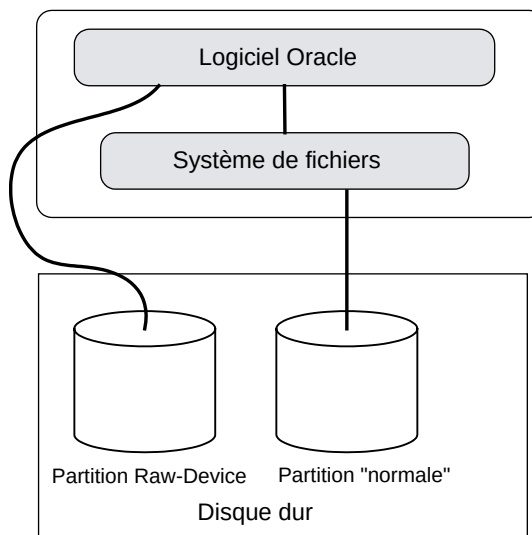
Les partitions de type Raw Devices offrent un accès plus rapide aux disques durs que d'autres partitions. Elles permettent d'exploiter des portions de disques durs qui ne sont pas affectées à des systèmes de fichiers. Les mécanismes d'échange entre Oracle 10g et les fichiers Raw Devices court-circuitent complètement le système de fichiers Windows. Les lectures et les écritures sont directes entre Oracle 10g et les partitions Raw Devices.

Dans le cas d'utilisation de fichiers Raw Devices, ils apparaissent comme des partitions non formatées au niveau du gestionnaire de disques.

Ce système de fichiers est le plus performant. On estime les gains d'accès en entrées/sorties disque, entre 2 % et 5 %. Cependant, l'exploitation est beaucoup plus délicate car vous manipulez des fichiers qui ne sont pas visibles depuis le système d'exploitation.

Figure 27-3

*Principe des partitions Raw
Devices*



Dans les configurations de type Oracle Real Application Clusters les machines composant le cluster accèdent simultanément à un jeu de disques commun. Les fichiers de la base sont obligatoirement du type Raw Devices. Cette possibilité est maintenant utilisée depuis Oracle 10g. Même si les performances disques sont accrues, ce type de partition est difficile à manipuler et complexifie les opérations de sauvegarde des bases.

Automatic Storage Management (ASM)

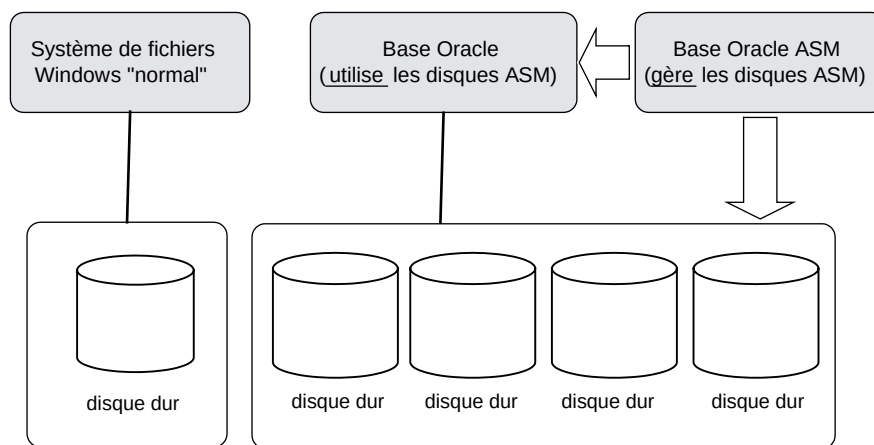
Oracle Automatic Storage Management (ASM) est annoncé en fanfare avec Oracle 10g. Très simplifié, ASM est un système de fichiers spécialisé pour répondre aux besoins d'une base de données. Ses principales caractéristiques sont :

- simplifie l'administration quotidienne ;
- procure des performances similaires à celles de fichiers « Raw Devices » ;
- permet de construire des configurations puissantes, performantes, avec une grande taille disque consacrée à Oracle à partir de disques de faible coût unitaire. C'est l'un des enjeux du Grid ;
- le logiciel Oracle, les fichiers de configuration sont toujours installés sur le système de fichiers du serveur. Seul l'ensemble des fichiers de la base de données est gérable sous ASM ;

- par un assemblage de disques durs ou de partitions d'un NAS, pratiquement n'importe quel niveau de tolérance aux pannes peut être construit : Raid, Mirroring, Stripping. Vous n'avez plus besoin de logiciels tiers supplémentaires pour cela ;
- l'équilibrage de la charge des données sur les différents disques est automatique, même en cas d'ajout d'un disque ;
- toute la gestion physique des disques et fichiers des bases est simplifiée ;
- les fichiers ASM n'étant pas visibles depuis le système d'exploitation, la sécurité est renforcée ;
- une base de données Oracle ASM doit être lancée. D'une faible consommation mémoire et CPU, elle est utilisée pour indiquer aux bases Oracle du serveur où sont situées les structures ASM qui les concernent.

La lumière apportant aussi l'ombre, voici quelques limites d'ASM :

- c'est une technologie récente, qu'il convient de tester soigneusement avant de la mettre en production ;
- les fichiers ASM ne sont pas visibles depuis le système d'exploitation, tout comme un disque présent sur un ordinateur mais qui ne contient pas encore de partition ;
- pour effectuer les sauvegardes physiques, vous devez obligatoirement utiliser Recovery Manager (RMAN), qui nécessite lui aussi une base dédiée. Avec la base ASM, cela fait deux bases nécessaires pour une en production ! Bien entendu, ce propos doit être nuancé, la base RMAN pouvant être partagée par plusieurs instances à sauvegarder.



Configuration disques "ASM"

Figure 27-4

Principe d'ASM

Il convient donc de bien évaluer les apports d'ASM et les contraintes qu'il engendre. Toutefois, c'est une technique qui semble très prometteuse. Il conviendra de surveiller l'activité des forums Oracle sur <http://otn.oracle.com> pour suivre son adoption par la communauté Oracle ainsi que les difficultés rencontrées.

Résumé

Ce chapitre intéresse à la fois l'administrateur système et celui d'Oracle 10g. Sur les systèmes Windows, c'est souvent une seule et même personne !

Toutes les informations ont été fournies dans le souci de vous aider à choisir et à appliquer une solution matérielle adaptée à vos besoins. On accorde beaucoup de temps au choix du type de processeur, du volume mémoire, etc., moins au choix et à la mise en œuvre des disques, en dépit de leur importance pour la performance globale d'une base de données.

Enfin, n'oubliez pas qu'il est difficile de modifier une configuration disque une fois établie : choisissez les bonnes options dès le début !

Optimisation et performances

Dans ce chapitre :

- identification des contentions ;
- le tuning de Windows pour Oracle 10g ;
- les étapes à suivre pour réussir un tuning ;
- les outils Oracle 10g disponibles.

Comment tirer le meilleur parti d'Oracle 10g sous Windows ? Le sujet est vaste : l'optimisation ou « tuning » consiste à établir le meilleur niveau de performances d'un système en tenant compte de contraintes souvent contradictoires. Notre objectif est de vous présenter les éléments fondamentaux pour assurer des performances aussi bonnes que possible.

Si les nouveautés d'Oracle 10g sont nombreuses, beaucoup vont dans le sens d'une simplification de l'administration et une amélioration des performances. Ce chapitre se trouve allégé par rapport aux précédentes éditions de ce livre.

Le chapitre précédent a abordé le fonctionnement du système de stockage de données. Nous vous proposons de le lire préalablement, car l'optimisation est limitée si le système de fichiers constitue un goulet d'étranglement.

Nous indiquons comment optimiser Windows pour obtenir des performances maximales dans un contexte de base de données et précisons les paramètres d'Oracle 10g qui permettront d'améliorer encore ces performances.

Pourtant, il n'existe pas de recette magique pour optimiser une application. Le tuning doit faire **partie intégrante** de la conception d'une application. Trop souvent, on ne se soucie

d'optimisation que lorsque l'application ne fonctionne pas, que les utilisateurs sont mécontents, bref que la situation est explosive ! Pourtant, **plus de 90 % de l'optimisation possible provient de l'application et du design de la base de données**. Ne cherchez pas le paramètre PERFORMANCE=MAXI dans les paramètres d'initialisation d'une base : il n'existe pas !

Le manuel d'optimisation d'Oracle 10g fait à lui seul plus de 500 pages. Je n'ai pas la prétention de vous apporter une information exhaustive sur ce thème mais les pistes essentielles.

L'optimisation de Windows

Comparé à Unix, Windows dispose de beaucoup moins de leviers pour ajuster les performances globales du système. Cela peut être gênant pour de grands systèmes Windows mais c'est en général un avantage pour toutes les configurations intermédiaires.

La plupart des actions menées au niveau de Windows ont pour but de limiter les ressources consommées par des tâches autres que la base de données. Pour tirer le meilleur d'Oracle, ne surchargez pas votre système par des tâches Windows annexes :

- Serveur de fichiers et d'imprimantes ;
- Serveur Active Directory ou contrôleur de domaine ;
- Passerelle d'accès distant ;
- Serveur DNS, etc.

Concernant le tuning Windows, les principales recommandations sont les suivantes.

Le support des larges pages

L'utilisation par Oracle des *larges pages* sur un serveur Windows 2003 économise le CPU et tire le meilleur parti des ressources mémoires : au lieu de manipuler des portions mémoire de 4 Ko, l'option large page permet de les passer à 4 Mo par portion.

Cette option est particulièrement utile si la taille des données en SGA est de plusieurs gigaoctets. Pour de petites configurations, le gain sera négligeable, voire négatif.

Cette option s'active sous Windows 2003 serveur en créant la valeur ORA_LPENABLE=1 dans les clés Oracle de la base de registre.

Réduire la priorité des applications de premier plan

Donnez un maximum de temps de traitement de Windows pour les tâches de fond, c'est-à-dire la base de données.

À partir du panneau de configuration, lancez l'utilitaire de configuration du système : Démarrer > Paramètres > Panneau de configuration > Système. Affectez un maximum de temps processeur et de ressource mémoire aux applications fonctionnant en arrière-plan.

Figure 28-1
Propriétés système avancées

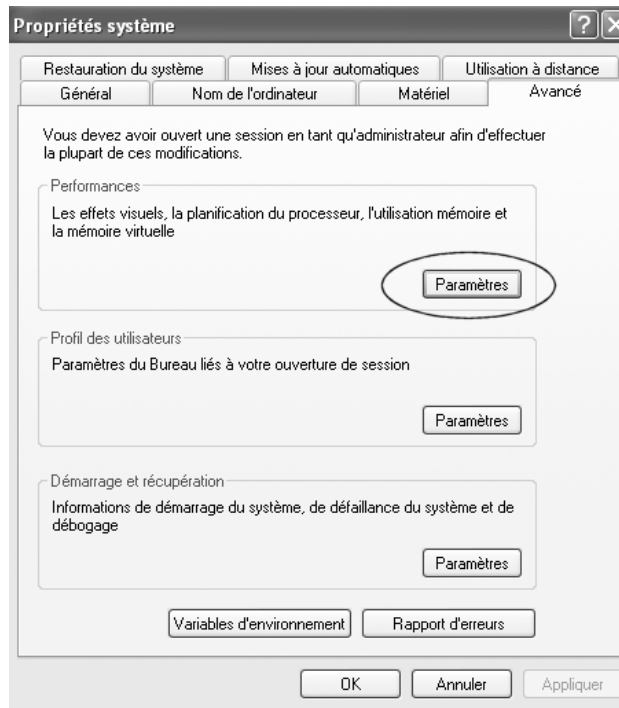
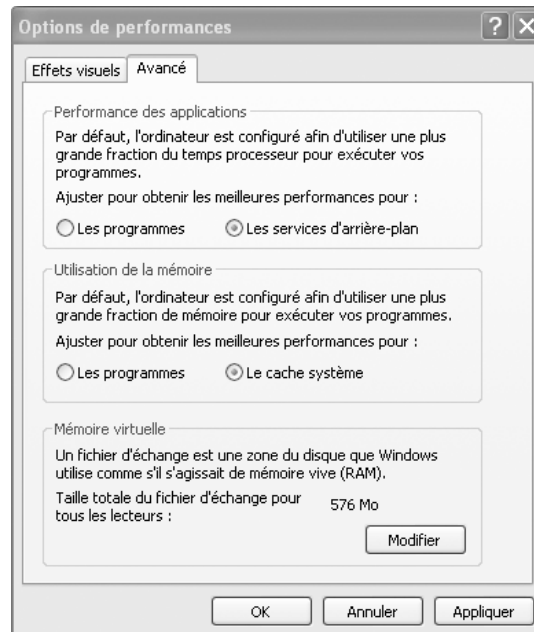


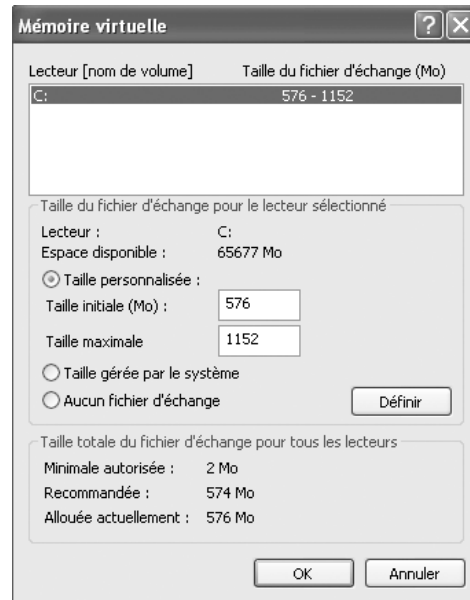
Figure 28-2
Privilégier la mémoire et le CPU pour Oracle



C'est aussi à partir de cet écran que vous définissez la taille du fichier d'échange de votre système, c'est-à-dire sa zone de pagination qui servira à stocker sur disque, lorsque la taille de la mémoire disponible est dépassée, des informations normalement en mémoire. L'utilisation de ce fichier d'échange est à surveiller, nous verrons comment dans le même chapitre, car les accès disques sont beaucoup plus lents que les accès mémoire.

Figure 28-3

Définition du fichier de pagination



Dans cet exemple, il n'existe qu'un seul fichier de pagination. Pour améliorer l'accès disque, on peut créer plusieurs fichiers de pagination. Placez-les toujours sur des partitions disques rapides : RAID 0,1,10 mais évitez RAID 5. La taille de l'ensemble des fichiers de pagination doit être suffisante pour contenir tous les « débordements » du système. Une taille trop grande ne présente aucun inconvénient : prévoyez large, de 4 à 6 fois la taille de la mémoire vive.

Supprimez les services non indispensables

De nombreux services sont lancés par Windows et Oracle. Annulez tous ceux qui ne sont pas indispensables à votre configuration. À partir du *Panneau de configuration*, choisissez l'icône *Services* et ne démarrez que les services indispensables.

Vous pouvez arrêter chaque service et définir ses propriétés de démarrage lors du prochain lancement de votre machine.

Certains services peuvent être stoppés :

- Plug and Play ;
- Gestionnaire de connexions d'accès distant ;

- Gestionnaire de numérotation automatique d'accès distant ;
- Serveur d'accès distant ;
- Service Téléphonie ;
- OracleAgent (en dehors de la phase de découverte d'Oracle Enterprise Manager et du lancement de Jobs et d'Events) ;
- ... et tous ceux qui ne sont ni utiles, ni utilisés dans votre contexte.

D'autres ne doivent pas être stoppés :

- Enregistrement d'événements ;
- Messagerie ;
- Service d'appels RPC ;
- Serveur ;
- Planning ;
- Aide TCP/IP NetBIOS ;
- Système d'exploitation ;
- Station de travail ;
- OracleServiceSID ;
- OracleOraHome92TNSListener80.

Supprimez les protocoles inutiles et adaptez leur ordre

Différents protocoles sont peut-être installés sur votre serveur et certains peuvent être supprimés. Accédez à l'écran par Démarrer > Paramètres > Configuration réseau.

Face à l'omniprésence de TCP/IP dans les réseaux, il est maintenant rare de rencontrer plusieurs protocoles sur un même serveur. Certains, comme SMB, peuvent être supprimés d'un serveur Oracle.

Supprimez les protocoles réseau qui ne sont pas indispensables à votre configuration. Attention, ceci peut avoir des conséquences importantes sur votre serveur (bien que tous les protocoles puissent être réinstallés) et doit être fait en collaboration avec votre administrateur réseau.

Si plusieurs protocoles sont utilisés sur votre serveur, donnez-leur un ordre qui met en tête le protocole le plus utilisé par votre base Oracle. Ce protocole est majoritairement TCP/IP.

Utilisez le dernier Service Pack de Windows

Microsoft améliore Windows en sortant régulièrement des correctifs ou Service Pack. Ils comprennent des améliorations touchant les performances, la sécurité, des corrections de bugs, etc. Ils sont numérotés et doivent être lancés dans l'ordre de leur parution : Service Pack 1, puis 2, puis 3...

Pour les systèmes Windows en langue française, attendez que Microsoft édite le Service Pack en français. N'installez jamais des Services Pack américains sur des Windows français.

N'installez pas de nouveau service Pack sans avoir préalablement vérifié sa compatibilité avec la version Oracle sur le site <http://metalink.oracle.com>.

Les outils disponibles sous Windows

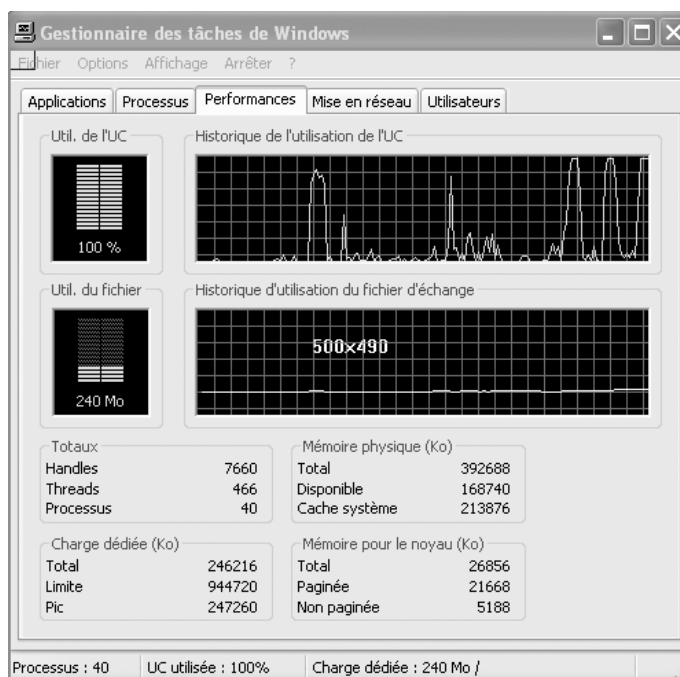
Nous vous présentons maintenant des outils Windows utiles pour surveiller l'activité du système. Certains peuvent aussi être utilisés pour surveiller des bases Oracle.

Le gestionnaire des tâches Windows

Le gestionnaire de tâches se lance en effectuant un « clic droit » sur la barre des tâches Windows.

Figure 28-4

Gestionnaire des tâches



Le gestionnaire des tâches visualise la quantité de mémoire, physique et logique, consommée par le système. Sur la figure précédente, la *charge dédiée* vous indique si le système pagine. Lorsque l'utilisation de la mémoire s'approche de sa limite maximale,

les performances sont dégradées et vous risquez un blocage du système : il est urgent d'agir ! Vous pouvez :

- arrêter des programmes consommateurs ;
- augmenter la taille du fichier de pagination ;
- augmenter la mémoire du système.

L'analyseur de performances Windows

Le suivi des performances entrées/sorties disque nécessite un paramétrage de Windows. En effet, les compteurs de performance ne sont pas disponibles et il faut les installer. L'installation des compteurs s'effectue en lançant diskperf.exe :

```
C:\>diskperf.exe -Y
```

Les compteurs de performance des disques logique et physique de ce système sont maintenant initialisés pour être lancés lors du démarrage Windows.

```
# Vous devez relancer Windows pour activer la mesure.
```

N'oubliez pas d'arrêter la collecte des informations par diskperf -N.

Les indicateurs disponibles pour les disques logique et physique sont :

- Longueur moyenne de la file d'attente (pourcentage moyen de l'attente en E/S) ;
- Longueur actuelle de la file d'attente (vue instantanée de la file d'attente en E/S) ;
- Nombre de bytes par transfert (volume de données échangées par E/S).

L'analyseur de performances Windows pour Oracle

Par défaut, tous les compteurs proposés ne concernent que Windows. Oracle a donc ajouté ses propres compteurs relatifs à la surveillance d'une base.

Une fois installés, il faut préciser quelle base il convient d'étudier. L'utilitaire operfcfg.exe inscrit les informations conservées en : HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\SERVICES\ORACLE9\Performance.

```
Usage: OPerfCfg [-U <username>] [-P <password>] [-D [database name]]
```

```
C:\> operfcfg -U system -P mot_de_passe -D PROD
```

Attention, le compte system et son mot de passe figurent en clair dans la base de registres ! N'utilisez pas cette option si vous avez des contraintes de sécurité. Choisissez plutôt Oracle Enterprise Manager qui dispose maintenant de graphiques de performances.

L'utilitaire Page Fault Monitor

Cet utilitaire du Kit de Ressources Techniques Microsoft analyse la pagination (*page fault*) causée par un processus. Les informations fournies sont très détaillées, nous vous conseillons d'écrire les données dans un fichier qui sera analysé par la suite. Les dernières lignes de l'analyse sont les plus intéressantes.

```
C:\> pfmon.exe oracle.exe > c:\fichier_texte_pour_analyse.txt
# La fin du fichier texte contient les informations suivantes :
```

oracle	Caused	1038 faults had	1073 Soft	11 Hard faulted VA's
ntdll	Caused	1749 faults had	49 Soft	4 Hard faulted VA's
oraclnt9	Caused	7 faults had	18 Soft	3 Hard faulted VA's
oracore9	Caused	18 faults had	20 Soft	5 Hard faulted VA's
oranls9	Caused	5 faults had	8 Soft	3 Hard faulted VA's
oraunls9	Caused	4 faults had	4 Soft	3 Hard faulted VA's
	Caused	41 faults had	24 Soft	3 Hard faulted VA's
kernel32	Caused	49 faults had	36 Soft	2 Hard faulted VA's
wsock32	Caused	0 faults had	2 Soft	0 Hard faulted VA's
ws2_32	Caused	10 faults had	10 Soft	2 Hard faulted VA's

Fichier d'échange utilisé par oracle.exe

Pour surveiller l'activité du fichier de pagination générée par le processus `oracle.exe`, vous pouvez aussi utiliser le compteur *Processus : Fichier d'échange (octets)* de l'analyseur de performance de Windows.

Fermez toutes les applications inutiles

Enfin, voici quelques conseils utiles pour un serveur hébergeant une base Oracle. Comme pour les services, fermez toutes les applications inutiles qui fonctionnent sur votre serveur :

Les applications démarrées au lancement de la machine

Certaines sont démarrées au lancement de la machine comme les barres d'outil de Microsoft Office. Ce sont des applications très consommatrices qui n'ont pas leur place sur un serveur base de données.

Les économiseurs d'écran

On les oublie souvent, mais ce sont de véritables catastrophes en terme de consommation processeur et mémoire. S'ils travaillent sur une station autonome, pourquoi pas, mais dans le cadre d'un serveur base de données, ne laissez pas un pourcentage important de vos ressources être gaspillé.

L'économiseur d'écran est accessible par un clic droit sur le bureau Windows.

Figure 28-5
Écran de veille



Choisissez l'économiseur le plus simple possible, en évitant les « Flying Toasters » et autres fractales, grands consommateurs de processeur.

Les étapes du tuning

Oracle 10g est un logiciel souple possédant de nombreuses options de paramétrage. Il s'adapte aussi bien aux petites configurations qu'aux très grosses bases de données. D'où proviennent alors de mauvaises performances ? Bien souvent, le tuning est oublié durant la conception de l'application et les applications sont testées sur de petites configurations avec peu de données. Elles sont ensuite déployées et le volume de données n'a plus rien en commun avec ce qu'il était initialement. Des problèmes, jusque-là masqués, apparaissent alors. Cette raison est la première cause nécessitant des actions de tuning.

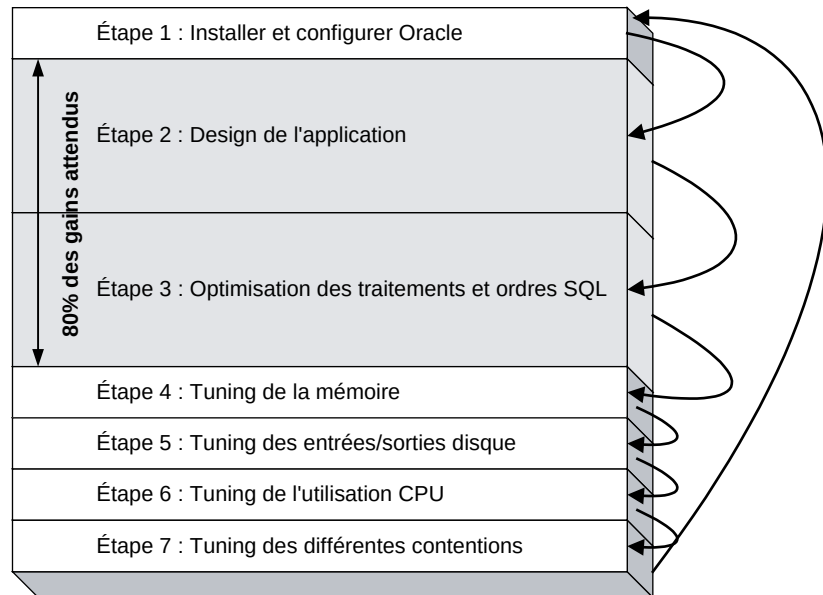
Vous pouvez améliorer de façon spectaculaire les performances en corrigeant des erreurs courantes **provenant de l'application** ou du paramétrage. Nous vous conseillons d'effectuer un tuning étape par étape.

Plus de 80 % des gains attendus par le tuning proviennent du design de l'application et de l'optimisation des traitements et ordres SQL.

Certaines étapes nécessitent d'utiliser des statistiques issues de SQL*Plus, l'outil Scratchpad d'Oracle Enterprise Manager ou encore les résultats des scripts SQL `utlbstat/utles-tat`. D'autres demandent la consultation des tables V\$ qui contiennent des informations dynamiques sur l'état de la base de données. Enfin, l'étude des caractéristiques du

système Windows, comme la mémoire, les entrées/sorties disque ou le temps processeur consommé, apportent des réponses pour orienter d'éventuelles actions.

Figure 28-6
Étapes d'un processus de tuning



La figure précédente illustre les étapes d'un processus de tuning. La taille respective de chaque étape indique les gains attendus. Bien que l'échelle soit volontairement imprécise, le design de l'application et l'optimisation des ordres SQL sont les étapes où les gains de performance sont les plus importants.

L'optimisation des performances d'un système doit être réalisée avec méthode. Chaque action doit être notée et son impact positif, négatif ou nul, mesuré. N'effectuez pas plusieurs actions simultanément, vous ne sauriez isoler celle qui est la plus efficace. De plus, certaines actions bénéfiques peuvent être annulées par d'autres qui diminuent les performances.

Pour chaque étape, les actions les plus importantes sont :

Étape 1 : Installer et configurer Oracle 10g

- installer et paramétrer Oracle 10g en utilisant les règles Optimal Flexible Architecture (OFA).

Étape 2 : Design de l'application

- choix des paramètres initiaux de la base ;
- schéma physique de la base ;

- techniques de développement des applications.

Étape 3 : Optimisation des traitements et ordres SQL

- mise à jour des statistiques de la base ;
- tuning des ordres SQL avec TKPROF, EXPLAIN PLAN et des outils d'Oracle Enterprise Manager.

Étape 4 : Tuning de la mémoire

- mémoire Oracle : tuning du nombre de buffers de données, buffers redo-log et d'autres paramètres de la System Global Area (SGA) ;
- tuning de la mémoire Windows en réduisant la pagination.

Étape 5 : Tuning des entrées/sorties disques

- répartir les entrées/sorties et les applications.

Étape 6 : Tuning de l'utilisation CPU

- répartition des applications ;
- réorganisation des processus.

Étape 7 : Tuning des différentes contentions

vision très fine des paramètres d'optimisation d'Oracle 10g.

Il est important de suivre le déroulement logique de ces étapes car des changements effectués sur l'une d'elles influent sur les suivantes. Par exemple, la modification d'un ordre SQL dans l'étape 3 influera sur les ressources mémoire, CPU et disque consommées, c'est-à-dire les étapes 4, 5 et 6.

Face à une situation où l'on constate de très mauvaises performances, la bonne approche consiste à rechercher le point de contention pour le supprimer. Vous pouvez commencer un tuning par étape, une fois ce point corrigé, car il masquait toutes les autres caractéristiques.

Certaines actions peuvent dégrader des points visualisés dans les étapes précédentes. C'est pourquoi le tuning doit être effectué en « boucle », en contrôlant la stabilité du résultat obtenu dans les précédentes étapes.

On peut facilement passer des journées entières à effectuer le tuning d'un système. Fixez, avant de commencer, des objectifs et un temps limité. N'hésitez pas à stopper votre tuning une fois ces objectifs atteints.

Identification des contentions

Avant d'effectuer le tuning d'un système, il est important de connaître les différentes contentions qui peuvent survenir. Bien souvent, un système est lent à cause d'un goulot d'étranglement qui limite les performances de l'ensemble. Les trois ressources systèmes où peut apparaître une contention sont : la mémoire, les systèmes d'entrée/sortie disque et réseau et enfin le ou les processeurs.

Sur un ordinateur, la principale cause de contention survient lorsque les processus ont besoin de plus de mémoire que celle disponible sur le système. Pour contourner ce problème, le système utilise la pagination, c'est-à-dire le transfert sur disque de pages mémoire. Il est important de surveiller ce point qui peut, à terme, fortement dégrader les performances du système. Des ordres SQL permettent de surveiller la contention mémoire côté Oracle 10g et des utilitaires graphiques côté Windows.

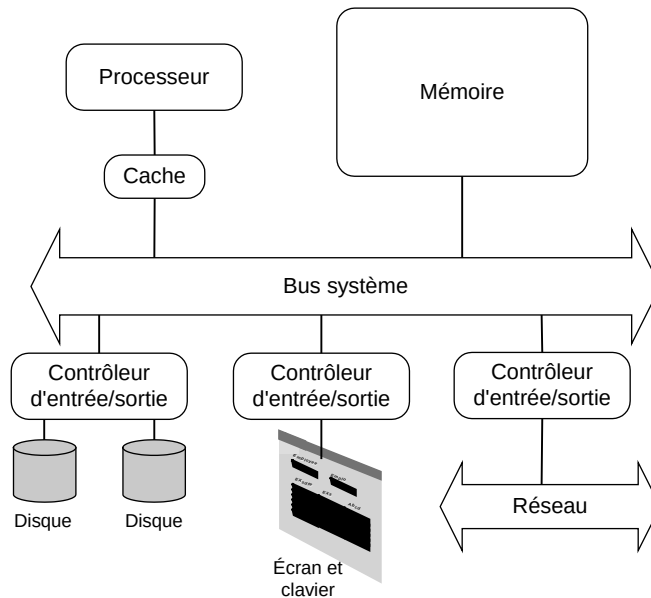


Figure 28-7

Rappel des composants d'un ordinateur

La contention disque peut provenir d'une mémoire mal gérée (avec des débordements disque sous forme de pagination) ou encore d'une mauvaise distribution des fichiers de base Oracle 10g, sur le système de fichiers Windows. Des outils système ou Oracle permettent d'étudier ces aspects.

Le système peut aussi être à court de ressource processeur. Bien que Windows la répartisse sur les différents processus de façon équitable, des outils comme l'analyseur de performance Windows permettent d'étudier la consommation globale du système.

La contention peut provenir d'Oracle 10g. Par exemple, des transactions en cours qui mettent en attente d'autres transactions. Pour les identifier rapidement, les table V\$, les scripts *utlbstat/utlestat* ou encore des outils d'Oracle Enterprise Manager sont d'une grande efficacité.

Nous mentionnerons aussi des outils autres que ceux d'Oracle qui peuvent être précieux dans l'analyse des consommations.

Étape 1 : installer et configurer Oracle 10g

Une installation Windows soigneuse est bénéfique car elle limite l'apparition de goulots d'étranglement.

Choisir des disques durs

La base Oracle 10g est conçue pour utiliser massivement la mémoire, beaucoup plus rapide d'accès que les disques. Le chapitre 27, *Configurer les disques pour Oracle*, aborde ce sujet, préalable important à toute installation.

Préparer une installation respectant les règles OFA

L'Optimal Flexible Architecture (OFA) définit un ensemble de règles à respecter pour faciliter le travail d'administrateur. Comme lors du choix des disques ou des droits d'accès des utilisateurs, OFA vous permet d'effectuer des choix initiaux pertinents. Les règles OFA sont décrites aux chapitres 9, *Les fichiers d'une base Oracle 10g* et 13, *L'installation d'Oracle 10g sous Windows*.

Le DB_BLOCK_SIZE d'Oracle 10g

Il y a de gros gains de performance à attendre de ce paramètre. Il a été étudié au chapitre 13, *L'installation d'Oracle 10g sous Windows* et 23, *Gestion de l'espace disque et des fichiers*. Il définit l'unité de taille qui servira de fondement à tous les échanges internes à la base de données, au niveau des disques et de la mémoire. Sous Windows, on rencontre principalement les valeurs 2 K, 4 K, 8 K ou 16 K.

Avec les ordinateurs actuels, un DB_BLOCK_SIZE=8K est maintenant une valeur courante.

Ce paramètre est figé pour le tablespace SYSTEM lors de la création de la base de données. Il ne peut être modifié par la suite, mais des tablespaces possédant une taille de bloc différente peuvent être créés. Il est défini dans le fichier d'initialisation et peut être vérifié en lançant l'ordre SQL suivant à partir de SQL*Plus :

```
SQL> connect system
SQL> select name, value from v$parameter where name='DB_BLOCK_SIZE' ;
```

Suivant la valeur de ce paramètre, vous pouvez réduire très fortement le nombre d'accès à vos disques. Entre un DB_BLOCK_SIZE à 2 Ko et 8 Ko les accès disque peuvent être diminués d'un facteur quatre ! Encore faut-il que l'application en bénéficie. On pratique souvent la répartition suivante :

- 2 Ko et 4 Ko : pour les applications réalisant beaucoup de transactions, qui accèdent et manipulent peu de données à la fois ;
- 8 Ko : valeur intermédiaire, correcte pour les applications transactionnelles et des requêtes balayant beaucoup de blocs de données sur disque ;
- 16 Ko : pour les applications décisionnelles : infocentre, datawarehouse, qui accèdent toutes à de gros volumes de données.

Pour que ce gain de performance soit valable, n'oubliez pas que la taille du DB_BLOCK_SIZE choisi doit être un multiple du facteur d'entrelacement (stripping) de vos disques RAID.

Étape 2 : design de l'application

Le design de l'application est **primordial pour bénéficier de bonnes performances**. Ce livre n'a pas pour objectif de vous enseigner comment concevoir le schéma conceptuel, logique et physique d'une base de données. Seules quelques recommandations générales sont abordées.

Même avec un tuning parfait, *vous ne compenserez jamais une application défaillante*. Les règles de l'art du développement doivent être respectées en préalable à toute optimisation.

Identifier les points sensibles

Tant que le volume de données est faible, il y a peu de risques que des problèmes apparaissent. Des tables plus volumineuses révèlent des problèmes jusque-là masqués.

Les points sensibles concernent principalement un schéma conceptuel mal élaboré, qui oblige à effectuer un nombre très important de jointures pour des opérations constituant le cœur de l'application, des index oubliés et des ordres SQL mal construits.

Des index oubliés

Assurez-vous que les clés primaires des tables possèdent toutes un index. Une clé primaire étant conçue pour effectuer des jointures avec d'autres clés primaires ou des clés secondaires, vérifiez que ces dernières sont elles aussi indexées.

Si l'application interroge vos tables sur des colonnes non indexées, étudiez la possibilité de créer des index sur ces colonnes pour diminuer le temps d'exécution des ordres SQL. Pour autant, ne tombez pas dans le travers qui consiste à indexer toutes les colonnes d'une table !

Des ordres SQL mal construits

Le langage SQL est très puissant : on a coutume de dire qu'il faut une journée pour l'apprendre et un an pour le maîtriser. De très nombreuses applications effectuent des ordres SQL simples mais passent un temps extrêmement long à effectuer des opérations sur les données, alors que les deux auraient pu être intégrés directement dans l'ordre SQL initial.

À l'annexe 3, Procédures pour le DBA, consultez l'ordre SQL « Taille totale et place libre dans les tablespaces ». Vous comprendrez rapidement l'utilité de ces ordres SQL évolués.

Étape 3 : optimisation des traitements et ordres SQL

La base de données et l'application sont opérationnelles. Avant d'effectuer le tuning mémoire, disque et processeur du système, il est important de vérifier que l'application est correctement écrite et que l'environnement technique de la base permet aux ordres SQL de s'exécuter dans les meilleurs délais.

Comment détecter les problèmes ?

Ce type de problème se détecte par de mauvaises performances ou une dégradation progressive des temps de réponse au fur et à mesure de l'évolution de la base de données. Souvent, il est révélé par les utilisateurs qui vous contactent et signalent que les temps de réponse augmentent.

Pour chaque application, il est assez facile de sélectionner quelques requêtes à surveiller. Construisez un jeu de test lancé régulièrement pour lequel vous enregistrerez les résultats mesurés. Cela vous permettra de disposer de courbes de tendance et d'effectuer des actions préventives.

L'organisation physique de la base de données

Dans une base Oracle 10g, l'organisation physique doit respecter la règle fondamentale du rangement : une place pour chaque chose et chaque chose à sa place. La base de données ne déroge pas à cette règle : si tout est bien organisé, elle travaille plus vite avec moins d'efforts.

Quelques points importants à vérifier : la taille adéquate du DB_BLOCK_SIZE, l'absence de fragmentation et de chaînage pour les tables et les index et enfin la pertinence des statistiques mises à la disposition de l'optimiseur d'Oracle 10g.

Disposer de statistiques à jour

Lorsque Oracle 10g reçoit une requête SQL, il décide, en interne, comment la traiter de manière optimale. C'est tout à fait similaire au choix d'une route lorsque vous êtes en voiture. Vous pouvez utiliser des règles, par exemple « je ne prends que les autoroutes », ou disposer d'un système GPS capable de vous faire éviter les travaux, les bouchons, etc.

Avec la 10g, Oracle abandonne l'ancien optimiseur à base de règles. C'est maintenant l'optimiseur basé sur des statistiques, plus performant, qui est l'option par défaut. Par conséquent, les valeurs CHOOSE et RULE du paramètre d'initialisation OPTIMIZER_MODE ne sont plus supportées.

Il est primordial de disposer d'analyses actualisées. C'est le rôle de la collecte de statistiques qui est effectuée automatiquement par Oracle pour tous les schémas. Elle s'effectue pendant la construction ou la reconstruction d'un index ou manuellement à l'aide du programme GATHER_STATS_PROG ou du job GATHER_STATS_JOB.

Le job GATHER_STATS_JOB est créé en même temps que toute base Oracle 10g. Il exécute la procédure DBMS_STATS. GATHER_DATABASE_STATS_JOB_PROC qui collecte toutes les statistiques. Ce job s'exécute lors d'une « fenêtre de maintenance » entre 21 h et 6 h du matin ainsi que le week-end. Vous avez la possibilité de modifier ces horaires s'ils ne vous conviennent pas.

Encore une nouveauté d'Oracle 10g : plus besoin de collecter les statistiques, Oracle s'en charge tout seul !

Le programme GATHER_STATS_PROG lancé manuellement collecte les statistiques sur les objets n'en possédant pas ou si elles sont obsolètes.

Les segments d'annulation

Contrairement aux rollbacks segments, son prédécesseur, le segment d'annulation ou UNDO tablespace ne nécessite plus aucune optimisation particulière. Ils sont tous deux étudiés au chapitre 11, *Transactions et accès concurrents*.

Ils contiennent les informations nécessaires pour annuler les modifications effectuées par une transaction. Ils sont utilisés essentiellement par les transactions qui doivent :

- être annulées (rollback) ;
- permettre une lecture consistante.

Vous pouvez utiliser Oracle Enterprise Manager pour consulter le contenu de vos segments d'annulation.

Les fichiers redo-log

De nombreux sites fonctionnent avec des fichiers redo-log sous-dimensionnés. Cela crée un point de contention et la base se met en attente d'écriture dans les redo-log.

L'utilisation des vues V\$LOGHIST et V\$LOG_HISTORY indique quand un fichier redo-log est plein et qu'il bascule vers un autre. La durée conseillée est de 20 minutes minimum.

La taille optimale peut être consultée en interrogeant la colonne OPTIMAL_LOGFILE_SIZE de la vue V\$INSTANCE_RECOVERY.

Si la base auditée montre une fréquence plus rapprochée, augmentez la taille des fichiers redo-log. Ce point est détaillé au chapitre 23, *Gestion de l'espace disque et des fichiers*.

L'optimisation des ordres SQL : utilisation de SQL*Plus

Face à une requête SQL longue, que faire ? Le plus simple est de vérifier la manière dont le moteur d'Oracle 10g la traite. Ce plan d'exécution vous renseigne totalement. Il permet d'observer les index utilisés (ou non utilisés), l'ordre de balayage des tables, etc. C'est le point de départ de tout tuning applicatif sérieux.

Le plan d'exécution des ordres SQL peut être visualisé graphiquement à l'aide d'Oracle Enterprise Manager ou d'une façon plus simple avec SQL*Plus. La première méthode est abordée au chapitre 25, *Oracle Enterprise Manager* et elle est moins complète que celle présentée maintenant.

SQL Tuning Advisor

Cette nouveauté d'Oracle 10g automatise les opérations de tuning d'ordres SQL. Elle est accessible en ligne, par des procédures et des fonctions ou une procédure PL/SQL. Vous pouvez aussi y accéder à partir d'Oracle Enterprise Manager pour examiner un ordre SQL :

1. Connectez-vous à Oracle Enterprise Manager.
2. En bas de la page d'accueil, dans la rubrique *Liens associés*, sélectionnez *Gérer les mesures* et *Mesures définies par l'utilisateur*.

Base de données : PROD.gilles.fr > Centre de conseil Connecté en tant que SYSTEM

Centre de conseil Page régénérée 2 juil. 2005 15:57:51 [Régénérer](#)

Fonctions de conseil

ADDM	Fonction de conseil sur la mémoire	Fonction de conseil sur les segments
SQL Tuning Advisor	Fonction de conseil sur la durée moyenne de récupération	Gestion de l'annulation (undo)
SQL Access Advisor		

Tâches de la fonction de conseil [Change Default Expiration](#)

Rechercher 989x623

Sélectionnez un type de conseil et entrez éventuellement un nom de tâche pour filtrer les données affichées dans l'ensemble de résultats (ResultSet).

Type de conseil	Nom de tâche	Exécutions de la fonction de conseil
Tous les types		989x623
		Dernière exécution

[Exécuter](#)

Résultats

[Visualiser le résultat](#) [Supprimer](#) Actions [Reprogrammer](#) [Exécuter](#)

Sélectionner	Type de conseil	Nom	Description	Utilisateur	Statut	Heure de début	Heure de fin	Expire dans (jours)
☒	ADDM	ADDM:4262180207_1_151	ADDM auto run: snapshots [150, 151], instance 1, database id 4262180207	SYS	TERMINE	26 juin 2005 23:00:42	26 juin 2005 23:00:42	24
☐	SQL Tuning Advisor	SQL_TUNING_1118868248731		SYSTEM	TERMINE	15 juin 2005 22:44:24	15 juin 2005 22:44:29	13

Figure 28-8

SQL Tuning Advisor

Le plan d'exécution des ordres SQL avec SQL*Plus

SQL*Plus est un outil interactif, qui permet de lancer des commandes SQL et PL/SQL (le langage de développement procédural d'Oracle). Cet outil existe sur toutes les machines où Oracle a été porté.

Bien que vieillissant et offrant moins de souplesse que l'utilitaire SQL*Plus Worksheet d'Oracle Enterprise Manager, SQL*Plus dispose d'une option très intéressante : vous pouvez visualiser le plan d'exécution d'une requête SQL en interactif. Ce sont les mêmes statistiques que nous pouvons observer dans Oracle Enterprise Manager.

Auparavant, ceci se faisait par l'utilitaire TKPROF, qui existe toujours, mais qui est d'un usage plus contraignant.

D'autres fonctionnalités permettent d'optimiser la base de données (cf. ADDM et AWR abordés dans ce chapitre) mais seule cette démarche permet d'optimiser et de réduire la consommation d'un ordre SQL.

Préparation de l'environnement

Préalablement, un ensemble d'opérations doit être mené sur chaque base Oracle 10g pour laquelle vous souhaitez effectuer des plans d'exécution.

Connectez-vous sous l'utilisateur SYS, lancez le fichier suivant qui crée le rôle PLUS-TRACE :

```
SQL> start c:\oracle\ora10\sqlplus\admin\plustrce.sql
```

Toujours sous le même utilisateur SYS, affectez le rôle aux utilisateurs qui analyseront les plans d'exécution des requêtes. Par exemple, pour l'utilisateur Scott :

```
SQL> grant PLUSTRACE to scott ;
```

Enfin, créez la table PLAN_TABLE qui sert à stocker temporairement le plan d'exécution des requêtes. Pour cela, utilisez le fichier *utlxplan.sql* livré avec la version Oracle 10g.

```
SQL> connect scott
SQL> start C:\oracle\product\10.1.0\db_1\RDBMS\ADMIN\utlxplan.sql
```

Utilisation d'AUTOTRACE avec SQL*Plus

L'utilisation d'AUTOTRACE est ensuite très simple. Ses options sont :

AUTOTRACE OFF | ON | TRACEONLY [EXPLAIN] [STATISTICS]

L'exemple suivant met en œuvre l'utilisation de l'autotrace :

```
SQL> spool fichier_spool.lis
SQL> connect scott
SQL> set AUTOTRACE TRACEONLY EXPLAIN STATISTICS
SQL> select ename, job, dname from emp, dept
      where emp.deptno = dept.deptno order by sal;

14 ligne(s) sélectionnée(s).

Plan d'exécution
-----
  0      SELECT STATEMENT Optimizer=CHOOSE
  1   0      SORT (ORDER BY)
  2   1      NESTED LOOPS
  3   2          TABLE ACCESS (FULL) OF 'EMP'
  4   2          TABLE ACCESS (BY INDEX ROWID) OF 'DEPT'
  5   4              INDEX (UNIQUE SCAN) OF 'PK_DEPT' (UNIQUE)

Statistiques
-----
      349 recursive calls
         5 db block gets
      94 consistent gets
      22 physical reads
         0 redo size
    1169 bytes sent via SQL*Net to client
     709 bytes received via SQL*Net from client
         4 SQL*Net roundtrips to/from client
         8 sorts (memory)
         0 sorts (disk)
      14 rows processed
SQL>
SQL> spool off
```

Tous les renseignements sur la façon dont Oracle 10g a traité l'ordre SQL figurent dans ces statistiques.

Nous retrouvons ces mêmes statistiques dans Oracle Enterprise Manager. En cas d'utilisation d'un « spool » de fichier avec l'option AUTOTRACE de SQL*Plus, vous pouvez imprimer le plan d'exécution de la requête, action plus complexe à partir de la console d'OEM.

La lecture du plan d'exécution s'effectue de bas à haut. Nous ne précisons pas dans ce livre l'interprétation d'un plan d'exécution ainsi que les statistiques, mais ce sont des points importants du processus d'amélioration des performances des ordres SQL.

Cette méthode vous permet de visualiser l'influence des statistiques d'Oracle 10g sur la résolution des ordres SQL : effectuez l'analyse d'une requête, mettez à jour les statistiques des tables et des index intervenant dans cette requête, relancez-la.



Figure 28-9

Visualisation du plan d'exécution sous OEM

Étape 4 : tuning de la mémoire

Dans tout système informatique, l'utilisation de mémoire est synonyme de performance. Les données accédées et manipulées en mémoire le sont beaucoup plus rapidement que sur disque. C'est pourquoi Oracle 10g, en liaison avec des processus efficaces, a recours le plus possible à la mémoire physique des ordinateurs.

La zone mémoire qu'Oracle 10g utilise pour résoudre les requêtes SQL est paramétrable. Toute l'astuce consiste à lui allouer autant de mémoire que possible, sans solliciter la zone de pagination qui intervient lorsque la capacité mémoire est dépassée et force Windows à pager.

Il faudra donc ajuster les paramètres en observant constamment l'évolution globale du système, au moyen de commandes Windows comme `pfmon.exe` ou d'utilitaires graphiques comme le gestionnaire de tâches ou l'analyseur de performances.

N'utilisez que les options nécessaires d'Oracle

Oracle propose d'installer dans chaque base de nombreuses options, dont le JServer, très consommateur en ressources mémoire et CPU. Nous vous indiquons où se situent ces options au chapitre 14, *Création d'une base Oracle 10g*. Ne les installez que si elles vous sont nécessaires.

La pagination

La *pagination* est une technique largement utilisée sous Windows pour pallier le manque de mémoire. Lorsque les processus ont besoin d'une quantité de mémoire supérieure à la mémoire physique, le système a recours à une zone de débordement disque, la *pagination* ou *page fault*. Dans ce cas, des pages du processus « dormant » sont basculées sur la zone de pagination, ce qui libère de la mémoire pour d'autres pages.

La pagination intervient lorsque la mémoire est chargée. Elle libère alors des ressources au prix d'une surcharge des entrées/sorties disque : c'est son inconvénient. Un recours trop important à la pagination dégrade fortement les performances du système.

Face à la pagination, plusieurs attitudes sont possibles : si c'est un effet occasionnel, ce n'est pas grave. Si c'est un état permanent, il convient soit d'augmenter la mémoire physique de la machine, soit de diminuer la consommation mémoire. Dans ce cas, il faut réduire la zone mémoire allouée à chaque instance Oracle 10g ou supprimer des processus inutiles présents sur le système.

Les gains de performance prévisibles, conséquences de la suppression de la pagination du système, vont de 0 à 100 %.

La définition des fichiers de pagination a été abordée au début de ce chapitre.

N'oubliez pas d'augmenter la zone de pagination Windows si vous ajoutez de la mémoire sur le système.

Surveillance de la pagination Windows

Le gestionnaire de tâche ou l'analyseur de performances sont adaptés à la surveillance de la pagination du système. Ils peuvent même surveiller la pagination effectuée par un processus : c'est utile pour surveiller le processus oracle.exe.

L'optimisation de la mémoire réservée pour Oracle 10g

Les paramètres d'allocation mémoire pour Oracle 10g sont modifiables dynamiquement. Lors du démarrage d'une instance, ils sont lus dans un fichier d'initialisation texte ou persistant. La commande suivante (que l'on retrouve dans les commandes d'audit en annexe), permet de connaître la taille mémoire allouée à l'instance :

```
SQL> connect system
SQL> select * from v$sga ;
```

Les paramètres influant sur les performances sont abordés au chapitre 8, *L'utilisation de la mémoire par Oracle 10g*. Pour mémoire, les paramètres les plus importants, influant sur la zone mémoire réservée par chaque instance Oracle 10g, sont les suivants :

SGA_TARGET (nouveau 10g)

Le paramètre SGA_TARGET spécifie la taille totale devant être allouée à la SGA. Si sa valeur est à 0, le tuning automatique des zones de la SGA est désactivé. Si une valeur lui est affectée, la taille des zones mémoire suivantes est automatiquement adaptée pour se répartir dynamiquement la taille définie par SGA_TARGET :

- DB_CACHE_SIZE (buffer cache)
- SHARED_POOL_SIZE (Shared Pool)
- LARGE_POOL_SIZE (Large pool)
- JAVA_POOL_SIZE (Java Pool)
- SGA_TARGET est modifiable uniquement par la commande ALTER SYSTEM.

DB_CACHE_SIZE

Ce paramètre définit la zone mémoire utilisée pour conserver les données des tables et des index accédés. Plus cette zone est grande, plus petite est la probabilité d'accéder au disque à la suite d'un ordre SELECT. C'est lui qui a le plus d'impact sur la taille totale de la SGA.

SHARED_POOL_SIZE

Cette zone mémoire est utilisée pour conserver les plans d'exécution des requêtes ainsi que les procédures PL/SQL traitées par le serveur. Elle peut être diminuée si vous sollicitez peu le PL/SQL, car c'est lui qui occupe le plus d'espace.

JAVA_POOL_SIZE

Cette zone mémoire est utilisée pour exécuter les applications Java au cœur de la base Oracle. Si vous n'utilisez pas cette option, ne lui allouez aucune mémoire.

DB_BLOCK_SIZE

Ce paramètre a été largement commenté dans le chapitre précédent.

PGA_AGGREGATE_TARGET (nouveau 10g)

La gestion automatique de la taille des zones mémoire de PGA est maintenant une option par défaut. Il est toutefois possible de le désactiver pour revenir à l'ancien mode en plaçant sa valeur à 0. La valeur par défaut de PGA_AGGREGATE_TARGET est de 20 % de la taille de la SGA. Lorsque vous l'utilisez, les paramètres *_AREA_SIZE (dont le suivant) sont inactifs.

SORT_AREA_SIZE

Lorsque le volume des données à trier est faible, Oracle 10g réalise l'opération dans une zone mémoire allouée au processus qui a demandé le tri. S'il s'agit d'un volume important,

cette zone peut s'avérer trop petite, auquel cas Oracle 10g utilise le segment temporaire sur disque. Ceci génère plus d'entrées/sorties disque et ralentit le tri.

LOG_BUFFER

Alloue une zone qui conserve les données à insérer dans les fichiers redo-log. Augmenter cette zone permet de réduire les entrées/sorties des fichiers log.

Étape 5 : tuning des entrées/sorties disque

Le chapitre précédent traitait des configurations disque. Cette technologie évolue moins vite et de façon moins médiatisée que celle des processeurs. Nous vous recommandons de prendre le plus grand soin des disques, qui représentent une partie significative des performances globales du système.

Il est facile de prévoir une configuration homogène des disques alors qu'il est difficile et contraignant de modifier une configuration existante.

Nous vous conseillons de vous reporter au chapitre précédent pour choisir le type de partition RAID et le meilleur placement possible des fichiers.

Le temps de lecture moyen d'un bloc de données se situe généralement entre 5 et 20 millisecondes suivant le système et la charge. Si la durée est plus longue, vous êtes en présence d'une saturation disque. Deux options :

- diminuez la charge en travaillant sur la base, les applications, etc. ;
- revoyez la configuration des disques et la répartition des données sur différents axes.

Étape 6 : tuning de l'utilisation CPU

Il y a peu de possibilités pour optimiser la puissance processeur : le changer pour en utiliser un autre plus puissant, en augmenter le nombre si la carte mère l'accepte ou tout simplement diminuer les ressources consommées, c'est-à-dire la charge de la machine.

Fermer toutes les applications inutiles

Suite à une installation de base, de très nombreux processus Windows sont lancés au démarrage de la machine : fermez toutes les applications inutiles qui fonctionnent sur le serveur.

Les applications démarrées au lancement de la machine

Certaines applications démarrées au lancement de la machine peuvent être inutilement encombrantes. Faites le tri et ne conservez que celles qui demeurent nécessaires.

Les économiseurs d'écran

On les oublie souvent, mais ce sont de véritables catastrophes en terme de consommation processeur et mémoire. S'ils travaillent sur une station autonome, pourquoi pas. Mais dans le cadre d'un serveur base de données, ne laissez pas un pourcentage important des ressources ainsi gaspillé. Choisissez l'économiseur le plus simple possible, en évitant les « Flying Toasters » et autres fractales.

Étape 7 : tuning des différentes contentions

Oracle 10g propose de nouveaux outils pour réaliser un tuning fin de la base de données. Le plus récent, basé sur Automatic Workload Repository (AWR), est l'Automatic Database Diagnostic Monitor (ADDM), les plus anciens sont tkprof et utlstat/utlestat.

Pour vous proposer des pistes d'optimisation, l'Automatic Database Diagnostic Monitor (ADDM) exploite les statistiques de tous ordres collectés par l'Automatic Workload Repository (AWR). Comme AWR conserve l'historique des événements, ADDM peut être utilisé *a posteriori* pour éviter qu'une baisse de performance constatée ne se reproduise.

AWR et ADDM évaluent la charge et le comportement général de la base de données. Ils ne sont donc pas adaptés à l'optimisation d'un ordre SQL identifié. Ce sera le rôle de tkprof.

Automatic Workload Repository (AWR)

AWR collecte des informations à une fréquence déterminée. Par défaut, la collecte s'effectue une fois par heure et la période de rétention des informations collectées est de 7 jours.

Le paramètre d'initialisation STATISTICS_LEVEL doit être placé à TYPICAL ou ALL. S'il correspond à la valeur BASIC, vous pouvez enregistrer manuellement un « cliché de performances » mais aucun cliché ne sera enregistré automatiquement.

Les statistiques collectées lors de chaque cliché concernent l'accès aux objets, l'activité et la durée des sessions, les ordres SQL les plus consommateurs de ressources (CPU, mémoire disque).

Ces informations nécessitent de l'espace de stockage. Avec une moyenne de 10 sessions concurrentes, une mesure par heure sur une période de 7 jours (durée par défaut de la rétention), l'espace consommé se situe entre 200 et 300 Mo. La réduction de la fréquence de mesure ou de la durée de rétention diminuent la finesse de l'échantillonnage et peuvent affecter les recommandations faites avec les outils ADDM, SQL Tuning Advisor, Undo Advisor, Segment Advisor.

Durée de rétention conseillée : si le cycle de fonctionnement de l'application est hebdomadaire avec des traitements chaque fin semaine, une période de rétention d'une semaine est conseillée. Si le cycle est mensuel, avec des traitements importants répartis sur tout le mois, la période de rétention devra être adaptée à cette fréquence.

Vous pouvez paramétrer AWR à partir d'Oracle Enterprise Manager :

1. connectez-vous à Oracle Enterprise Manager ;
2. sélectionnez l'onglet *Administration* ;
3. sélectionnez les options de *Charge globale*.

Vous pouvez aussi paramétrer AWR en ligne :

```
begin
dbms_workload_repository.modify_snapshot_settings (retention=> 10080, interval => 30 )
end;
/
```

Automatic Database Diagnostic Monitor (ADDM)

L'analyse d'ADDM est exécutée chaque fois qu'un cliché de performances est pris par AWR. Il convient donc que les clichés soient fréquents pour bénéficier d'un échantillonnage correct mais pas trop de façon à ce qu'AWR et ADDM ne viennent pas perturber les performances de la base.

ADDM permet :

- d'analyser la base heure par heure ;
- de bénéficier du croisement d'une multitude d'éléments mesurés ;
- de bénéficier d'années d'expériences d'Oracle en ce domaine ;
- d'identifier la cause d'une baisse de performance et pas uniquement les symptômes ;
- de profiter des recommandations d'Oracle pour améliorer les performances ;
- d'appliquer immédiatement ces recommandations sur la base.

La puissance et la simplicité d'ADDM font qu'il peut être utilisé sur tout type de base : en production, en test, en développement. Cette simplicité permet d'effectuer un tuning en « boucle » : mesure de performances par AWR, recommandations par ADDM, mise en place de ces recommandations et ainsi de suite.

ADDM peut être lancé manuellement ou graphiquement via Oracle Enterprise Manager. Le plus simple est d'utiliser OEM :

1. connectez-vous à Oracle Enterprise Manager ;
2. en bas de la page d'accueil, dans la rubrique *Liens associés*, sélectionnez *Centre de conseil*. Vous accédez à l'écran suivant.

Base de données : PROD.gilles.fr > Centre de conseil Connecté en tant que SYSTEM

Centre de conseil Page régénérée 2 juil. 2005 15:57:51 [Régénérer](#)

Fonctions de conseil

ADDM	Fonction de conseil sur la mémoire	Fonction de conseil sur les segments
SQL Tuning Advisor	Fonction de conseil sur la durée moyenne de récupération	Gestion de l'annulation (undo)
SQL Access Advisor		

Tâches de la fonction de conseil [Change Default Expiration](#)

Rechercher 989x623

Sélectionnez un type de conseil et entrez éventuellement un nom de tâche pour filtrer les données affichées dans l'ensemble de résultats (ResultSet).

Type de conseil : Tous les types Nom de tâche : Exécutions de la fonction de conseil : Dernière exécution [Exécuter](#)

Résultats [Visualiser le résultat](#) [Supprimer](#) Actions [Reprogrammer](#) [Exécuter](#)

Sélectionner	Type de conseil	Nom	Description	Utilisateur	Statut	Heure de début	Heure de fin	Expire dans (jours)
☒	ADDM	ADDM:4262180207_1_151	ADDM auto run: snapshots [150, 151], instance 1, database id 4262180207	SYS	TERMINE	26 juin 2005 23:00:42	26 juin 2005 23:00:42	24
☐	SQL Tuning Advisor	SQL_TUNING_1118868248731		SYSTEM	TERMINE	15 juin 2005 22:44:24	15 juin 2005 22:44:29	13

Figure 28-10

ADDM avec Oracle Enterprise Manager

À partir de cet écran, vous précisez les périodes de prise de mesures et vous visualisez l'ensemble des propositions émises par ADDM.

Vous pouvez aussi accéder à ces analyses en lançant le script \$ORACLE_HOME/rdbms/admin/addmrpt.sql ou le package DBMS_ADVISOR.

L'analyse d'ADDM vise à réduire le temps nécessaire aux traitements en incluant le temps CPU consommé en temps d'attente. Les problèmes traités par ADDM sont triés par temps de traitement décroissants.

En réduisant le temps de traitement des requêtes les plus consommatrices, la base de données est capable de traiter plus de requêtes sans augmenter la charge.

Les problèmes, symptômes et informations signalés par ADDM concernent principalement :

- le manque de ressources processeur ;
- un dimensionnement inadapté des structures mémoire d'Oracle (SGA, PGA...) ;
- une surcharge sur les entrées/sorties disque ;

- l'identification d'ordres SQL consommateurs ;
- des recommandations d'optimisation de l'application ;
- optimisation du paramétrage d'Oracle ;
- identification de problèmes.

Les recommandations peuvent être appliquées directement sur le système. C'est d'une efficacité redoutable !

Comme ADDM s'appuie sur les informations collectées par AWR, il faut que le rythme de la collecte soit adapté au cycle de vie de la base. Par exemple, pour bénéficier de données plus rapprochées, vous pouvez modifier temporairement la collecte de données par AWR pour bénéficier de plus de recommandations d'ADDM.

Pour éviter de fausser l'analyse d'ADDM :

- elle doit survenir entre deux mesures AWR, la dernière n'ayant pas rencontré d'erreur ;
- les clichés de mesure ne doivent pas avoir été purgés ;
- la base de données ne doit pas être arrêtée entre les deux clichés.

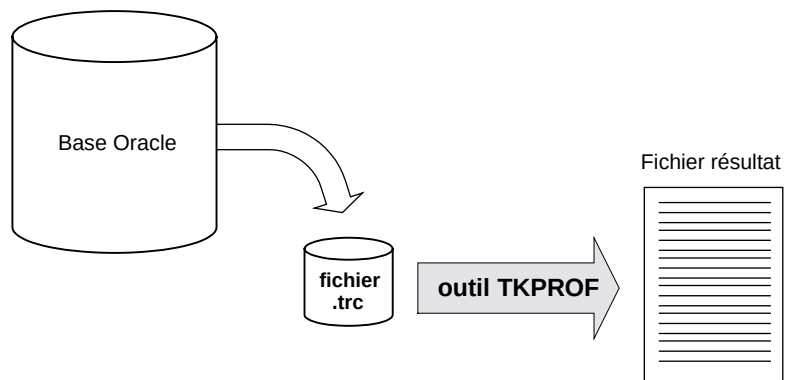
Le SQL Access Advisor

Le SQL Access Advisor d'Oracle Enterprise Manager permet d'obtenir des recommandations concernant les index, les vues, les snapshots pour améliorer les performances des requêtes.

L'outil tkprof

Figure 28-11

Principe de l'outil tkprof



Le principe de *tkprof* consiste à forcer la base de données à créer un fichier trace, puis à exploiter le contenu de ce fichier pour y accéder de façon lisible.

L'avantage principal de *tkprof* est de ne pas être intrusif : vous pouvez ainsi étudier une session utilisateur, récupérer l'ensemble des ordres SQL transmis à la base de données, visualiser leur plan d'exécution, etc., sans disposer des sources de cette application. Vous pouvez ainsi visualiser les ordres SQL défaillants pour agir sur la base (ajout d'index, réorganisation des données...) ou faire pression sur un prestataire pour qu'il améliore son application.

La création du fichier trace s'effectue soit pour une session (cas de déclenchement à partir de SQL*Plus), soit pour toutes les sessions de la base de données. Dans ce cas, contrôlez le volume des fichiers trace créés. La démarche la plus simple :

1. Modifiez le fichier d'initialisation de la base :

```
# ajoutez les lignes
sql_trace = true
timed_statistics = true
```

2. Démarrez la base

3. Supprimez toutes les traces créées (pour identifier le fichier trace qui vous intéresse)

```
del c:\oracle\admin\TEST\udump\*.trc
```

4. Lancez l'application à étudier ;

5. Récupérez le fichier trace créé en c:\oracle\admin\TEST\udump. (ou le répertoire précisé par la variable d'initialisation *user_dump_dest* du fichier *initSID.ora*) ;

6. Rétablissez la base en mode « normal » ;

```
# sql_trace = true
# timed_statistics = true
```

7. Arrêtez et relancez la base ;

8. Créez la table PLAN_TABLE utilisée par *tkprof* :

```
# créez la table Plan Table (utilisée par tkprof)
sqlplus scott/tiger @C:\oracle\product\10.1.0\db_1\RDBMS\ADMIN\utlxplan.sql
```

9. Étudiez le fichier trace :

```
tkprof <fichier entrée> <fichier produit> explain=<user>/<pwd> sys=no
```

Dans notre exemple :

```
tkprof ora_892.trc ora_892.txt explain=scott/tiger sys=no
```

Le fichier résultat produit est :

```
TKPROF: Release 10.1 - Production on Web Sept 7 22:01:51 2001
```

```
(c) Copyright 2004 Oracle Corporation. All rights reserved.
```

```
Trace file: ora_892.trc
Sort options: default
```

```

*****
count    = number of times OCI procedure was executed
cpu      = cpu time in seconds executing
elapsed  = elapsed time in seconds executing
disk     = number of physical reads of buffers from disk
query    = number of buffers gotten for consistent read
current  = number of buffers gotten in current mode (usually for update)
rows     = number of rows processed by the fetch or execute call
*****

```

```

*****
Des lignes ont été supprimées pour plus de lisibilité
*****

```

```
select * from emp
```

call	count	cpu	elapsed	disk	query	current	rows
Parse	1	0.00	0.00	0	0	0	0
Execute	1	0.00	0.00	0	0	0	0
Fetch	2	0.00	0.00	0	2	4	14
total	4	0.00	0.00	0	2	4	14

```

Misses in library cache during parse: 0
Optimizer goal: CHOOSE
Parsing user id: 20 (SCOTT)

```

```

Rows      Row Source Operation
-----
14  TABLE ACCESS FULL EMP

```

```

Rows      Execution Plan
-----
0  SELECT STATEMENT  GOAL: CHOOSE
14  TABLE ACCESS (FULL) OF 'EMP'

```

```
*****
```

```
select ename, job, dname from emp, dept
where emp.deptno=dept.deptno
```

call	count	cpu	elapsed	disk	query	current	rows
Parse	1	0.01	0.00	0	0	0	0
Execute	1	0.00	0.00	0	0	0	0
Fetch	2	0.00	0.00	0	18	4	14
total	4	0.01	0.00	0	18	4	14

Misses in library cache during parse: 1

Optimizer goal: CHOOSE

Parsing user id: 20 (SCOTT)

Rows	Row Source Operation
14	NESTED LOOPS
15	TABLE ACCESS FULL EMP
14	TABLE ACCESS BY INDEX ROWID DEPT
28	INDEX UNIQUE SCAN (object id 2841)

Rows	Execution Plan
0	SELECT STATEMENT GOAL: CHOOSE
14	NESTED LOOPS
15	TABLE ACCESS (FULL) OF 'EMP'
14	TABLE ACCESS (BY INDEX ROWID) OF 'DEPT'
28	INDEX (UNIQUE SCAN) OF 'PK_DEPT' (UNIQUE)

OVERALL TOTALS FOR ALL NON-RECURSIVE STATEMENTS

call	count	cpu	elapsed	disk	query	current	rows
Parse	9	0.01	0.00	0	0	0	0
Execute	9	0.00	0.00	0	0	0	4
Fetch	7	0.00	0.00	0	22	16	30
total	25	0.01	0.00	0	22	16	34

Misses in library cache during parse: 1

OVERALL TOTALS FOR ALL RECURSIVE STATEMENTS

call	count	cpu	elapsed	disk	query	current	rows
Parse	2	0.00	0.00	0	0	0	0
Execute	2	0.00	0.01	0	0	0	0
Fetch	3	0.00	0.00	0	12	0	2
total	7	0.00	0.01	0	12	0	2

Misses in library cache during parse: 0

- 9 user SQL statements in session.
- 2 internal SQL statements in session.
- 11 SQL statements in session.
- 4 statements EXPLAINed in this session.

```
*****
Trace file: ora_892.trc
Trace file compatibility: 7.03.02
Sort options: default

   1 session in tracefile.
   9 user SQL statements in trace file.
   2 internal SQL statements in trace file.
  11 SQL statements in trace file.
   9 unique SQL statements in trace file.
   4 SQL statements EXPLAINED using schema:
      SCOTT.prof$plan_table
      Default table was used.
      Table was created.
      Table was dropped.
  121 lines in trace file.
```

Le fichier résultat montre les ordres SQL traités par la base au cours de cette session utilisateur. Pour chacun de ces ordres, des statistiques et le plan d'exécution sont produits. Le fichier produit se termine par des statistiques globales sur l'ensemble des ordres SQL de la session.

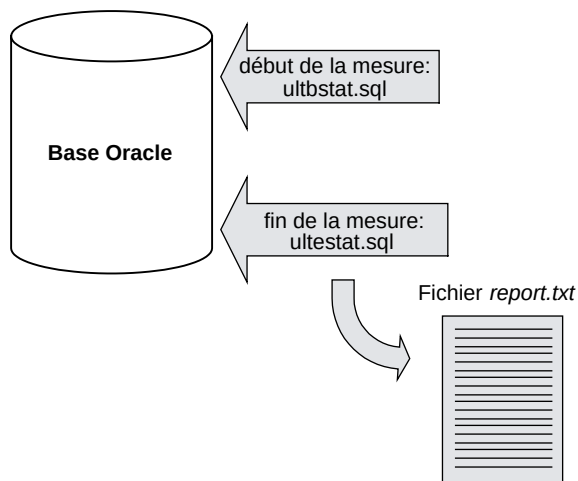
Face à une application, même si vous ne disposez pas de ces sources, vous pouvez connaître grâce à *tkprof* tous les ordres SQL traités par Oracle.

Les scripts *utlbstat*/*utlstat*

Le principe de ces scripts est d'effectuer une collecte d'informations pendant une période d'activité. Le début de la mesure correspond à l'exécution du script *utlbstat.sql* et l'arrêt à *utlstat.sql*. Ce dernier produit le rapport d'exécution à exploiter.

Figure 28-12

Principe des scripts
utlbstat/*utlstat*



Le rapport généré est très volumineux, et ne doit pas se lire de façon linéaire, mais vous devez vous y déplacer pour y rechercher les informations importantes.

Pour chaque rubrique, un commentaire indique l'élément mesuré, l'ordre SQL exécuté puis le résultat obtenu. Par exemple, pour étudier les entrées sorties disques survenues durant la période d'étude, le résultat obtenu est :

```
I/O should be spread evenly accross drives. A big difference between
phys_reads and phys_blks_rd implies table scans are going on.
```

```
select table_space, file_name,
phys_reads reads, phys_blks_rd blks_read, phys_rd_time read_time,
phys_writes writes, phys_blks_wr blks_wrt, phys_wrt_tim write_time,
megabytes_size megabytes
from stats$files order by table_space, file_name;
```

TABLE_SPACE	FILE_NAME	READS	BLKS_READ	R_TIME	WRI.	BLKS_WRT	WRITE_TIME	MB
INDX	c:\oradata\ORCL\indx01.dbf	4	4	0	4	4	0	10
RBS	c:\oradata\ORCL\rbs01.dbf	4	4	0	4	4	0	27
SYSTEM	c:\oradata\ORCL\system01.dbf	238	238	22	826	826	474	184
TEMP	c:\oradata\ORCL\temp01.dbf	12	76	0	76	76	0	10
USERS	c:\oradata\ORCL\users01.dbf	4	4	0	4	4	0	10

5 lignes sélectionnées

Les scripts SQL *utlbstat*/*utlestat* constituent un moyen très puissant d'étude du comportement global de la base de données. Le revers de cette précision est l'investissement temps qu'il faut y consacrer.

Les outils complémentaires

Des outils complémentaires existent sur le marché. Ils évoluent rapidement et je vous conseille une recherche sur www.google.com avec *tuning oracle* comme mot-clé. Pour mémoire, je mentionnerai deux outils qui proposent une version d'évaluation : Toad www.toadsoft.com et Tora www.globecom.se/tora. Leur orientation résolument efficace en fait des outils à ne pas négliger. Vous pouvez aussi consulter www.dbcool.com. La liste de ces outils évolue fréquemment. Utilisez des moteurs de recherche comme www.google.com pour les identifier.

Résumé

Ce chapitre a abordé l'optimisation d'une base de données Oracle 10g sous Windows en vous proposant une démarche structurée :

- comment optimiser son serveur Windows ;
- des principes généraux concernant les spécificités du tuning et ce que l'on peut en attendre ;

- les étapes pour effectuer le tuning d'une base Oracle 10g ;
- l'étude détaillée de chacune de ces étapes.

Le tuning est un processus itératif sur lequel il est obligatoire de revenir régulièrement, car le contenu des bases de données ainsi que l'environnement Windows évoluent. Encore une fois, il n'y a rien de magique à attendre si l'application ou le schéma de la base de données sont déficients : il faut les corriger en premier.

Comme vous avez pu le constater, ce chapitre ne suffit pas, à lui seul, pour assurer de bonnes performances : il est dans la lignée de toutes les recommandations faites dans les chapitres précédents.

Partie VI

Annexes

Annexe 1 : Glossaire des termes utilisés

Annexe 2 : Mémento SQL

Annexe 3 : Procédures pour le DBA

Annexe 4 : Extensions des fichiers d'Oracle

Annexe 5 : NLS : support des langues par Oracle 10g et l'euro

Annexe 6 : Les logiciels livrés

Annexe 7 : Sites Internet et bibliographie

Annexe 8 : Installation d'Oracle 10g XE

Glossaire des termes utilisés

Accident d'instance	Un accident d'instance survient lorsque la base de données est brutalement empêchée de fonctionner, par exemple en cas d'arrêt de Linux, alors que la base de données fonctionne. L'instance n'est pas fermée « proprement » et une restauration est nécessaire au démarrage suivant.
Active Directory	Annuaire compatible LDAP intégré aux serveurs Microsoft. Cette brique de base est utilisée pour gérer les utilisateurs, leur authentification, les droits d'accès, etc. Oracle utilise Active Directory pour gérer les droits d'accès aux bases.
Alias	Un alias regroupe sous un nom facile à comprendre et à manipuler une chaîne de connexion qui permet à Oracle Net de se connecter à une base de données cible.
Apache	Serveur Web le plus utilisé dans le monde. Issu du logiciel « libre », il est intégré dans l'offre commerciale de nombreux éditeurs comme IBM et Oracle.
Applet	Terme propre au langage Java pour désigner un programme qui peut être importé dynamiquement dans des pages Web ou des applications, suivant les besoins.
Appliance	Selon le cabinet d'études Merrill Lynch, les <i>appliances</i> sont des équipements informatiques dédiés à la réalisation de fonctions précises. Ils concentrent dans une « boîte noire » une seule fonction. Ils sont simples, peu coûteux, par opposition au PC, qui se veut universel et généraliste.
Archivage	Option facultative qui permet d'archiver tout fichier redo-log saturé. Les fichiers redo-log archivés peuvent servir à restaurer la base de données, à la suite d'un crash disque par exemple.
ASP	Active Server Pages. Technologie d'origine Microsoft destinée à fonctionner en liaison avec le serveur Web IIS pour construire des pages Web dynamiques. Ces pages HTML contiennent un ou plusieurs scripts qui sont exécutés dynamiquement sur le serveur Web, avant que la page soit envoyée au navigateur, c'est-à-dire à l'utilisateur.
Audit	Ordre SQL qui permet de tracer tout type d'activité sur la base de données (manipulation des tables, connexions à la base...).
Authentification	Consiste à vérifier l'identité d'un utilisateur, d'un équipement ou d'un programme avant qu'il accède à des ressources.

Autorisation	Droit d'accès à une table, qui s'octroie par l'ordre SQL GRANT.
Base de données	Une base de données peut indiquer un logiciel servant à créer et administrer des bases de données. Dans ce cas, Oracle, SQL Server, DB2 sont des logiciels de bases de données. Pour Oracle, une base de données fait référence aux fichiers physiques qui constituent chaque base. Une base de données « en marche » se nomme alors une instance.
Base de registre	Lieu central où sont conservées toutes les informations touchant la configuration du système Microsoft comme la séquence d'initialisation, la configuration matérielle, les stratégies de sécurité, les options propres à un utilisateur particulier, le lancement d'applications ou de services.
Bloc Oracle	La plus petite unité entrée/sortie entre la SGA et les fichiers de la base. Un fichier base est composé de n blocs Oracle. La taille d'un bloc Oracle varie et peut être de 2 ko, 8 ko, 16 ko ou 32 ko. Elle est figée à la création de chaque tablespace. C'est un élément majeur du tuning d'une base.
Browser Web, Navigateur (en français)	Un navigateur est un programme qui permet aux utilisateurs de lire les documents HTML et se déplacer sur intranet ou Internet. Les plus connus sont Navigator de Netscape et Internet Explorer de Microsoft.
CGI (Common Gateway Interface)	Technique largement répandue dans l'industrie pour exécuter des applications sur un serveur Web. Alors que les pages HTML standards sont « statiques » (le même texte est retrouvé à chaque fois), CGI permet d'exécuter n'importe quel programme pour générer des documents Web HTML dynamiques.
Chaîne de connexion	Voir Oracle Net.
Checkpoint	Littéralement « point de contrôle ». C'est un événement qui se déclenche lorsqu'un fichier redo-log est saturé ou lorsque le paramètre LOG_CHECKPOINT_INTERVAL est atteint. Son déclenchement implique l'écriture des buffers modifiés de la SGA vers les fichiers de la base.
Client léger (thin client)	Par opposition aux clients lourds (fat clients), les clients légers sont des terminaux minimalistes capables de se connecter à un large éventail de systèmes informatiques, au moyen d'une émulation appropriée. Pour les mainframes, on utilisera un émulateur au protocole type 3270 ou encore VT220 pour les serveurs Unix. Pour les serveurs Windows, les protocoles RDP (Microsoft) ou ICA (Citrix) ont pour but de rendre les serveurs Microsoft multi tâche et multi-utilisateur. Tous les programmes s'exécutent alors sur le serveur, seul leur affichage est déporté. Le client Citrix existe pour différents Unix, dont Linux.
Client-serveur	Architecture matérielle et logicielle fondée sur la séparation des traitements. Le CPU du client lance des requêtes auprès du CPU serveur qui lui répond. Contrairement au « client léger », on parle dans ce cas de client « lourd » ou « fat client ».
Cluster (de serveurs)	Mise en commun des ressources de plusieurs machines pour augmenter la puissance totale et disposer d'une plus grande fiabilité. La base de données Oracle fonctionne sur ce type d'architecture.
Cluster (de tables)	Technique de stockage physique des tables qui consiste à réunir les tables les plus souvent jointes, en vue de favoriser l'opération de jointure. C'est une technique d'optimisation physique (proche du matériel) qui est très peu utilisée.
COM/COM+	Component Object Model. Standard d'origine Microsoft qui permet à un objet d'interagir avec un autre objet, quel que soit le langage de programmation des objets.
Commit	Ordre SQL de validation des modifications apportées aux données. Le <i>rollback</i> est utilisé pour annuler les données non encore validées.

Concurrence d'accès	La concurrence d'accès survient lorsque plusieurs applications (ou utilisateurs) tentent d'accéder à la même ressource (même zone mémoire, même zone disque, même enregistrement d'une table...) au même instant. Oracle9i gère la concurrence d'accès de manière très performante.
Connexion Oracle	Ce sont le nom, le mot de passe et l'alias (facultatif) qui permettent de se connecter à une base Oracle9i. Ces renseignements sont souvent dissociés de la connexion au système d'exploitation.
CPU	Central Processing Unit ou processeur.
Database link	Lien entre deux bases de données, que l'on crée avec un ordre SQL. Il est alors possible d'accéder aux données distantes en faisant référence au lien. Un database link utilise Oracle Net comme middleware.
DCOM	Distributed Component Object Model : extension du standard COM qui autorise la communication entre objets au travers d'un réseau.
Dictionnaire de données	Ensemble de tables et de vues qui décrivent la base de données (les utilisateurs, leurs privilèges, leurs tables...). Le dictionnaire de données est géré à 100 % par Oracle9i. Les utilisateurs peuvent l'interroger par des ordres SQL.
DLL	Dynamic Link Library : fichier exécutable qu'une application Windows peut charger lorsqu'elle en a le besoin.
Domaine	Un domaine de sécurité est mis en œuvre dans les systèmes Windows de façon hiérarchique.
DTD	Document Type Definition. Ce document (non XML) permet de décrire les structures des documents XML. L'utilisation des DTD est facultative mais elles sont précieuses pour valider un document XML dit « bien formé ». Les DTD peuvent être internes ou externes aux documents XML dont elles expriment la structure. Contrairement aux Schémas XML, les DTD ne renseignent pas le type de données, ce qui signifie qu'une DTD ne suffit pas pour savoir si une donnée doit être numérique ou au format texte.
EJB	Un EJB (Enterprise Java Bean) est un composant logiciel qui respecte les spécifications techniques décrites dans l'architecture J2EE (Java 2 Enterprise Edition).
Email EMAP4	En émission, un e-mail IMAP4 est composé localement puis envoyé au bureau de poste (serveur de messagerie) qui a la responsabilité de le diffuser. En réception, un e-mail EMAP4 ne « bouge pas » du bureau de poste (serveur de messagerie). Le logiciel de messagerie chargé de le lire (client) consulte uniquement les messages du bureau de poste distant. Il est ainsi possible d'accéder à ses messages (nouveaux et anciens) depuis n'importe quel poste connecté au bureau de poste.
Email POP3	En émission, un e-mail POP3 est composé localement puis envoyé au bureau de poste (serveur de messagerie), qui a la responsabilité de le diffuser. En réception, un e-mail POP3 est un courrier transféré depuis le bureau de poste (serveur de messagerie) vers le logiciel chargé de le lire (client). Une fois transféré, le courrier est présent uniquement sur le poste client. Les messages sont consultables uniquement sur le poste qui les a reçus.
ETL	Un outil ETL (Extraction, Transformation and Loading) sert à extraire des données de plusieurs sources (bases de données, fichiers, internet...), à les transformer puis les charger pour alimenter une base décisionnelle. Souvent, ces actions peuvent être réalisées manuellement mais un outil ETL facilite leur enchaînement et leur répétition.
Fichier de contrôle	Fichier décrivant la structure physique de la base de données. Chaque fichier de contrôle contient le nom de la base, les noms et emplacements des fichiers de données et redo-log. Il contient aussi des informations comme la date de création de la base, la séquence de redo-log ou le numéro de checkpoint.

Fichier de données	Fichier contenant toutes les données mais aussi les informations utiles au fonctionnement de la base.
Fichier redo-log	Fichier de journalisation qui enregistre au fil de l'eau toutes les actions modifiant les données. Si un arrêt brutal survient (une panne électrique par exemple) les fichiers redo-log sont utilisés pour écrire dans la base les données validées qui n'avaient pas pu être écrites. C'est l'un des mécanismes les plus puissants d'Oracle.
Fichier trace	Véritable carnet de santé de chacune des instances Oracle. Les différents processus Oracle écrivent automatiquement des informations dans un fichier trace. Sa lecture régulière est source d'informations pour les DBA consciencieux.
FireWall	Littéralement pare-feu. Désigne un ordinateur de sécurité qui contrôle tous les échanges entre un réseau privé et un réseau public, bloquant ainsi les intrus.
FTP	File Transfert Protocol. Protocole permettant de transférer des fichiers de machine à machine.
GUID	Globally Unique Identifier : identifiant unique d'un objet COM conservé dans la base de registre.
Hébergement	Prestation de service Internet. Votre site Web (ou autre service) est hébergé par une société tierce qui assure sa gestion au quotidien et les accès réseau. La prestation de mise à jour et d'évolution de votre site est à dissocier de l'hébergement.
HOMEID	Chaque version Oracle installée utilise une sous-clé de registre comme identifiant. Le HOMEID contient tous les paramètres de configuration pour cette version Oracle. Une nouvelle HOMEID est créée à chaque installation Oracle et son ID est incrémenté : HOME0, HOME1, HOME2, etc.
HOME_NAME	Le HOME_NAME donne un nom significatif à chaque installation Oracle qui crée un HOMEID.
HTML	Hypertext Markup Language. Langage ascii reposant sur des marqueurs utilisé pour spécifier le contenu et les liens hypertexte vers d'autres documents se trouvant sur des serveurs WWW d'Internet. Les utilisateurs visualisent les documents HTML à l'aide de navigateurs Web.
HTTP	Hypertext Transfert Protocol. Protocole de communication utilisé pour transporter l'information entre les navigateurs clients et le serveur WWW qui est accédé.
HyperText	Ensemble de documents contenant des références croisées qui, à l'aide du navigateur Web, permettent aux lecteurs de se déplacer facilement d'un document vers un autre.
IID	Identifiant unique d'une interface COM.
IIS	Internet Information Server. Serveur Web http d'origine Microsoft, concurrent d'Apache.
Index	Technique très utilisée, qui permet d'accéder à une ligne quasi instantanément (au lieu d'un balayage séquentiel complet d'une table). Un index se définit sur une ou plusieurs colonnes d'une table et occupe de l'espace disque. Différentes techniques d'index sont disponibles dans Oracle9i.
Index Bitmap	Technologie de constitution des index, avantageuse par sa rapidité d'accès et son faible volume pour de très grosses tables.
INIT.ORA	Fichier d'initialisation contenant des paramètres mis en œuvre lors du démarrage de l'instance. Il existe souvent un fichier de configuration par instance, appelé <i>initSID.ora</i> , le SID correspondant à l'identifiant de l'instance.
Instance	C'est une base de données Oracle9i « en action », c'est-à-dire lancée et en ordre de marche. Dans ce cas, les programmes nécessaires au fonctionnement d'Oracle9i sont démarrés et une zone de mémoire (la SGA) est réservée au fonctionnement de l'instance. C'est la combinaison des processus et de la mémoire SGA qu'on nomme l'instance.

Instance SID	System Identifier. C'est le nom qui identifie de manière certaine l'instance Oracle9i avec laquelle on travaille.
Internet	Réseau mondial d'ordinateurs qui se fonde sur le protocole réseau TCP/IP.
Intranet	Réseau interne de protocole TCP/IP, dont l'accès est restreint (souvent à l'aide d'un pare-feu) aux personnes d'une société ou d'une organisation. Un intranet propose des services identiques à ceux proposés par Internet, mais il n'est pas nécessaire d'être connecté à Internet.
IO ou I/O	Input/Output. Entrées/sorties disque ou mémoire.
J2EE	J2EE signifie Java 2 Enterprise Edition. C'est un ensemble de standards techniques développés en Java : comment accéder à une base de données, à un annuaire, à une carte à puce, quelles interfaces un composant doit présenter pour être utilisable dans un tel environnement, etc. C'est la combinaison de tous ces services techniques qui constitue l'architecture J2EE. Un logiciel estampillé « J2EE » signifie qu'il respecte et a passé les jeux de test qui valident ces services techniques.
Java	Langage de programmation d'origine Sun largement utilisé dans la communauté Internet.
Javascript	Langage issu de Java utilisé pour personnaliser le fonctionnement des pages HTML en les rendant moins statiques par l'exécution de scripts locaux (contrôle de saisie, affichage, graphiques évolués...).
JavaServer Pages (JSP)	Les JavaServer Pages (JSPs) sont une extension de l'API des Servlet Java. Les JSP étendent les fonctionnalités d'un Servlet en permettant d'incorporer et d'appeler un Servlet Java dans une page HTML.
LDAP	Lightweight Directory Access Protocol. LDAP est le principal standard pour la gestion des annuaires. Active Directory (Microsoft), NDS (Novell), OpenLDAP (« libre ») sont tous basés sur LDAP.
Listener	Ce processus Oracle9i fonctionnant sur un serveur est en attente (en écoute) de toute connexion Oracle Net. C'est lui qui gère la communication côté serveur dans les communications client-serveur. Il permet aussi l'accès à des procédures externes.
LISTENER.ORA	Fichier de configuration du Listener Oracle9i.
Mainframe	Nom donné aux ordinateurs centraux, de forte puissance, qui ne fonctionnent pas sous Unix mais sous des systèmes propriétaires.
MAPI	Messaging Application Programming Interface : interfaces de programmation inter application et inter systèmes. Différentes applications peuvent dialoguer avec différents systèmes d'échange de messages quelle que soit leur plate-forme.
Microsoft .NET	Prononcer « Microsoft dot NET ». Architecture promue par Microsoft comme direction de ses futurs développements.
MMC	Microsoft Management Console. Regroupe dans une console unique l'ensemble des outils d'administration Windows 2000, sous forme de modules logiciels enfichables ou snap-ins. Seule, la MMC ne permet aucune fonctionnalité.
MSCS	Microsoft Clustering Server. Service de cluster Microsoft nécessitant Windows 2000 Advanced Server. Si le système d'exploitation permet de faire fonctionner plusieurs nœuds, il faut encore que les logiciels ou applications tirent le meilleur parti de cette architecture.
MTS	Microsoft Transaction Server. Service transactionnel permettant principalement de réaliser des programmes reliant le serveur Web IIS à la base Microsoft SQL Server.
Mount	Après le démarrage d'une instance, l'ordre SQL « Mount » associe les fichiers de la base de données à l'instance.

NetBEUI	Protocole réseau utilisé par Microsoft.
NetBios	Interface logicielle utilisée par Microsoft et s'appuyant sur NetBEUI. Les logiciels s'interfacent à NetBios au même titre que TCP/IP.
NLS	National Language Support. Paramétrage de la base Oracle9i et des applications clientes pour intégrer les langues autres que l'anglais.
OCI	Oracle Call Interface. C'est un ensemble de routines de bas niveau, que les développeurs utilisent en langage C pour accéder à Oracle9i.
ODBC	Open DataBase Connectivity. Standard de faits qui permet à des applications Windows d'accéder à des bases de données distantes.
OFA	Optimal Flexible Architecture. Méthode d'organisation des fichiers de données et des logiciels Oracle9i sur un système. L'approche OFA, rigoureuse et structurée, simplifie énormément le travail d'un administrateur Oracle9i.
OLE	Object Linking and Embedding. Moyen logiciel permettant de transférer et partager des informations entre des applications Windows. N'existe pas sous Linux.
Operating System	Système d'exploitation. Linux, Unix, Windows NT, Palm OS, Mac OS... sont différents systèmes d'exploitation.
ORACLE_BASE	C'est la racine de toutes les versions Oracle installées sur un système. Chaque <i>ORACLE_HOME</i> s'installe sous une <i>ORACLE_BASE</i> unique.
ORACLE_HOME	À chaque version Oracle correspond un <i>HOMEID</i> et un <i>ORACLE_HOME</i> . Il contient le répertoire d'installation des logiciels Oracle ainsi que toutes les variables définies dans la sous-clé de registre <i>HOMEID</i> .
Oracle COM Automation Feature	Caractéristique d'Oracle permettant aux développeurs PL/SQL d'accéder aux objets COM par l'interface COM Automation.
Oracle Enterprise Manager	Logiciel graphique d'administration des bases Oracle9i.
Oracle /FS (Internet File System)	Oracle Internet File System étend la base de données au rôle de système de fichiers en réseau. Des données structurées ou non-structurées sont conservées dans un dictionnaire qui offre de grandes possibilités de recherche. Les données sont accessibles par la majorité des protocoles d'Internet (HTML, FTP, IMAP4).
Oracle JServer	Oracle Java Server voir Oracle JVM.
Oracle JVM	Oracle intègre une machine virtuelle Java ou JVM (Java Virtual Machine) dans le moteur d'Oracle9i Database Server. Ce moteur Java peut être accédé à partir de très nombreuses interfaces de programmation : procédures et triggers écrits en Java, objets CORBA, Enterprise JavaBeans, Servlets Java, JSP (Java Server Pages).
Oracle Net	Anciennement SQL*Net, c'est le logiciel de communication Oracle9i entre un poste client et un serveur Oracle9i.
Oracle Objects for OLE	Logiciel Oracle respectant les normes OLE de Microsoft qui permet à tout outil « parlant » OLE de communiquer avec le middleware Oracle Net.
Oracle Partitionning	Le partitionnement permet d'agir sur le stockage physique des tables et des index en les éclatant en des éléments plus petits, administrables individuellement. Cette optimisation procure des gains sans avoir à agir sur les applications. Par exemple, pour de très gros volumes de données, il peut être intéressant de grouper les données année par année. Les avantages concernent les performances, les possibilités d'administration et procurent une grande souplesse d'utilisation pour des applications de type DataWarehouse.
Oracle Real Application Clusters	Version d'Oracle fonctionnant sur une architecture matérielle de type cluster.

Oracle Server	Moteur de la base de données Oracle9i.
Pare-feu	Voir FireWall.
PATH	Littéralement chemin d'accès.
PL/SQL	Langage procédural (propriété d'Oracle), complémentaire du SQL. PL/SQL permet, dans les programmes, d'effectuer des boucles (LOOP, WHILE...), poser des conditions (IF, THEN, ELSE...), traiter les erreurs rencontrées. Même si les programmes écrits en PL/SQL ne sont pas stockés dans une base de données, ils sont envoyés d'un bloc au serveur, réduisant ainsi le trafic réseau entre le client et serveur.
PMON (Processus Monitor)	C'est l'un des processus Oracle9i, en charge d'effectuer une restauration automatique de l'instance Oracle9i lors de son démarrage, si l'arrêt précédent n'a pas été « normal ».
Précompilateur	Une « moulinette » Oracle qui permet d'écrire plus facilement des programmes en langage C ou autre. Le programme contenant des ordres SQL est passé au précompilateur, puis au compilateur du langage.
Privilège	Droit d'exécution de tel ou tel type d'ordre SQL. Par exemple, c'est un privilège qui permet à un utilisateur de créer ou non une table.
Procédure	Programme composé de commandes SQL et PL/SQL, exécuté par le serveur Oracle9i. Les procédures et les fonctions sont presque identiques : la fonction accepte des paramètres d'entrée et retourne obligatoirement une valeur, de 0 à « n » valeurs.
Procédure externe	Oracle9i permet à des procédures PL/SQL d'appeler des programmes écrits en langage C externes à la base de données.
Process ou Processus	C'est un programme exécuté sous contrôle du système d'exploitation. Chaque instance d'Oracle9i est gérée par un ensemble de processus, comme PMON (Process Monitor), SMON (System Monitor), etc.
Quota	Limite composite de ressources Oracle9i données à un utilisateur. S'il dépasse son quota, l'opération est annulée.
RAM	Read Access Memory. Mémoire vive (comparée à la mémoire de masse ou disque dur) de l'ordinateur.
Redo-log	Fichier enregistrant séquentiellement toutes les modifications apportées aux données. Les fichiers redo-log sont utilisés en cas de problème si les données n'ont pas eu le temps d'être écrites dans les fichiers de la base. Plusieurs fichiers redo-log sont chaînés circulairement : l'un se remplit, puis l'autre, etc.
Réplication	Possibilité d'effectuer des copies de données vers plusieurs bases, de manière synchronisée ou non.
Restauration	Opération inverse de la sauvegarde, qui consiste à appliquer tout ou partie d'une sauvegarde sur une base de données, pour la restaurer.
Restauration d'instance	Si le processus d'arrêt normal d'une base Oracle9i n'a pas été respecté, une restauration d'instance automatique (et transparente pour l'administrateur) a lieu au démarrage suivant de l'instance. Les transactions validées sont inscrites dans la base de données et celles qui n'ont pas été validées sont annulées.
Rôle	Regroupe sous un même nom un ensemble d'autorisations d'accès à des tables, procédures, fonctions, etc. On affecte à un utilisateur ou à un rôle, un ou plusieurs rôles.
Rollback	« Retour en arrière ». Permet dans des commandes SQL d'annuler des opérations modifiant les données mais non encore validées explicitement par un COMMIT.
Sauvegarde	Sauvegarde de l'ensemble des fichiers composant votre base de données. Si votre système tombe en panne, vous pouvez copier vos données et relancer votre base sur un autre système. La sauvegarde est aussi utile si des données sont altérées : la restauration résout le problème.

Scalability	Capacité à monter en charge.
Schéma	Ensemble d'objets (tables, vues, procédures, fonctions, synonymes, etc.). Un schéma est souvent abusivement associé à l'utilisateur propriétaire de ces objets.
Schéma XML	En cours de définition par un groupe de travail du W3C, les schémas permettront comme les DTD de décrire la structure d'un document XML, mais avec une précision accrue, notamment au niveau du type des données contenues. Contrairement aux DTD, les schémas respecteront la syntaxe XML.
SCOTT	Utilisateur « historique » d'Oracle : c'est le nom du propriétaire du jeu d'essai installé par défaut dans les bases Oracle9i. Mot de passe de SCOTT : TIGER. Pour l'anecdote, SCOTT est l'un des premiers employés d'Oracle !
Script SHELL	Langage de commande sous Linux, principalement utilisé pour l'administration et la gestion du système.
SELECT	Ordre SQL permettant de lire des données contenues dans une base Oracle9i.
Services	Les services sont des programmes Windows invisibles pour l'utilisateur. Ils s'exécutent en « tâche de fond » dès le lancement de l'ordinateur : on peut les comparer à des daemons sous Unix. Les services sont des pilotes de périphérique, des serveurs d'application, le démarrage automatique d'une base de données ou toute autre tâche d'arrière-plan. Définis dans la base de registre, ils sont lancés dans le contexte du système local ou sous un compte utilisateur particulier.
Servlets	Les servlets sont des composants logiciels écrits en Java. Situés sur un serveur, ils remplacent les scripts CGI en augmentant la souplesse et la puissance d'un serveur Web.
Session	Une session s'étend de la connexion à la déconnexion d'une base Oracle9i.
SGA	System Global Area : zone mémoire réservée par une base de données Oracle9i en marche.
SID	Le SID ou System Identifier est un nom unique désignant une instance. En changeant de SID, on peut se connecter à une instance ou à une autre.
Snapshot	Cliché. Dispositif logiciel Oracle de réplication des données. Il effectue un cliché des données, qu'il est possible de répercuter sur d'autres bases Oracle9i.
Solaris	Nom de la version Unix commercialisée par Sun.
SQL	Structured Query Language. Langage standard utilisé pour manipuler des données contenues dans des bases de données relationnelles.
Synonyme	Un alias pour une table, une vue, une séquence, etc. Le synonyme permet de masquer le nom et le propriétaire réel de l'objet. Un synonyme peut être privé ou public.
SYS	L'un des deux utilisateurs (l'autre est SYSTEM) créés par défaut sur chaque base Oracle9i et nécessaires à son fonctionnement. Le mot de passe de l'utilisateur SYS est maintenant obligatoirement changé lors de la création de la base. SYS ne doit être utilisé que lors d'opérations spéciales.
SYSDBA	Fournit, outre le rôle SYSOPER, le privilège DBA qui donne le droit d'accéder à l'ensemble du contenu de la base. Le possesseur de ce rôle est autorisé à créer une base par l'ordre SQL CREATE DATABASE.
SYSOPER	Rôle d'administration spécial qui permet d'effectuer des opérations élaborées sur une base (démarrage, arrêt, restauration) sans disposer du privilège DBA (Data Base Administrator) qui autorise l'accès à l'ensemble du contenu de la base.
SYSTEM	L'un des deux utilisateurs (l'autre est SYS) créés par défaut sur chaque base Oracle9i et nécessaires à son fonctionnement. Le mot de passe de l'utilisateur SYSTEM est maintenant obligatoirement changé lors de la création de la base. SYSTEM constitue le compte d'administration d'une base Oracle9i.

Système d'exploitation	Operating System : système d'exploitation.
Tablespace	Une base Oracle9i conserve ses données dans des tablespaces. À chaque tablespace peut être associé un (ou plusieurs) fichier(s) physique(s). Une tâche précise est souvent affectée à chaque tablespace : contenir les tables, les index, effectuer des tris, des segments d'annulation, etc.
TCP/IP	Standard mondial des protocoles réseau. Largement popularisé par Unix et Internet.
Terminal Server	Service d'émulation de terminaux Windows (service Terminal Server) qui permettent à des programmes de s'exécuter sur un serveur, leur affichage étant déporté sur des clients distants. Ce service est intégré dans tous les serveurs Windows 2000. Il permet ainsi de les administrer à distance.
TNSNAMES.ORA	Fichier contenant la description d'alias Oracle Net, qui permettent à un poste client Oracle9i de se connecter aux différents serveurs.
TSE	Voir Terminal Server.
Tuning ou mise au point	Optimisation. Ce terme a été utilisé par analogie au « tuning » des postes de radio grâce auquel on recherche une station par approches successives.
URL	Uniform Resource Locator. Adresse utilisée pour localiser un serveur Web et une page HTML d'accueil. Par exemple : www.oracle.com
WebDAV	Web-based Distributed Authoring and Versioning. Protocole d'échange de fichiers et de gestion des révisions. Étend les capacités du protocole Web HTTP afin de permettre aux utilisateurs de modifier des documents sur un serveur distant mais aussi de collaborer durant le cycle de vie de ces documents.
Wintel (Windows + Intel)	Cet acronyme désigne la plate-forme PC qui combine habituellement le logiciel Windows de Microsoft, les microprocesseurs et autres circuits d'Intel.
WSH	Windows Script Host. Le WSH est un puissant environnement de script utilisé pour créer et exécuter des scripts sur les systèmes d'exploitation Windows les plus récents. Les scripts peuvent être écrits dans différents langages : Jscript, VBScript ou Windows script.
WWW	World Wide Web. Réseau mondial de serveurs sur Internet possédant chacun une ou plusieurs pages HTML et fournissant des informations et des liens hypertextes vers d'autres documents qui se trouvent sur le même serveur ou non.
XMLQUERY	Langage universel de requête (concurrent des propositions XQL et des dialectes associés) en cours de définition par un groupe W3C impliquant de nombreux acteurs. Cette future norme aura notamment comme objectif de réconcilier SQL et l'ensemble des dialectes actuellement voisins de XQL.
XPATH	Standard du W3C permettant de naviguer au sein d'un document XML. XPath construit une représentation arborescente des documents. Il fournit un ensemble de fonctions permettant de parcourir l'arbre ainsi construit et d'appliquer sur ce dernier des critères de tests et des fonctions de recherche. XQL intègre XPath.
XQL	XML Query Language. Projet du consortium W3C visant à normaliser un langage de requête destiné à manipuler des documents XML. Xpath est un sous-ensemble de XSQL. XSQL n'est pas encore une norme.
XSL	XML Stylesheet Language. Standard du W3C basé sur XML permettant de construire des feuilles de style. L'association d'une feuille de style à un document XML normalise la présentation des données. Pour un même document XML, différentes feuilles de styles peuvent être utilisées suivant le rendu visuel désiré.
XSLT	XML Stylesheet Language Transformation. Langage basé sur XML permettant de modifier la structure et le contenu d'un document XML par application de transformations élémentaires à l'aide d'un langage de script.

2

Mémento SQL

Ce mémento recense les commandes SQL les plus utilisées. Il en existe beaucoup d'autres ! Nous ne vous fournissons pas la syntaxe que vous trouverez dans les documentations papier ou en ligne, mais nous vous expliquons brièvement l'utilité de chacune de ces commandes.

Langage de définition des données

Le DDL (*Data Definition Language*) regroupe les commandes qui permettent de travailler sur la structure destinée à contenir les données. Ces commandes vous permettent de :

- créer, modifier, supprimer tout objet (table, index...) ;
- attribuer et retirer des autorisations ;
- analyser les tables et index pour piloter l'optimiseur statistique d'Oracle9i ;
- mettre en place un audit sur tous les accès aux bases.

Disponibilité d'accès

Certaines commandes nécessitent un accès exclusif à la structure des données pour être en mesure de les utiliser. Ainsi, on ne peut pas modifier la structure d'une table avec la commande ALTER TABLE si un utilisateur, au même instant, travaille ou modifie des données sur cette table.

D'autres commandes n'ont pas besoin d'un accès exclusif aux données. Par exemple, plusieurs utilisateurs peuvent analyser simultanément le contenu d'une table ou d'un index avec la commande ANALYZE.

Impact des modifications

Certaines modifications ont un impact sur les programmes (procédures, fonctions, triggers...) qui exploitent les structures de données. Dans ce cas, les programmes sont notés « à recompiler ». La recompilation de ces objets est automatique lors de l'exécution suivante.

Si vous souhaitez visualiser les dépendances entre des objets, Oracle Enterprise Manager vous y aidera.

Validation des modifications

Attention ! Beaucoup de commandes DDL comme TRUNCATE, CREATE TABLE, etc., intègrent un COMMIT automatique. C'est-à-dire que l'action est définitive et que le ROLLBACK ou retour en arrière, est inactif. Cela signifie aussi que toutes les modifications de données intégrées avant cette commande DDL sont, elles aussi, validées.

Liste commentée des commandes DDL

Voici la liste des commandes DDL. Presque toutes ont un nom précédé de CREATE (création), ALTER (modification) ou DROP (suppression) :

Commande DDL	Action de la commande
ALTER CLUSTER	Modifie les caractéristiques d'un cluster. Les clusters sont très rarement utilisés, car peu utiles. Ils sont conservés pour compatibilité ascendante avec les anciennes versions.
ALTER DATABASE	Ouvre (open) ou monte (mount) une base. Choisit les modes ARCHIVELOG/NOARCHIVELOG. Restaure à partir de bandes. Agit sur les membres des groupes redo-log. Renomme un fichier de la base ou des membres d'un redo-log. Sauvegarde le fichier de contrôle courant. Sauvegarde des commandes SQL dans un fichier trace. Ajoute un nouveau fichier à la base de données. Autorise l'extension automatique des fichiers de la base. Positionne un fichier de la base en ligne ou hors ligne. Modifie le nom de la base de données.
ALTER FUNCTION	Recompile une fonction stockée dans la base.
ALTER INDEX	Modifie les caractéristiques de stockage (storage) d'un index.
ALTER PACKAGE	Recompile un package stocké dans la base.
ALTER PROCEDURE	Recompile une procédure stockée dans la base.
ALTER PROFILE	Modifie les ressources affectées à un profil.
ALTER RESOURCE COST	Formule permettant de calculer le total des ressources mises à disposition d'une session.
ALTER ROLE	Modifie les autorisations regroupées sous le rôle.
ALTER ROLLBACK SEGMENT	Modifie les caractéristiques de stockage (storage) d'un rollback segment. Positionne un rollback segment en ligne, hors ligne. Spécifie une taille optimale à un rollback segment.

Commande DDL	Action de la commande
ALTER SEQUENCE	Modifie les valeurs données à une séquence.
ALTER SNAPSHOT	Modifie le fonctionnement d'un cliché (snapshot) : ses caractéristiques, son type de rafraîchissement, sa fréquence de rafraîchissement.
ALTER SHAPSHOT LOG	Modifie les caractéristiques de stockage (storage) du log d'un cliché.
ALTER TABLE	Ajoute une colonne ou une contrainte d'intégrité à une table. Modifie une colonne ou les caractéristiques de stockage (storage) d'une table. Active, désactive, supprime des contraintes d'intégrité. Active et désactive les triggers d'une table. Agit sur les extensions (extents) d'une table. Modifie le degré de parallélisme affecté à une table (nombre de processus actifs simultanément sur une table).
ALTER TABLESPACE	Ajoute et renomme des fichiers de la base. Modifie les caractéristiques de stockage (storage). Met en ligne, hors ligne un tablespace. Commence et arrête une sauvegarde (backup) en ligne. Autorise ou interdit les écritures sur un tablespace (cas des disques optiques ou CD-Rom).
ALTER TRIGGER	Active et désactive un trigger lié à une table.
ALTER TYPE	Modifie un type.
ALTER USER	Modifie les caractéristiques d'un utilisateur : mot de passe, tablespace affecté par défaut, tablespace temporaire, quotas, profils ou rôles.
ALTER VIEW	Modifie et recompile une vue.
ANALYZE	Analyse une table, un index, pour fournir des statistiques fiables et actualisées à l'optimiseur statistique. Permet d'identifier si une table est chaînée.
AUDIT	Permet de surveiller toutes les actions ou opérations effectuées sur la base de données.
COMMENT	Commente les tables, colonnes, vues, clichés d'une base à des fins d'informations.
CREATE CLUSTER	Crée un cluster contenant une ou plusieurs tables (peu utile).
CREATE CONTROLFILE	Crée un nouveau fichier de contrôle.
CREATE DATABASE	Crée une nouvelle base de données.
CREATE DATABASE LINK	Crée un nouveau lien vers une base de données distante.
CREATE DIRECTORY	Utilisé pour les BFILE conservés hors de la base de données.
CREATE FUNCTION	Crée une fonction dans la base de données.
CREATE INDEX	Crée un index sur une table.
CREATE LIBRARY	Crée une bibliothèque de fonctions qu'il est possible d'appeler hors de la base de données.
CREATE PACKAGE	Crée des packages regroupant des procédures, fonctions, variables locales.
CREATE PACKAGE BODY	Crée le corps du package.
CREATE PROCEDURE	Crée une procédure stockée dans la base.
CREATE PROFILE	Crée un profil limitant les ressources consommées.
CREATE ROLE	Crée un rôle qui regroupera tout un ensemble de droits d'accès.

CREATE ROLLBACK SEGMENT	Crée un rollback segment pour conserver les données en attente de validation et assurer les lectures consistantes.
CREATE SYNONYM	Crée un synonyme.
CREATE TABLE	Crée une table, ses colonnes, ses caractéristiques de stockage (storage) et ses contraintes d'intégrité.
CREATE TABLESPACE	Crée un emplacement dans la base de données où seront stockés des tables, des index, des rollback segments, des espaces de tri temporaires. Cet espace est lié à un ou plusieurs fichiers répartis sur les disques de l'ordinateur.
CREATE TRIGGER	Crée un déclencheur (trigger) lié à une table.
CREATE TYPE	Crée un type objet.
CREATE USER	Crée un utilisateur.
CREATE VIEW	Crée une vue portant sur une ou plusieurs tables ou vues.
DROP CLUSTER	Supprime un cluster.
DROP DATABASE LINK	Supprime un lien vers une autre base de données.
DROP DIRECTORY	Supprime un répertoire.
DROP FUNCTION	Supprime une fonction.
DROP INDEX	Supprime un index.
DROP LIBRARY	Supprime une bibliothèque.
DROP PACKAGE	Supprime un package.
DROP PROCEDURE	Supprime une procédure.
DROP PROFILE	Supprime un profil utilisateur.
DROP ROLE	Supprime un rôle regroupant un ensemble d'autorisations.
DROP ROLLBACK SEGMENT	Supprime un rollback segment.
DROP SEQUENCE	Supprime un compteur (séquence).
DROP SNAPSHOT	Supprime un cliché.
DROP SNAPSHOT LOG	Supprime le log d'un cliché.
DROP SYNONYM	Supprime un synonyme.
DROP TABLE	Supprime une table et ses données.
DROP TABLESPACE	Supprime un tablespace et tout ce qu'il contient.
DROP TRIGGER	Supprime un déclencheur (trigger).
DROP TYPE	Supprime un type objet.
DROP USER	Supprime un utilisateur et tous les objets qu'il possède.
DROP VIEW	Supprime une vue.
GRANT	Donne des droits, privilèges, rôles à des utilisateurs ou à des rôles.
NOAUDIT	Affecte une commande AUDIT transmise préalablement.
RENAME	Change le nom d'une table, d'un index, d'une vue...
REVOKE	Retire des droits, privilèges, rôles attribués à des utilisateurs ou à des rôles.
TRUNCATE	Supprime de manière irréversible l'ensemble des données contenues dans une table, sans affecter la structure de la table.

Langage de manipulation de données (DML)

Le langage de manipulation de données ou DML (*Data Manipulation Language*) est utilisé pour interroger, modifier ou supprimer les données contenues dans des schémas, c'est-à-dire dans les structures de données.

Disponibilité d'accès

Les ordres SELECT (lecture des données) ne sont jamais bloqués, même si des utilisateurs modifient simultanément les données consultées. Dans ce cas, vous lisez les données avant modification, telles qu'elles sont conservées dans les rollback segments.

Le cas le plus fréquent de blocage (bien que rare) survient lorsque vous essayez de modifier/supprimer un enregistrement sur lequel un autre utilisateur a posé un verrou. Il faut alors attendre que ce dernier valide ou annule son action pour avoir accès, en modification/suppression, à l'enregistrement.

Ce cas est rare, car les verrous (qui sont posés automatiquement par Oracle au cours des modifications) ne concernent que les enregistrements modifiés. Oracle ne verrouille pas des pages mémoire ou disque (qui contiennent alors des données modifiées ou non), mais des enregistrements. C'est le niveau de granularité le plus fin.

Validation des modifications

Les ordres DML modifiant des données (INSERT, UPDATE, DELETE) doivent être explicitement validés par un COMMIT ou annulés par un ROLLBACK.

L'ordre de consultation SELECT ne pose pas de verrou en lecture et ne nécessite aucune action supplémentaire.

Liste commentée des commandes DML

La liste suivante répertorie les commandes DML (*Data Manipulation Language*) :

Commande DML	Action de la commande
DELETE	Supprime des enregistrements d'une table.
EXPLAIN PLAN	Montre le plan d'exécution d'un ordre SQL, c'est-à-dire la façon dont la base Oracle9i l'a traité.
INSERT	Ajoute de nouveaux enregistrements à une table.
LOCK TABLE	Permet de placer des verrous manuellement. Très peu utile et très peu utilisée. Le mécanisme par défaut d'Oracle est, bien souvent, largement suffisant.
SELECT	Recherche des enregistrements dans une ou plusieurs tables via un ensemble de critères de recherche, de tri, etc.
UPDATE	Modifie des données d'enregistrements existants.

Contrôle des transactions

Le contrôle des transactions permet de valider ou d'annuler les modifications effectuées avec les commandes de manipulation des données (DML).

Commande de contrôle	Action de la commande
COMMIT	Valide de façon permanente l'ensemble des actions effectuées depuis le début de la transaction (c'est-à-dire depuis le COMMIT ou le ROLLBACK précédent).
ROLLBACK	Annule toutes les actions effectuées depuis le début de la transaction ou depuis le dernier SAVEPOINT.
SAVEPOINT	Positionne un point qui permet de faire un ROLLBACK jusqu'à lui.
SET TRANSACTION	Modifie les propriétés de la transaction courante.

Contrôle des sessions

Le contrôle des sessions permet de modifier les propriétés de la session en cours. Ces commandes ne comportent pas de COMMIT automatique.

Le langage PL/SQL n'offre pas de commandes de contrôle des sessions.

Contrôle des sessions	Action de la commande
ALTER SESSION	Active/désactive la trace d'un ordre SQL. Change les valeurs NLS (National Language Support) affectées à une session. Ferme explicitement un Database Link (très utile si un réseau lent, dont la facturation dépend du temps de connexion, relie des bases distantes). Oblige des transactions distribuées à valider des transactions douteuses. Modifie le fonctionnement de l'optimiseur statistique d'Oracle9i.
SET ROLE	Active et désactive des rôles affectés à la session en cours.

Contrôle du système

Cette commande modifie dynamiquement certaines propriétés d'une instance en fonctionnement, c'est-à-dire qu'elle est capable d'agir sur la zone mémoire réservée à Oracle9i lors de son fonctionnement.

Seul un nombre limité de paramètres présents dans le fichier *initSID.ora* (*SID* étant votre instance) sont modifiables dynamiquement par cette commande.

Cette commande ne comporte pas de COMMIT automatique.

Commande	Action de la commande
ALTER SYSTEM	Modifie dynamiquement les paramètres d'une instance.

Procédures pour le DBA

Ces procédures regroupent un ensemble d'ordres SQL utiles aux administrateurs de données (DBA : DataBase Administrator).

Elles permettent de visualiser un ensemble de caractéristiques de la base de données. Vous pouvez aussi utiliser Oracle Enterprise Manager qui permet d'obtenir et d'imprimer des documents au format HTML. L'évolution des bases est alors observable et contrôlable, mais un script de commandes SQL paramétrable et modifiable suivant vos besoins est souvent fort utile.

Les vues et tables utilisées sont accessibles par tout utilisateur possédant le rôle DBA. Toutes les vues nécessaires à ces scripts ont des synonymes publics et sont accessibles à tout utilisateur disposant du privilège `SELECT ANY TABLE`.

Ce script d'audit est conçu pour être lancé avec l'outil SQL*Plus et crée automatiquement un fichier résultat. Pour plus de clarté, nous commentons l'objectif et l'utilisation de chacun de ces ordres.

Tous les ordres SQL peuvent être regroupés dans un fichier *audit.sql*, lancé à partir de SQL*Plus par la commande suivante :

```
SQL> connect system -ou un autre utilisateur DBA
SQL> start lieu_ou_se_trouve_le_fichier/audit.sql
```

Nous vous encourageons, bien évidemment, à adapter ces commandes à vos besoins.

Présente dans les précédents ouvrages, cette annexe a été maintenue. Pourtant, avec l'amélioration décisive d'Oracle Enterprise Manager version 10g, je vous engage fortement à utiliser maintenant cet outil plutôt que de construire vos propres scripts d'administration et de surveillance.

Bannière d'accueil

Un fichier *spool* enregistre tous les résultats des commandes SQL lancées par la suite. La bannière permet d'identifier le moment où l'audit a été lancé. L'utilisateur connecté y figure.

Syntaxe SQL :

```
spool /emplacement/AUDIT.LIS

col jour nopri new_value to_day
tttitle center '*****' skip 2 -
      left  '          ' to_day -
      right 'Auteur : G.BRIARD ' skip 2 -
      center 'AUDIT DE LA BASE ORACLE' skip 2 -
      center '*****'
select to_char(sysdate,'1e "DD/MM/YY HH:MM') jour from dual
/
set pause off
set pagesize 150
col tablespace_name for a15
col owner for a6
tttitle center 'Utilisateur Connecté' skip 2
show user
```

Résultat :

```
*****
1e 21/09/05 22:02          G.BRIARD
AUDIT DE LA BASE ORACLE
*****

1'utilisateur est "SYSTEM"
```

Version Oracle utilisée

Récupération de la version Oracle utilisée.

Syntaxe SQL :

```
tttitle center "Version Oracle" skip 2
select banner from v$version
/
```

AUTRE OPTION : SELECT * FROM PRODUCT_COMPONENT_VERSION ;

Résultat :

Version Oracle

BANNER

```
-----
Oracle9i Enterprise Edition Release 9.0.1.0.0 - Production
PL/SQL Release 9.0.1.0.0 - Production
CORE          9.0.1.0.0      Production
TNS for Linux: Version 9.0.1.0.0 - Production
NLSRTL Version 9.0.1.0.0 - Production
```

Base auditée

Identification de la base auditée. Figurent son nom, sa date de création, son mode ARCHIVELOG ou NOARCHIVELOG.

Syntaxe SQL :

```
tttitle center "Base auditée" skip 2
select name "Nom Base", created, log_mode,checkpoint_change#,
archive_change# from v$database
/
```

Résultat :

Base auditée

Nom Base	CREATED	LOG_MODE	CHECKPOINT_CHANGE#	ARCHIVE_CHANGE#
GILLES	28/07/01	NOARCHIVELOG	392503	331044

Taille de la SGA

Taille totale de la zone mémoire réservée par Oracle pour l'instance.

Syntaxe SQL :

```
tttitle center 'Taille de la SGA' skip 2
break on report
compute sum of value on report
col name for a20
col value for 99999999999
select * from V$SGA
/
clear break
clear compute
```

Résultat :

Taille de la SGA

Nom du parametre	Valeur
Fixed Size	279580
Variable Size	46137344
Database Buffers	8388608
Redo Buffers	532480
sum	55338012

Cache Hit Ratio

Le Cache Hit Ratio est une information importante pour le tuning. Elle permet de connaître le pourcentage de requêtes SQL résolues, en utilisant les données déjà en mémoire, plutôt que d'avoir recours aux accès disque. Plus sa valeur est grande, mieux c'est.

Cette valeur dépend du délai écoulé depuis le dernier démarrage de base, car il faut « monter » en mémoire les données avant de pouvoir les réutiliser. Elle dépend aussi du type d'utilisation de la base, transactionnelle (toujours les mêmes données accédées) ou décisionnelle (données différentes accédées).

Syntaxe SQL :

```

tttitle center 'Calcul du Cache Hit Ratio' skip 2
select distinct ( to_number(S2.value) +
                  to_number(S3.value) -
                  to_number(S1.value))/
                  (to_number(S2.value) +
                  to_number(S3.value))*100 "Cache Hit Ratio"
from v$sysstat S1,
     v$sysstat S2,
     v$sysstat S3,
     v$statname N1,
     v$statname N2,
     v$statname N3
where S1.statistic# = N1.statistic#
and S2.statistic# = N2.statistic#
and S3.statistic# = N3.statistic#
and N1.name = 'physical reads'
and N2.name = 'consistent gets'
and N3.name = 'db block gets'
/

```

Résultat :

Calcul du Cache Hit Ratio

Cache Hit Ratio

63,463895

Paramètres de l'Init.ora

Figure ici l'ensemble des paramètres d'initialisation, ceux qui se trouvent dans le fichier *initSID.ora* de l'instance *SID* et ceux qui, non précisés dans ce fichier, ont des valeurs par défaut.

Syntaxe SQL :

```

tttitle center "Paramètres de INIT.ORA" skip 2
col name for a40 heading "Nom du parametre"
col value for a38 heading "Valeur"
select name, value from v$parameter
order by name
/

```

Résultat :

Paramètres de INIT.ORA

Nom du parametre	Valeur

active_instance_count	
aq_tm_processes	0
archive_lag_target	0
audit_file_dest	?/rdbms/audit
audit_trail	NONE
background_core_dump	partial
background_dump_dest	/u01/app/oracle/admin/gilles/bdump
backup_tape_io_slaves	FALSE
bitmap_merge_area_size	1048576
blank_trimming	FALSE
buffer_pool_keep	
buffer_pool_recycle	
circuits	170
cluster_database	FALSE
cluster_database_instances	1
cluster_interconnects	
commit_point_strength	1
compatible	9.0.0
control_file_record_keep_time	7
control_files	/u01/app/oracle/oradata/gilles/control01.ctl, /u01/app/oracle/oradata/gilles/control03.ctl, /u01/app/oracle/oradata/gilles/control02.ctl
core_dump_dest	/u01/app/oracle/admin/gilles/cdump
cpu_count	1
create_bitmap_area_size	8388608
cursor_sharing	EXACT
cursor_space_for_time	FALSE
db_block_buffers	0
db_block_checking	FALSE
db_block_checksum	TRUE
db_block_size	8192
db_cache_advice	OFF
db_cache_size	8388608
transactions_per_rollback_segment	5
undo_management	AUTO
undo_retention	900
undo_suppress_errors	FALSE
undo_tablespace	UNDOTBS
use_indirect_data_buffers	FALSE
user_dump_dest	/u01/app/oracle/admin/gilles/udump
utl_file_dir	
workarea_size_policy	MANUAL

251 ligne(s) sélectionnée(s).

Liste des fichiers physiques

Liste des fichiers physiques rattachés à des tablespaces, leurs noms, tailles, emplacements et propriétés.

Syntaxe SQL :

```
tttitle center "Liste des fichiers physiques rattachés aux tablespaces" skip 2
col "Fichier physique" for a40
col TBS for a11
col bytes for 999999999
break on report
compute sum of BYTES on status report
select file_name "Fichier Physique",tablespace_name "TBS",bytes,status
  from sys.dba_data_files
  order by tablespace_name
/
clear break
clear compute
```

Résultat :

Liste des fichiers physiques rattachés aux tablespaces

Fichier physique	TBS	BYTES	STATUS
/u01/oradata/gilles/indx01.dbf	INDX	26214400	AVAILABLE
/u01/oradata/gilles/system01.dbf	SYSTEM	340787200	AVAILABLE
/u01/oradata/gilles/temp01.dbf	TEMP	41943040	AVAILABLE
/u01/oradata/gilles/tools01.dbf	TOOLS	10485760	AVAILABLE
/u01/oradata/gilles/tools02.dbf	TOOLS	10485760	AVAILABLE
/u01/oradata/gilles/tools03.dbf	TOOLS	10485760	AVAILABLE
/u01/oradata/gilles/undotbs01.dbf	UNDOTBS	209715200	AVAILABLE
/u01/oradata/gilles/users01.dbf	USERS	28835840	AVAILABLE
/u01/oradata/gilles/drsys01.dbf	DRSYS	5242880	AVAILABLE
/u01/oradata/gilles/example01.dbf	EXAMPLE	86507520	AVAILABLE

Liste des fichiers physiques auto extensibles

Liste des fichiers physiques dont la taille peut s'accroître automatiquement, ainsi que les caractéristiques d'accroissement.

Syntaxe SQL :

```
tttitle center "Liste des fichiers physiques auto extensibles" skip 2
col "Fichier physique" for a40
select file_name "Fichier Physique",auto extensible, maxbytes, maxblocks, increment_by
  from sys.dba_data_files
  order by tablespace_name
/
```

Résultat :

Liste des fichiers physiques auto extensibles

Fichier Physique	AUT	MAXBYTES	MAXBLOCKS	INCREMENT_BY
-----	-----	-----	-----	-----
/u01/oradata/ORCL/RBS10RCL.ORA	YES	157286400	76800	2560
/u01/oradata/ORCL/SYS10RCL.ORA	YES	209715200	102400	5120
T_COMPTOIR	NO	0	0	0
/u01/oradata/ORCL/TMP10RCL.ORA	YES	157286400	76800	2560
/u01/oradata/ORCL/USR10RCL.ORA	YES	157286400	76800	2560

Liste des fichiers redo-log

Liste des fichiers redo-log, leurs noms, tailles, emplacements et propriétés.

Syntaxe SQL :

tttitle center "Liste des fichiers redo-log physiques" skip 2
col "nom des fichiers redo-log " for a60
select group#,status,member "Nom des fichiers redo-log" from v\$logfile
/
/

Résultat :

Liste des fichiers redo-log physiques

GROUP#	STATUS	Nom des fichiers redo-log
-----	-----	-----
1		/u01/oradata/gilles/redo01.log
2	STALE	/u01/oradata/gilles/redo02.log
3	STALE	/u01/oradata/gilles/redo03.log

Liste des fichiers de contrôle

Liste des fichiers de contrôle, leurs noms, emplacements et propriétés.

Syntaxe SQL :

tttitle center "Liste des fichiers de contrôle physiques" skip 2
col "nom des fichiers de controle" for a70
select status, name "nom des fichiers de contrôle" from v\$controlfile
/
/

Résultat :

Liste des fichiers de contrôle physiques

STATUS	nom des fichiers de contrôle
-----	-----
	/u01/oradata/gilles/CTL10RCL.ORA
	/u01/oradata/gilles/CTL20RCL.ORA

Liste et propriétés des tablespaces

Liste des tablespaces existants ainsi que leurs propriétés.

Syntaxe SQL :

tttitle center "Liste des tablespaces" skip 2
col initial_extent for 9999999 head "Ini.Ext"

```
col next_extent    for 9999999 head "Nex.Ext"
col pct_increase   for 999999  head "Pct.In"
col min_extents    for 9999999 head "Min.Ext"
col max_extents    for 9999999 head "Max.Ext"
```

```
select * from sys.dba_tablespaces
/
```

Résultat :

Liste des tablespaces

Tbs	Status	CONTENTS	LOGGING	EXTENT_MAN	ALLOCATIO	PLU	SEGMENT
SYSTEM	ONLINE	PERMANENT	LOGGING	DICTIONARY	USER	NO	MANUAL
UNDOTBS	ONLINE	UNDO	LOGGING	LOCAL	SYSTEM	NO	MANUAL
DRSYS	ONLINE	PERMANENT	LOGGING	LOCAL	SYSTEM	NO	MANUAL
EXAMPLE	ONLINE	PERMANENT	LOGGING	LOCAL	SYSTEM	NO	MANUAL
INDX	ONLINE	PERMANENT	LOGGING	LOCAL	SYSTEM	NO	MANUAL
TEMP	ONLINE	TEMPORARY	NOLOGGING	LOCAL	UNIFORM	NO	MANUAL
TOOLS	ONLINE	PERMANENT	LOGGING	LOCAL	SYSTEM	NO	MANUAL
USERS	ONLINE	PERMANENT	LOGGING	LOCAL	SYSTEM	NO	MANUAL

Taille totale et espace disponible dans les tablespaces

Pour calculer la taille de chacun des tablespaces, l'on additionne la taille des fichiers composant chaque tablespace. La taille totale des tablespaces ne constitue qu'une « enveloppe ». L'ordre SQL suivant permet de récupérer l'espace disponible restant à l'intérieur de cette enveloppe. Ce résultat est intéressant, que les fichiers soient AUTOEXTENSIBLE ou non.

Syntaxe SQL :

```
SQL> tttitle center "Taille et occupation des tablespaces" skip 2
SQL> col "% occupé" for 99.9
SQL>
SQL> select A.tablespace_name,
2      A.total_size "Taille totale (Ko)",
3      B.free_size  "Espace disponible (Ko)",
4      ((A.total_size - B.free_size)*100 / A.total_size) "% occupé"
5  from
6  (select tablespace_name,
7      sum(bytes)/1024 total_size
8      from sys.dba_data_files
9      group by tablespace_name) A ,
10 (select tablespace_name,
11      sum(bytes)/1024 free_size
12      from sys.dba_free_space
13      group by tablespace_name) B
14 where a.tablespace_name = b.tablespace_name
15 /
```

Résultat :

Taille et occupation des tablespaces

Tablespace	Taille totale (Ko)	Espace disponible (Ko)	% occupé
DRSYS	25600	17600	31.3
EXAMPLE	160640	512	99.7
INDX	30720	30592	.4
OEM_REPOSITORY	35848	3264	90.9
SYSTEM	337920	39576	88.3
TOOLS	15360	15232	.8
UNDOTBS	209920	180800	13.9
USERS	33280	2112	93.7

Espace disponible dans les tablespaces

Un tablespace peut offrir beaucoup d'espace disponible, mais cet espace peut être morcelé. C'est la raison pour laquelle, même s'il reste beaucoup d'espace, vous ne pourrez pas créer une table d'une taille initiale de 50 Mo, si le plus grand extent dans le tablespace n'est pas de taille égale ou supérieure.

L'ordre SQL suivant permet de mesurer l'étendue du morcellement des tablespaces. L'espace disponible est indiqué en nombre de blocks. La taille d'un bloc est fournie dans le paramètre DB_BLOCK_SIZE de l'*init.ora* (voir plus haut dans cette annexe).

Syntaxe SQL :

```

ttitle center "Espace disponible en nombre de blocks contigus" skip 1 -
       center " (1 bloc vaut db_block_size octets)"          skip 2
select tablespace_name,
       round(avg(blocks)) "En Moyenne",
       min(blocks) "Minimum",
       max(blocks) "Maximum",
       count(*) "Nombre"
from sys.dba_free_space
group by tablespace_name
/

```

Résultat :

Espace disponible en nombre de blocs contigus
(1 block vaut db_block_size octets)

Tablespace	En Moyenne	Minimum	Maximum	Nombre
DRSYS	2200	2200	2200	1
EXAMPLE	64	64	64	1
INDX	1912	632	3192	2
OEM_REPOSITORY	408	408	408	1
SYSTEM	2474	6	4941	2
TOOLS	952	632	1272	2
UNDOTBS	1027	8	13560	22

Liste des segments d'annulation

Les segments d'annulation (anciennement rollback segments) permettent de conserver les données modifiées, insérées, supprimées, en attente de validation ou d'annulation de la transaction.

Syntaxe SQL :

```
ttitle center "Liste des segments d'annulation " skip 2
col nomsegment for a10
col iniext for 999999999
col nextex for 999999999
col pctin for 9999
col status for a8
col tablespace_name for a13 heading "Tablespace"
select segment_name "NomSegment",owner,tablespace_name,initial_extent "IniExt",
       next_extent "NextEx",pct_increase "PctIn",status
from sys.dba_rollback_segs
/
```

Résultat :

Liste des segments d'annulation

NomSegment	OWNER	Tablespace	IniExt	NextEx	PctIn	STATUS
SYSTEM	SYS	SYSTEM	57344	57344	0	ONLINE
_SYSSMU1\$	PUBLIC	UNDOTBS	131072			ONLINE
_SYSSMU2\$	PUBLIC	UNDOTBS	131072			ONLINE
_SYSSMU3\$	PUBLIC	UNDOTBS	131072			ONLINE
_SYSSMU4\$	PUBLIC	UNDOTBS	131072			ONLINE
_SYSSMU5\$	PUBLIC	UNDOTBS	131072			ONLINE
_SYSSMU6\$	PUBLIC	UNDOTBS	131072			ONLINE
_SYSSMU7\$	PUBLIC	UNDOTBS	131072			ONLINE
_SYSSMU8\$	PUBLIC	UNDOTBS	131072			ONLINE
_SYSSMU9\$	PUBLIC	UNDOTBS	131072			ONLINE
_SYSSMU10\$	PUBLIC	UNDOTBS	131072			ONLINE

Place utilisée par segment

Cet ordre SQL est l'un des plus intéressants de cette série. Il visualise, utilisateur par utilisateur, les tables, les index et les rollback segments propriété de l'utilisateur SYS.

Pour chaque objet sont indiqués : le tablespace dans lequel il est placé, la taille de l'objet et le nombre d'extensions. Ce nombre d'extensions indique la « fragmentation » de l'objet, comme indiqué au chapitre 25, *Optimisation et performances*.

Syntaxe SQL :

```
ttitle center "Place utilisée par segment " skip -
       center "(tri par owner puis type de segment)" skip 2
col owner for a8
col segment_name for a17
col segment_type for a9 heading "Type Seg"
```

```

col extents for 9999 heading "Ext"
col max_extents for 99999 heading "Mext"
select segment_name,tablespace_name,segment_type,extents,max_extents,bytes,owner
  from sys.dba_segments
  where owner != 'SYS' or ( owner = 'SYS' and segment_type = 'ROLLBACK')
  order by owner,segment_Type desc, segment_name
/

```

Résultat :

Place utilisée par segment
(tri par owner puis type de segment)

SEGMENT_NAME	Tablespace	Type Seg	Ext	Mext	BYTES	OWNER
---	---	---	---	---	---	---
CATÉGORIES	T_COMPTOIR	TABLE	7	121	266240	COMPTOIR
CLIENTS	T_COMPTOIR	TABLE	2	121	20480	COMPTOIR
COMMANDES	T_COMPTOIR	TABLE	6	121	174080	COMPTOIR
DÉTAILS_COMMAND	T_COMPTOIR	TABLE	5	121	112640	COMPTOIR
EMPLOYÉS	T_COMPTOIR	TABLE	9	121	593920	COMPTOIR
...						
CUSTOMER	USER_DATA	TABLE	1	121	10240	DEMO
DEPARTMENT	USER_DATA	TABLE	1	121	10240	DEMO
EMPLOYEE	USER_DATA	TABLE	1	121	10240	DEMO
ITEM	USER_DATA	TABLE	2	121	20480	DEMO
JOB	USER_DATA	TABLE	1	121	10240	DEMO
...						
EVT_DEST_PROFI	USER_DATA	TABLE	1	121	10240	OEM
EVT_HISTORY	USER_DATA	TABLE	1	121	10240	OEM
EVT_INSTANCE	USER_DATA	TABLE	1	121	10240	OEM
EVT_MAIL_CONFIG	USER_DATA	TABLE	1	121	10240	OEM
...						
CHAINED_ROWS	USER_DATA	TABLE	1	121	10240	SCOTT
DEPT	USER_DATA	TABLE	1	121	10240	SCOTT
EMP	USER_DATA	TABLE	1	121	10240	SCOTT
EMPBIS	USER_DATA	TABLE	1	121	10240	SCOTT
PLAN_TABLE	USER_DATA	TABLE	1	121	10240	SCOTT
SALGRADE	USER_DATA	TABLE	1	121	10240	SCOTT
TEST	USER_DATA	TABLE	1	121	10240	SCOTT
PK_DEPT	USER_DATA	INDEX	1	121	10240	SCOTT
PK_EMP	USER_DATA	INDEX	1	121	10240	SCOTT
SYS_C00935	USER_DATA	INDEX	1	121	10240	SCOTT
RB_TEMP	SYSTEM	ROLLBACK	21	121	2150400	SYS
RB1	ROLLBACK_DATA	ROLLBACK	43	121	2201600	SYS
RB10	ROLLBACK_DATA	ROLLBACK	2	121	102400	SYS
RB11	ROLLBACK_DATA	ROLLBACK	2	121	102400	SYS
...						

Liste des procédures, fonctions, triggers, packages

Fournit la liste des procédures, fonctions, triggers, packages hébergés dans la base de données.

Syntaxe SQL :

```
ttitle "Liste des procédures, fonctions, triggers" skip 2

col object_name for a20
col owner for a20
select owner, object_name, object_type, status from dba_objects
where object_type in ('PROCEDURE','FUNCTION','TRIGGER','PACKAGE')
and owner not in ('SYS','SYSTEM')
order by owner, object_name
/
```

Résultat :

Liste des procédures, fonctions, triggers

OWNER	OBJECT_NAME	OBJECT_TYPE	STATUS
GB_SITE	CORNERS_IU	TRIGGER	VALID
GB_SITE	HOME	PROCEDURE	VALID
GB_SITE	PERSPECTIVES_I	TRIGGER	VALID
GB_SITE	TOPICS_I	TRIGGER	VALID
GB_SITE	TRAVELER	PROCEDURE	VALID
GB_SITE	WWV_URLENCODE	FUNCTION	VALID
SCOTT	F_EURO	FUNCTION	VALID
SCOTT	GILLES	PROCEDURE	VALID
SCOTT	GIVE_RAISE	PROCEDURE	INVALID
SCOTT	TESTJDBC	PROCEDURE	INVALID
....			

Liste et profil des utilisateurs

Fournit la liste des utilisateurs déclarés et les profils (ressources autorisées) qui leur sont octroyés.

Syntaxe SQL :

```
ttitle center "Liste et profil des utilisateurs" skip 2
col "Def. TBS" for a11
col "Temp. TBS" for a10
col C for 9
col R for 9
col DBA for 999
col owner for a6
select username, default_tablespace "Def. TBS",
temporary_tablespace "Temp. TBS", profile
from sys.dba_users
order by username
/
```

Résultat :
Liste et profil des utilisateurs

USERNAME	Def. TBS	Temp. TBS	PROFILE
COMPTOIR	SYSTEM	SYSTEM	DEFAULT
CTXSYS	USER_DATA	USER_DATA	DEFAULT
DBSNMP	SYSTEM	SYSTEM	DEFAULT
DEMO	USER_DATA	TEMPORARY_DATA	DEFAULT
DRSYS	SYSTEM	SYSTEM	DEFAULT
MDSYS	USER_DATA	TEMPORARY_DATA	DEFAULT
OEM	USER_DATA	SYSTEM	DEFAULT
ORDSYS	SYSTEM	SYSTEM	DEFAULT
SCOTT	USER_DATA	TEMPORARY_DATA	DEFAULT
SYS	SYSTEM	SYSTEM	DEFAULT
SYSTEM	USER_DATA	TEMPORARY_DATA	DEFAULT

Liste des utilisateurs possédant les privilèges SYSDBA et SYSOPER

Ces utilisateurs ont la possibilité de démarrer/arrêter les bases et disposent de droits étendus.

Syntaxe SQL :
ttitle center "Privilèges SYSOPER et SYSDBA donnés" skip 2

select username, sysoper, sysdba from v\$pwfile_users
order by username
/

Résultat :
Privilèges SYSOPER et SYSDBA donnés

USERNAME	SYSOPER	SYSDBA
INTERNAL	TRUE	TRUE
ORCL	FALSE	TRUE
SYS	TRUE	TRUE
SYSTEM	FALSE	TRUE

Liste des rôles existants

Un rôle regroupe sous un même nom un ensemble de droits sur différents objets. Une seule opération permet d'allouer l'ensemble de ces droits à un utilisateur (ou à un autre rôle).

Syntaxe SQL :
ttitle center "Liste des rôles existants" skip 2
select role, password_required from sys.db_roles
order by role
/

Résultat :

Liste des rôles existants

ROLE	PASSWORD

AQ_ADMINISTRATOR_ROLE	NO
AQ_USER_ROLE	NO
CONNECT	NO
CTXADMIN	NO
CTXAPP	NO
CTXUSER	NO
DBA	NO
DELETE_CATALOG_ROLE	NO
EXECUTE_CATALOG_ROLE	NO
EXP_FULL_DATABASE	NO
IMP_FULL_DATABASE	NO
PLUSTRACE	NO
RECOVERY_CATALOG_OWNER	NO
RESOURCE	NO
SELECT_CATALOG_ROLE	NO
SNMPAGENT	NO

Rôles affectés aux utilisateurs

Visualise la liste des rôles attribués aux utilisateurs.

Syntaxe SQL :

```
tttitle center "Rôles affectés aux utilisateurs" skip 2
break on grantee
select grantee, granted_role,admin_option,default_role
from sys.dba_role_privs
order by grantee, granted_role
/
```

Résultat :

Rôles affectés aux utilisateurs

GRANTEE	GRANTED_ROLE	ADM	DEF

COMPTOIR	CONNECT	NO	YES
	RESOURCE	NO	YES
DBA	DELETE_CATALOG_ROLE	YES	YES
	EXECUTE_CATALOG_ROLE	YES	YES
	EXP_FULL_DATABASE	NO	YES
	IMP_FULL_DATABASE	NO	YES
	PLUSTRACE	YES	YES
	SELECT_CATALOG_ROLE	YES	YES
DBSNMP	CONNECT	NO	YES
	RESOURCE	NO	YES
	SNMPAGENT	NO	YES

DEMO	CONNECT	NO	YES
	RESOURCE	NO	YES
EXP_FULL_DATABASE	EXECUTE_CATALOG_ROLE	NO	YES
	SELECT_CATALOG_ROLE	NO	YES
IMP_FULL_DATABASE	EXECUTE_CATALOG_ROLE	NO	YES
	SELECT_CATALOG_ROLE	NO	YES
OEM	CONNECT	NO	YES
	DBA	NO	YES
	RESOURCE	NO	YES
SCOTT	CONNECT	NO	YES
	PLUSTRACE	NO	YES
	RESOURCE	NO	YES
SYSTEM	DBA	YES	YES
...			

Liste des profils existants

Syntaxe SQL :

```
tttitle center "Liste des profils existants" skip 2
col profile for a15
col resource_name for a25
col limit for a35
break on profile
select profile, resource_name, limit from sys.dba_profiles
/
```

Résultat :

Liste des profils existants

PROFILE	RESOURCE_NAME	LIMIT
DEFAULT	COMPOSITE_LIMIT	UNLIMITED
	FAILED_LOGIN_ATTEMPTS	UNLIMITED
	SESSIONS_PER_USER	UNLIMITED
	PASSWORD_LIFE_TIME	UNLIMITED
	CPU_PER_SESSION	UNLIMITED
	PASSWORD_REUSE_TIME	UNLIMITED
	CPU_PER_CALL	UNLIMITED
	PASSWORD_REUSE_MAX	UNLIMITED
	LOGICAL_READS_PER_SESSION	UNLIMITED
	PASSWORD_VERIFY_FUNCTION	UNLIMITED
	LOGICAL_READS_PER_CALL	UNLIMITED
	PASSWORD_LOCK_TIME	UNLIMITED
	IDLE_TIME	UNLIMITED
	PASSWORD_GRACE_TIME	UNLIMITED
	CONNECT_TIME	UNLIMITED
	PRIVATE_SGA	UNLIMITED

Fin du script

La sortie du script peut être automatisée.

```
clear break
spool off
set pagesize 20
exit
```

Extensions de fichiers

Cette annexe précise la signification des différentes extensions de fichiers utilisées dans une installation Oracle. Certaines peuvent être modifiées, mais il est toujours bon de conserver un standard d'appellation.

Extension	Description
.a	librairie
.aud	fichier audit Oracle9i
.bdf	fichier X11 de description des polices
.bmp	fichier image X11
.c	source d'un programme C
.ctl	fichier de contrôle de SQL*Loader
.dat	fichier de données ASCII en entrée de SQL*Loader
.dbf	fichier de tablespaces de la base de données
.dmp	fichier provenant d'une exportation, destiné à être manipulé par importation
.doc	fichier texte ascii
.env	shell script servant à fixer des variables d'environnement
.h	fichier header d'un programme C
.jar	fichier d'archive Java (Java archive class)
.l	fichier du <i>man</i> Linux
.lis	fichier obtenu en sortie de scripts SQL*Plus
.log	fichier de trace ou log
.mk	fichier <i>make</i> d'un programme C
.msb	fichier binaire comportant les messages Oracle traduits

Extension	Description
.msg	fichier texte comportant les messages Oracle traduits
.o	module
.ora	fichier de configuration Oracle
.pc	fichier écrit en Pro*C destiné à être transformé en .c puis exécutable par le précompilateur
.pco	idem pour le Pro*Cobol
.sh	fichier shell écrit en Bourne Shell
.sql	script SQL
.sys	fichier shell écrit en Bourne Shell
.tab	script SQL
.trc	fichier trace
.zip	fichier zip (compressé)

NLS : support des langues par Oracle 10g et l'euro

Dans cette annexe :

- définition d'un jeu de caractères ;
- l'utilisation des jeux de caractères ;
- l'impact du jeu de caractères sur les applications ;
- Oracle et le caractère euro.

Cette annexe traite du NLS (*National Language Support*), ou support des langues nationales par Oracle. Les spécificités du stockage du caractère euro dans une base Oracle sont abordées.

Qu'est-ce qu'un jeu de caractères ?

Un jeu de caractères, ou *character set*, définit une table de correspondance entre des caractères et des codes binaires. Dans un jeu de caractères codé sur 8 bits, les valeurs binaires vont de 0 à 255, et un caractère est affecté à chacune de ces valeurs. Cela signifie que si votre ordinateur obéit à un jeu de caractères, dès qu'une touche du clavier est frappée, la valeur binaire du caractère est envoyée à l'ordinateur pour être traitée. Par exemple, la valeur binaire 65 correspond normalement à la lettre A.

Qu'est-ce qu'un jeu de caractères pour la base Oracle ?

Le jeu de caractères (NLS_CHARACTERSET) utilisé par la base de données est figé lors de la création de la base. Il est impossible de le modifier ultérieurement. La base Oracle reçoit des valeurs binaires à stocker et c'est son jeu de caractères qui effectue la correspondance valeur binaire/caractère stocké. On comprend alors pourquoi il ne peut y avoir qu'un seul jeu de caractères pour la base de données.

La valeur du character set est consultable par la vue SYS.PROPS\$. L'ordre SQL suivant détermine le jeu de caractères d'une base de données existante.

```
SQL> select * from sys.props$
```

NAME	VALUE\$	COMMENT\$
-----	-----	-----
DICT.BASE	2	dictionary base tables
version #NLS_LANGUAGE	AMERICAN	Language
NLS_TERRITORY	AMERICA	Territory
NLS_CURRENCY	\$	Local currency
NLS_ISO_CURRENCY	AMERICA	ISO currency
NLS_NUMERIC_CHARACTERS	.,	Numeric characters
NLS_CALENDAR	GREGORIAN	Calendar system
NLS_DATE_FORMAT	DD-MON-YY	Date format
NLS_DATE_LANGUAGE	AMERICAN	Date language
NLS_CHARACTERSET	WE8ISO8859P15	Character set
NLS_SORT	BINARY	Linguistic definition
NLS_NCHAR_CHARACTERSET	WE8MSWIN1252	NCHAR Character set
NLS_RDBMS_VERSION	8.1.5.0.0	RDBMS version for NLS
		parameters
GLOBAL_DB_NAME	ORACLE.WORLD	Global database name
EXPORT_VIEWS_VERSION	7	Export views revision #

Un ordre SQL de création d'une base Oracle avec le jeu de caractères indiqué :

```
create database TEST
  controlfile reuse
  logfile '/u01/oradata/TEST/redo01.log' size 1M reuse,
         '/u01/oradata/TEST/redo02.log' size 1M reuse,
         '/u01/oradata/TEST/redo03.log' size 1M reuse,
         '/u01/oradata/TEST/redo04.log' size 1M reuse
  datafile '/u01/oradata/TEST/system01.dbf' size 10M reuse autoextend on
    next 10M maxsize 200M
  character set WE8MSWIN1252;
```

Qu'est-ce que le jeu de caractères d'un client ?

Le jeu de caractères utilisé par un ordinateur client doit être identique à celui utilisé par le client Oracle. De plus, chaque caractère d'un client doit avoir son correspondant dans le jeu de caractères du serveur. Cela signifie que des clients de langues différentes peuvent accéder à une même base de données, à condition que le jeu de caractères de

chacun des clients coïncide avec celui du serveur. C'est là toute la difficulté du choix d'un jeu de caractères en environnement multilingue : il faut que celui de la base de données soit un « sur ensemble » des jeux de toutes les langues utilisées par les postes clients.

Le jeu de caractères est fixé sur le poste client par la variable `NLS_LANG`, au format : `LANGUAGE_TERRITORY.CHARACTERSET`.

Cette variable est fixée comme une variable d'environnement sous Linux, dans la base de registre du poste client sous Windows.

Sur quoi agit le `LANGUAGE` du jeu de caractères d'un client ?

La partie `LANGUAGE` de la variable d'environnement `NLS_LANG` d'un poste client agit sur la langue des messages utilisés par les outils clients Oracle, sur le nom des jours et des mois, ainsi que sur le format de présentation par défaut des éléments pilotés par le `CHARACTERSET`, si celui-ci n'est pas modifié.

Sur quoi agit le `TERRITORY` du jeu de caractères d'un client ?

La partie `CHARACTERSET` de la variable d'environnement `NLS_LANG` d'un poste client agit sur le nom des jours et des mois, sur le jour qui débute une semaine, sur le symbole de la monnaie, ainsi que sur le format de présentation des dates et des nombres (par exemple : `JJ/MM/YY` ou `MM/JJ/YYYY`).

Des clients de langues diverses peuvent disposer de présentations différentes d'une même donnée. Par exemple :

pour une date:	01-FEV-99	ou	EB/01/99
pour un nombre:	999.999,99	ou	999,999.99
pour une monnaie:	F	ou	\$

Comment gérer les paramètres NLS au niveau de la base de données ?

Pour les paramètres NLS (*National Language Support*), on peut fixer des valeurs par défaut au niveau de la base de données, dans le fichier `init<SID>.ora`. Ces paramètres affectent la base de données, mais pas le poste client. Ainsi, fixer le paramètre `NLS_LANGUAGE` au niveau de la machine hébergeant la base de données ne modifie pas la langue des messages des postes clients qui accèdent à cette base.

Toutes les valeurs `NLS_xx` peuvent être modifiées dans le fichier `initSID.ora`, sauf celles concernant le `CHARACTERSET`.

Comment visualiser les paramètres NLS de la base et d'une session ?

Valeurs par défaut utilisées lors de la création de la base de données :

```
SQL> select * from NLS_DATABASE_PARAMETERS ;
```

PARAMETER	VALUE
NLS_LANGUAGE	AMERICAN
NLS_TERRITORY	AMERICA
NLS_CURRENCY	\$
NLS_ISO_CURRENCY	AMERICA
NLS_NUMERIC_CHARACTERS	.,
NLS_CALENDAR	GREGORIAN
NLS_DATE_FORMAT	DD-MON-YY
NLS_DATE_LANGUAGE	AMERICAN
NLS_CHARACTERSET	WE8MSWIN1252
NLS_SORT	BINARY
NLS_NCHAR_CHARACTERSET	WE8MSWIN1252
NLS_RDBMS_VERSION	9.0.1.0.0

Paramètres utilisés par la variable NLS_LANG du poste client utilisé (sans le jeu de caractères) :

```
SQL> select * from NLS_SESSION_PARAMETERS ;
```

PARAMETER	VALUE
NLS_LANGUAGE	FRENCH
NLS_TERRITORY	FRANCE
NLS_CURRENCY	F
NLS_ISO_CURRENCY	FRANCE
NLS_NUMERIC_CHARACTERS	.,
NLS_CALENDAR	GREGORIAN
NLS_DATE_FORMAT	DD/MM/YY
NLS_DATE_LANGUAGE	FRENCH
NLS_SORT	FRENCH

Paramètres fixés ou modifiés dans le fichier *initSID.ora* :

```
SQL> select * from NLS_INSTANCE_PARAMETERS ;
```

PARAMETER	VALUE
NLS_LANGUAGE	AMERICAN
NLS_TERRITORY	AMERICA
NLS_SORT	
NLS_DATE_LANGUAGE	
NLS_DATE_FORMAT	
NLS_CURRENCY	
NLS_NUMERIC_CHARACTERS	
NLS_ISO_CURRENCY	
NLS_CALENDAR	

Valeurs actuelles des variables NLS (incluant le jeu de caractères) :

```
SQL> select * from V$NLS_PARAMETERS ;
```

PARAMETER	VALUE
NLS_LANGUAGE	FRENCH
NLS_TERRITORY	FRANCE
NLS_CURRENCY	F
NLS_ISO_CURRENCY	FRANCE
NLS_NUMERIC_CHARACTERS	.,
NLS_CALENDAR	GREGORIAN
NLS_DATE_FORMAT	DD/MM/YY
NLS_DATE_LANGUAGE	FRENCH
NLS_CHARACTERSET	WE8MSWIN1252
NLS_SORT	FRENCH
NLS_NCHAR_CHARACTERSET	WE8MSWIN1252

Valeurs qui peuvent être utilisées pour les paramètres CHARACTERSET, LANGUAGE, TERRITORY, SORT (cette liste est écourtée, car elle comporte près de quatre cents valeurs) :

```
SQL> select * from V$NLS_VALID_VALUES ;
```

PARAMETER	VALUE
LANGUAGE	AMERICAN
LANGUAGE	GERMAN
LANGUAGE	FRENCH
LANGUAGE	CANADIAN FRENCH
LANGUAGE	SPANISH
...	
TERRITORY	AMERICA
TERRITORY	UNITED KINGDOM
TERRITORY	GERMANY
TERRITORY	FRANCE
TERRITORY	CANADA
TERRITORY	SPAIN
TERRITORY	ITALY
...	
CHARACTERSET	US7ASCII
CHARACTERSET	WE8DEC
CHARACTERSET	WE8HP
CHARACTERSET	WE8ISO8859P1
CHARACTERSET	EE8ISO8859P2
...	
CHARACTERSET	WE8MSWIN1252
...	
SORT	BINARY
SORT	WEST_EUROPEAN
SORT	XWEST_EUROPEAN
SORT	FRENCH

```
SORT          NORWEGIAN
SORT          SWEDISH
...
378 ligne(s) sélectionnée(s).
```

Comment s'assurer que des paramètres autres qu'anglo-saxons peuvent être affichés ?

1. Créez une base de données avec un jeu de caractères incluant l'ensemble des langues que vous souhaitez afficher. **Les valeurs recommandées sont WE8MSWIN1252 pour Windows et WE8ISO8859P15 pour Linux / Unix.** Ces valeurs conviennent pour le français, l'anglais et la majorité des langues européennes.
2. Assurez-vous que le poste client a la capacité d'afficher les caractères que vous souhaitez.
3. Attribuez à la variable NLS_LANG à la valeur FRENCH_FRANCE.WE8ISO8859P15 sur un poste Linux ou FRENCH_FRANCE.WE8MSWIN1252 sur un poste Windows, pour avoir les messages en français. NLS_LANG est une variable d'environnement sous Linux, une valeur dans la base de registre sous Windows NT/2000/XP (voir *Oracle et l'euro* à la fin de ce chapitre).

En quoi le jeu de caractères influe sur les Export/Import ?

Export/Import sont des outils standards d'Oracle abordés au chapitre 18, *Les outils d'Oracle 10g*. Import et Export, comme SQL*Plus, sont des programmes affectés par la valeur du paramètre NLS_LANG de l'environnement client. À ce titre, ils transforment le jeu de caractères en provenance de la base de données pour le rendre conforme à celui du poste client indiqué par la variable NLS_LANG. Le jeu de caractères du poste client ayant réalisé une exportation est indiqué dans le fichier résultant de cette exportation.

Lors d'une importation, si le jeu de caractères du fichier à importer est différent de la valeur NLS_LANG de l'environnement utilisateur, le programme d'Import effectue une conversion vers le jeu de caractères précisé par le NLS_LANG, puis, si nécessaire, vers le jeu de caractères de la base de données. Des problèmes difficiles à détecter peuvent se produire si vous effectuez des Export/Import avec des jeux de caractères différents.

Pour éviter tout problème : l'Export/Import doivent toujours se faire avec le même jeu de caractères que celui de la base de données. C'est une source d'erreur très fréquente. Elle est difficile à détecter (l'Export et l'Import se déroulent correctement) mais les conséquences peuvent être catastrophiques : perte de tous les caractères accentués. Prudence si vous utilisez l'Export comme moyen de sauvegarde !

*En quoi le jeu de caractères influe sur SQL*Loader ?*

SQL*Loader est un outil standard d'Oracle abordé au chapitre 18, *Les outils d'Oracle 10g*. Il permet de charger des fichiers texte de tous formats dans la base Oracle. C'est particulièrement intéressant dans les phases de reprise de données existantes d'un autre système ou d'alimentation d'un infocentre.

Lors du chargement du fichier dans la base, vous pouvez indiquer à SQL*Loader le jeu de caractères d'origine du fichier texte. C'est très utile et très performant. Par exemple, les données de production se trouvent sur un mainframe IBM qui les encode au format FEBCDIC297 et la base de données est au format WE8ISO8859P15. Comment effectuer la conversion entre les deux jeux de caractères ? Il suffit de transférer en mode binaire les données sur la machine hébergeant la base et d'indiquer à SQL*Loader que le jeu de caractères du fichier en entrée est FEBCDIC297. La conversion sera assurée par SQL*Loader. Un exemple est présenté au chapitre cité ci-dessus.

Comment s'effectue la conversion entre jeux de caractères ?

Lorsque des caractères sont convertis d'un jeu à un autre, ils doivent exister dans les deux jeux. Si ce n'est pas le cas, un caractère de substitution est affecté. Bien souvent, les caractères de substitution sont prévus entre différents jeux, par exemple entre les divers caractères européens. Si un cas inconnu se présente, un caractère de remplacement unique, le « ? » par exemple est mis à la place du caractère non traduisible. La conversion inverse, du caractère de substitution vers le caractère d'origine, est alors impossible.

Qu'est-ce qu'un jeu de caractères multi-byte ?

Un jeu de caractères multi-byte permet de stocker des caractères non ASCII dans une base de données, comme les caractères japonais, kanjis ou arabes. Oracle autorise la création de bases avec des jeux de caractères multi-bytes comme UTF8 ou Unicode.

Faites attention, comme les caractères ne sont plus codés sous un byte mais deux, le passage d'une base d'un mode vers l'autre peut faire doubler la taille de la base de données !

Quel jeu de caractères choisir pour les architectures Internet ?

Un jeu de caractères UNICODE, comme UTF-8 ou UTF-16, est nécessaire pour utiliser des standard tels que XML, JAVA, JavaScript, LDAP. Cela permet de transporter les données entre différents matériels, logiciels, sans corruption de données difficiles à détecter.

Comment afficher les messages Oracle en français ?

Choisissez une langue pour l'installation d'Oracle. Assurez-vous que ce sera celle des produits ajoutés ultérieurement. Dans le cas de l'installation de la base de données, sous Linux ou Windows, il est possible de prévoir les messages pour une ou plusieurs langues. Pour l'installation des clients, afin d'éviter que trop de messages dans des langues inutiles à l'utilisateur soient installés, seuls les messages anglais et ceux de la langue choisie sont installés. L'introduction des messages dans une langue qui n'existe pas sur le poste nécessite la réinstallation du produit dans la langue choisie.

Les messages sont contenus dans des fichiers de type : *<abréviation du produit><abréviation de la langue>.msb*.

Pour les messages en provenance de la base de données, positionnez la variable NLS_LANGUAGE du fichier *initSID.ora*. Par exemple NLS_LANGUAGE = FRENCH donnera les messages en français. Sur le poste client, si la variable locale NLS_LANG n'indique aucune langue, celle choisie par NLS_LANGUAGE est employée. Si NLS_LANG indique une autre langue, alors la valeur par défaut choisie par NLS_LANGUAGE est ignorée et les messages seront affichés dans la langue précisée par NLS_LANG. Si les fichiers message n'existent pas sur le poste, une erreur est indiquée.

Comment modifier la présentation par défaut des dates ?

Les paramètres NLS_DATE_FORMAT et NLS_LANGUAGE présents dans le fichier *init<SID>.ora* précisent un format d'affichage par défaut des dates. La valeur de NLS_LANG du poste client peut, elle aussi, influencer l'affichage par défaut. Il est toujours possible de forcer l'affichage d'une date en jouant sur des conversions. Par exemple :

```
SQL> select to_char(sysdate, ' DD-month-YYYY ' ) from dual ;
SQL>
SQL> 07-novembre-2001
```

En France, nous vous conseillons la valeur par défaut suivante : NLS_LANGUAGE =FRENCH.

Ce tableau résume quelques cas possibles, avec le résultat par défaut obtenu par un ordre sous SQL*Plus, `select sysdate from dual`. Cela vous aidera à choisir le paramétrage approprié à votre situation.

NLS_DATE_FORMAT	NLS_LANGUAGE	NLS_LANG (environnement client)	select sysdate from dual
sans	AMERICAN	Sans	07-nov-01
"DD-month-YYYY "	AMERICAN	Sans	07-november-2001
sans	AMERICAN	FRENCH_FRANCE.WE8ISO8859P15	07/11/01
"DD-month-YYYY "	AMERICAN	FRENCH_FRANCE.WE8ISO8859P15	07-novembre-2001
sans	FRENCH	sans	07-nov-01
"DD-month-YYYY "	FRENCH	Sans	07-novembre-2001
sans	FRENCH	FRENCH_FRANCE.WE8ISO8859P15	07/11/01
"DD-month-YYYY "	FRENCH	FRENCH_FRANCE.WE8ISO8859P15	07-novembre-2001

Comment modifier la présentation par défaut des chiffres ?

Les paramètres NLS_CURRENCY et NLS_LANGUAGE présents dans le fichier *init<SID>.ora* donnent un format d'affichage par défaut des chiffres et des symboles monétaires (F, \$, £,...). La valeur de NLS_LANG du poste client influe elle aussi sur l'affichage par défaut.

Vous pouvez formater un ordre SQL sous SQL*Plus par :

```
select to_char( sum(sal), '999G999D99') from scott.emp ,
```

avec :

G : séparateur de groupe (, pour les US . pour l'Europe à l'exclusion

→des Britanniques)

D : séparateur des décimales (. pour les US , pour l'Europe à l'exclusion

→des Britanniques)

Oracle et l'euro

On peut toujours stocker des valeurs numériques dans une base Oracle, qu'elles soient exprimées en degrés, en kilos, en francs ou en euros. Le stockage des valeurs numériques est parfaitement maîtrisé.

Le seul problème possible concerne le stockage du caractère euro (€) dans des zones textuelles de la base de données. Comme tout caractère possède une représentation graphique (visuelle) et un codage binaire interne, il faut que celui de l'euro existe à la fois dans les deux représentations. L'environnement graphique (visuel) doit être capable de l'afficher et le jeu de caractères de la base doit incorporer son correspondant binaire.

Si l'environnement graphique essaie d'enregistrer le caractère euro dans une base de données dont le jeu de caractères ignore ce caractère, un caractère de substitution remplace le caractère euro qui est alors perdu. C'est déjà le cas pour les caractères accentués.

L'environnement graphique et la base de données peuvent figurer dans des jeux de caractères différents s'il existe une correspondance du caractère euro entre les deux.

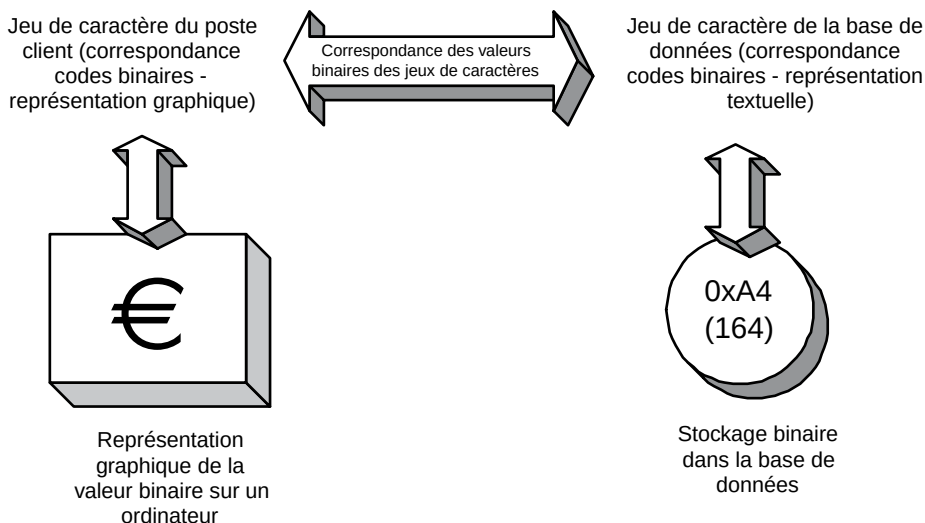


Figure A5-1

Correspondance des jeux de caractères pour l'euro

Ce paragraphe décrit comment paramétrer une base de données Oracle pour y stocker le caractère semi-graphique euro (€). Les objectifs sont :

- comprendre pourquoi le stockage du caractère euro pose des problèmes ;
- créer une base Oracle avec un jeu de caractère correct ;
- pouvoir insérer/modifier/visualiser le caractère euro depuis un poste Windows ;
- échanger des données entre différents systèmes sans perdre ni altérer le caractère euro.

Pourquoi le stockage de l'euro pose-t-il problème ?

En informatique, toutes les informations sont codées en une suite de chiffres binaires. Des tables de correspondance sont indispensables pour effectuer les traductions entre ces données binaires et leur « valeur » représentative. Cette traduction doit être bilatérale et unique : chaque valeur doit correspondre à une seule représentation binaire et inversement.

Une fois qu'un jeu de caractères a été défini, on doit s'assurer qu'il existe des « traductions », c'est-à-dire que la transformation d'un jeu de caractères vers un autre (et inversement) est possible. Par exemple, les échanges de fichiers de données entre différents systèmes d'exploitation doivent permettre ces traductions, sans perte ni altération de données.

Si les échanges pour les données de l'alphabet sont stables car existant depuis longtemps, la récente adoption du sigle euro (€) pose un problème pour les tables de correspondance : elles doivent être mises à jour pour inclure ce nouveau sigle. Si les tables de traduction ne sont pas compatibles, le caractère euro (€) peut être « perdu » ou altéré lors d'une traduction.

Suivant leurs besoins et leur histoire, les constructeurs ou éditeurs de logiciels ont adopté leurs propres tables de correspondance. Pour les unifier et permettre des échanges, l'ISO (International Organization for Standardization) a défini des standards. Oracle, pour définir les jeux de caractères utilisés, s'appuie sur ces standards. En 1999, l'ISO a défini le jeu de caractères ISO 8859-15, qui regroupe les caractères spécifiques utilisés par les langues de l'Europe de l'Ouest et code le sigle euro (€) sous la valeur binaire 164.

L'affichage graphique de l'euro

Le problème de traduction se complique dès que l'on parle de l'affichage, c'est-à-dire de la représentation graphique de données binaires. Une même valeur binaire stockée sur disque peut avoir une représentation graphique différente suivant le programme informatique utilisé pour l'afficher. Cela signifie uniquement que des tables de correspondance différentes sont utilisées pour réaliser l'affichage de la donnée binaire. C'est le cas de l'application *notepad* sous Windows qui est incapable d'afficher le caractère euro (€) alors que *Microsoft Word* et la majorité des applications sous Windows en sont capables. De même, *SQL*Plus* sous Windows qui utilise le même jeu de caractères d'affichage graphique que *notepad* est lui aussi incapable d'afficher le sigle euro (€), alors que les

autres applications Oracle (Forms, Reports, Oracle Applications...) peuvent parfaitement afficher le sigle euro (€).

Créer une base de données de test

Avant de tester l'affichage euro, il faut disposer d'une base de données Oracle compatible euro. L'assistant graphique *dbassist* permet de créer une base de données de test avec un jeu de caractères particulier. À l'issue des écrans de l'assistant, un script SQL pouvant être exécuté est généré. Il doit contenir le jeu de caractères WE8ISO8859P15 qui est compatible euro :

```
CREATE DATABASE "test"
...
CHARACTER SET WE8ISO8859P15
....
```

WE8ISO8859P15 est une convention utilisée par Oracle qui signifie : « Western European 8-bit ISO 8859 Part 15 character set ». (Seuls quelques jeux de caractères compatibles euro peuvent être utilisés. Pour les connaître, consultez la documentation Oracle *National Language Support Guide* référence A67789-01).

Si vous disposez d'une base de données existante, vérifiez avec quel jeu de caractères elle a été créée :

```
SELECT * FROM sys.props$
WHERE name LIKE 'NLS%' ;
```

Le résultat de cet ordre SQL sera :

NAME	VALUE\$	COMMENT\$
-----	-----	-----
NLS_LANGUAGE	AMERICAN	Language
NLS_TERRITORY	AMERICA	Territory
NLS_CURRENCY	\$	Local currency
NLS_ISO_CURRENCY	AMERICA	ISO currency
NLS_NUMERIC_CHARACTERS	.,	Numeric characters
NLS_CHARACTERSET	WE8ISO8859P15	Character set
NLS_CALENDAR	GREGORIAN	Calendar system
NLS_DATE_FORMAT	DD-MON-RR	Date format
NLS_DATE_LANGUAGE	AMERICAN	Date language
NLS_SORT	BINARY	Linguistic definition
NLS_TIME_FORMAT	HH.MI.SSXXFF AM	Time format
NLS_DUAL_CURRENCY	\$	Dual currency symbol
NLS_COMP	BINARY	NLS comparison
NLS_NCHAR_CHARACTERSET	WE8ISO8859P15	NCHAR Character set
NLS_RDBMS_VERSION	8.1.7.0.0	RDBMS version for NLS parameters
...		

Comment ne pas se perdre ?

Une donnée binaire peut être traduite lors d'un échange entre deux systèmes ou visualisée de façon différente par deux programmes. Comment éviter de se perdre ? Comment comprendre ce qui se passe ? L'unique solution est de toujours revenir au codage binaire ASCII ou Hexadécimal de la donnée. Pour comprendre la démarche suivie, utilisons une table de test :

```
CREATE TABLE TableEuro ( libelle varchar2(20), valeur varchar2(1));

INSERT INTO TableEuro VALUES ('A majuscule','A');
INSERT INTO TableEuro VALUES ('B majuscule','B');
INSERT INTO TableEuro VALUES ('C majuscule','C');
INSERT INTO TableEuro VALUES ('Euro',CHR(164));
INSERT INTO TableEuro VALUES ('Euro (incorrect)',CHR(128));
COMMIT;
```

On regarde le contenu de la table par l'ordre SQL suivant qui visualise le libellé, affiche le caractère et indique quelle est la valeur binaire stockée dans la base :

```
SELECT libelle, valeur, ASCII(valeur) FROM TableEuro;
```

LIBELLE	V	ASCII(VALEUR)
A majuscule	A	65
B majuscule	B	66
C majuscule	C	67
Euro	€	164
Euro (incorrect)	ï	128

L'exemple précédent a été réalisé à partir de SQL*Plus qui n'est pas capable de visualiser le caractère euro. Le code euro correct (164) et un code euro incorrect (128) ont sous SQL*Plus une représentation graphique identique mais incorrecte. Cet exemple montre clairement qu'il ne faut pas se fier à la seule représentation graphique d'un caractère. Il faut toujours identifier le code binaire stocké dans la base. Avec un poste client correctement paramétré, on obtient :

Le support Oracle propose un contournement qui permet d'afficher le sigle _ dans SQL*Plus (voir note technique n°132453.1)

Figure A5-2

*Visualisation correcte
du sigle euro*

Libelle	Valeur	Code Binaire
A majuscule	A	65
B majuscule	B	66
C majuscule	C	67
Euro	€	164
Euro (incorrect)	ï	128

Le paramétrage du poste client

Les systèmes d'exploitation de Microsoft utilisent leurs propres jeux de caractères. Celui utilisé en langue française est le code page Microsoft 1252 et le caractère euro est stocké sous la valeur binaire 128. Il faut donc paramétrer l'environnement Oracle sous Windows pour que le caractère euro Windows soit converti et stocké dans la base de test sous la valeur binaire 164. Cette conversion doit fonctionner aussi pour l'insertion de données, leur modification ou leur lecture. Pour cela, il faut changer la variable Oracle NLS_LANG qui influe sur les caractères nationaux (langue des messages d'erreur, format des chiffres et des dates...).

Sous Windows 2000, cette valeur est située dans la base de registre sous Hkey_Local_Machine/Software/Oracle. Il convient de la remplacer la valeur de la clé NLS_LANG par NLS_LANG=FRENCH_FRANCE.WE8MSWIN1252 qui signifie pour Oracle « MS Windows Code Page 1252 8-bit West European ». La variable NLS_LANG peut aussi se retrouver dans d'autres ruches du registre sous la même racine. Par exemple, si deux ORACLEHOME existent, on modifiera NLS_LANG sous :

```
..\ORACLE_ALL_HOMES\ID0
..\ORACLE_ALL_HOMES\ID0
..\ORACLE\HOME0
..\ORACLE\HOME1
...
```

Note : on peut changer automatiquement toutes les clés de registre à l'aide de l'utilitaire « regfind » du « Kit de ressources techniques Windows NT » de Microsoft. La commande est alors :

```
regfind -p "HKEY_LOCAL_MACHINE\SOFTWARE\ORACLE"
          "AMERICAN_AMERICA.WE8ISO8859P15"
-r "FRENCH_FRANCE.WE8MSWIN1252"
```

Si la valeur NLS_LANG diffère de celle préconisée, il est parfois possible d'insérer le caractère euro et de le visualiser, mais la conversion entre le poste client et la base est erronée et une valeur binaire incorrecte est utilisée pour stocker le caractère euro. **Dans ce cas, le sigle euro sera perdu dans les opérations d'échange de données intersystèmes.**

Dans le cas suivant, la valeur incorrecte NLS_LANG=AMERICAN_AMERICA.WE8ISO8859P15 affiche le sigle euro pour un code binaire incorrect. Le sigle euro s'affiche mais le paramétrage du poste client est erroné :

Figure A5-3

Visualisation incorrecte
du sigle euro

Libelle	Valeur	Code Binaire
A majuscule	A	65
B majuscule	B	66
C majuscule	C	67
Euro	€	164
Euro (incorrect)	€	128

Conclusion

Une base de données créée sous Linux/Unix avec le jeu de caractères WE8ISO8859P15 ou sous Windows avec WE8MSWIN1252, des ordres SQL simples pour vérifier la valeur binaire de caractères dans la base et une valeur adaptée de la variable d'environnement NLS_LANG vous assurent du respect des normes internationales et une utilisation sans souci du caractère €.

6

Les logiciels livrés

Oracle 10g Universal Installer (OUI) permet d'installer les logiciels Oracle 10g pour Windows. Dans la vaste palette des logiciels Oracle 10g, il est important de connaître l'utilité de chacun. Cette liste commentée a pour but de vous présenter succinctement les logiciels Oracle fournis avec Oracle 10g pour Windows. Ils sont classés par ordre alphabétique pour permettre une recherche facile.

Produit	Fonction
Advanced Queueing	Support de l'API Advanced Queueing.
Advanced Replication	Support de la réplication avancée. Permet de diffuser des modifications de données d'une base vers une autre, de manière synchrone ou asynchrone.
Character Set Scanner	Utilitaire permettant de valider un jeu de caractères avant de le migrer vers un autre.
Assistant Common Files (installé avec divers assistants)	Ensemble de fichiers nécessaires à l'exécution des assistants Oracle écrits en Java comme Oracle Database Configuration Assistant ou Oracle Net Services Assistant.
Enterprise JavaBeans	Architecture pour développer des composants transactionnels distribués en Java.
Generic Connectivity	Passerelles permettant d'accéder à d'autres bases de données qu'Oracle. Une partie de cette technologie, commune à plusieurs bases, est maintenant embarquée directement dans la base.
Java Runtime Environment (JRE)	Basé sur un environnement d'exécution Java, il permet de faire fonctionner le programme Oracle Installer. Pour développer des applications Java, il faudra installer un JDK de votre choix, non livré par Oracle.
Object Type Translator	Cet assistant graphique permet de créer ou de modifier des fichiers de type INTYPE. Ces fichiers seront utilisés par l'OTT (<i>Object Type Translator</i>) pour réaliser des programmes d'accès aux objets en OCI (<i>Oracle Call Interface</i>) ou en C++.

Produit	Fonction
Oracle Advanced Security	Logiciel comprenant différents modules destinés à renforcer la sécurité d'Oracle (cryptage, authentification renforcée, liens Kerberos...). Permet d'atteindre un niveau de sécurité quasi militaire.
Oracle Call Interface (OCI)	C'est l'interface de programmation de bas niveau la plus proche d'Oracle. Elle permet de faire des appels directs aux fonctions Oracle à partir du langage C ou C++.
Oracle Connection Manager	Permet d'assurer une fonction de routage pour Oracle Net. Très peu utilisé.
Oracle Data Migration Assistant	Assistant qui permet de migrer les versions précédentes d'Oracle vers la version actuelle, de la 7.3.4 à la 8.1.7.
Oracle Database Configuration Assistant (installé avec Oracle 10g Server)	Cet assistant graphique est d'une grande utilité dans la création ou la suppression d'une base Oracle 10g. Écrit en Java, il est détaillé au chapitre 14, <i>Création d'une base Oracle 10g</i> .
Oracle DBA Management Pack	Ensemble de logiciels liés à Oracle Enterprise Manager qui permettent une administration graphique des bases Oracle.
Oracle DBA Studio (installé avec Oracle DBA Management Pack)	Utilitaire graphique très précieux pour une administration quotidienne des bases Oracle.
Oracle Documentation	Documentation Oracle au format HTML ou Adobe Acrobat (PDF). C'est une aide précieuse à utiliser sans retenue.
Oracle Dynamic Services	Infrastructure Java de gestion et de déploiement de services Internet.
Oracle Enterprise Login Assistant	En liaison avec Oracle Wallet Manager, il offre une fonction de connexion unique à de multiples applications (sigle sign-on).
Oracle Enterprise Manager	Suite intégrée d'outils graphiques pour administrer tous les aspects des bases Oracle, tant locales que distantes, à partir d'une console unique. Oracle Enterprise Manager regroupe une console graphique, un référentiel accessible depuis plusieurs consoles, des agents intelligents chargés d'exécuter des travaux et d'assurer la surveillance de paramètres déterminés, ainsi que de nombreux composants dédiés à des actions précises. Il est largement abordé au chapitre 25, <i>Oracle Enterprise Manager</i> .
Oracle Enterprise Manager Configuration Assistant	Aide à la configuration du référentiel d'Oracle Enterprise Manager.
Oracle Enterprise Manager Console	Console graphique de pilotage d'Oracle Enterprise Manager. C'est l'application qui permet d'accéder à tout le potentiel d'Enterprise Manager. Elle offre : • l'accès aux applications intégrées ; • la gestion simultanée de plusieurs bases ; • l'ordonnancement des tâches sur les bases ou les machines ; • l'intégration de la gestion des nouveaux logiciels Oracle.
Oracle Enterprise Manager Integrated Applications	Ces applications gèrent les bases au quotidien : utilisateurs, schémas, espace disque, etc. Elles sont accessibles depuis la console ou peuvent être lancées séparément.
Oracle Enterprise Manager Management Packs	Ces options facturées sont centrées sur des aspects précis : • Diagnostic Pack ; • Tuning Pack ; • Change Management Pack ; • Management Pack for ERP.

Produit	Fonction
Oracle Enterprise Manager Quick Tour	Présentation d'Oracle Enterprise Manager (OEM) au format HTML qui décrit son fonctionnement sans avoir besoin d'installer OEM.
Oracle Enterprise Manager Web Site	Permet à des administrateurs Oracle de lancer la console d'Oracle Enterprise Manager depuis un navigateur.
Oracle HTTP Server	Serveur Web HTTP d'origine Apache. Il est livré préconfiguré pour permettre l'accès à la console d'Oracle Enterprise Manager depuis un simple navigateur.
Oracle Integration Server	Installe une suite de composants techniques destinés à faciliter l'intégration de logiciels ERP.
Oracle Intelligent Agent	Agent installé sur un serveur. En liaison avec Oracle Enterprise Manager, il surveille l'exécution des actions qui lui sont confiées (ordonnancement, surveillance d'événements), il collecte et remonte au référentiel des informations (CPU, IO, compte rendu d'exécution...) destinées à des modules complémentaires d'Enterprise Manager. Il existe des agents pour de nombreux systèmes d'exploitation différents.
Oracle interMedia	Les produits de la famille Oracle <i>interMedia</i> servent à interroger et manipuler dans Oracle toutes sortes de données non textuelles : images, sons, vidéos. Les possibilités proposées sont spécifiques et très puissantes.
Oracle Internet Directory	Annuaire Oracle au format LDAP pour centraliser les utilisateurs, la configuration d'Oracle Net et les accès aux bases et applications. Il doit être installé dans une base Oracle dédiée.
Oracle Internet Directory Configuration Assistant	Assistant d'aide à la création du schéma d'Oracle Internet Directory dans une base.
Oracle Internet Directory Client Toolset	Composant Windows qui effectuent le lien avec Oracle Internet Directory.
Oracle Internet Directory Manager	Utilitaire Java d'administration d'Oracle Internet Directory.
Oracle Internet File System	Oracle iFS (Internet File System) permet l'accès à des documents depuis un compte ftp, le Web, la messagerie, sans passer par un serveur de fichiers et son protocole propriétaire.
Oracle Java Database Connectivity (JDBC)	Le pilote JDBC permet l'accès aux bases Oracle 10g à partir de programmes écrits en Java. L'utilisation de ces classes Java est abordée au chapitre 19, <i>Pilotes JDBC et SQLJ</i> . Le pilote Thin JDBC est téléchargeable, il permet de communiquer avec un serveur Oracle 10g directement à partir d'un poste client qui n'a pas de protocole Net Service installé.
Oracle Java Tools	Suite d'outils et d'utilitaires pour programmer l'environnement Oracle 10g JVM.
Oracle JavaServer Pages	Oracle JSP (JavaServer Pages) sont des pages Web dynamiques créées depuis un serveur d'application Java.
Oracle Label Security	Complément des Virtual Private Databases, Oracle Label Security est une option supplémentaire permettant une gestion très fine des droits d'accès aux données.
Oracle Local Builder	Ce composant simplifie la modification des jeux de caractères locaux. Inutile pour la langue française, parfaitement couverte par Oracle.
Oracle Locator	Composant lié à Oracle Spatial installé avec <i>interMedia</i> .
Oracle Management Server	Composant intermédiaire d'Oracle Enterprise Manager situé entre les consoles d'administration et le référentiel. C'est avec lui que dialoguent les agents intelligents. Les consoles doivent se connecter au Management Server pour administrer les bases, les serveurs et les applications.

Produit	Fonction
Oracle Names	Simplifie l'administration d'un environnement Oracle Net en proposant un serveur de nom. La configuration n'a plus à être distribuée sur les postes clients et sur les serveurs mais se trouve centralisée dans le serveur Oracle Names.
Oracle Net	Oracle Net (anciennement Net8 ou SQL*Net) permet d'établir via un réseau une communication entre une application et une base de données. Ce composant est indispensable pour une communication client-serveur.
Oracle Net Configuration Assistant (installé avec Oracle Net)	Assistant graphique écrit en Java qui permet de réaliser rapidement et simplement la création de vos « alias » Oracle Net (ancien Net8 ou SQL*Net) et la configuration avancée d'options d'Oracle Net.
Oracle Net Listener	Oracle Net Listener est présent uniquement sur le serveur qui assure la communication client-serveur ou serveur-serveur, utilisant un réseau TCP/IP. Surnommé le « Listener » (textuellement, celui qui écoute), il est décrit au chapitre 15, <i>Oracle Net, le middleware Oracle</i> .
Oracle Net Manager	Oracle Net Manager est intégré dans Oracle Enterprise Manager. Il permet aux administrateurs Oracle de surveiller et d'administrer les composants d'Oracle Net.
Oracle OLAP	Oracle OLAP est une extension d'Oracle destinée à la réalisation de cubes de données multidimensionnelles. Maintenant intégré, il était auparavant situé à l'extérieur de la base sous le nom d'Oracle Express.
Oracle Partitioning	Permet une gestion plus fine de l'organisation physique des tables et index volumineux. Utile pour des stockages de données d'un très grand volume.
Oracle Personalization	Produit annexe qui permet de gérer l'intégration d'éléments dans des sites Web, par exemple des bannières publicitaires.
Oracle PL/SQL Gateway	Passerelle Java fonctionnant avec le serveur HTTP d'origine Apache. Le <i>servlet mod_plsql</i> externe à la base permet l'accès direct à des procédures Oracle écrites en PL/SQL dans la base. L'accès peut aussi s'effectuer à partir du <i>servlet mod_ose</i> qui appelle des programmes Java stockés dans la base Oracle 10g.
Oracle Policy Manager	Installé avec Oracle Enterprise Manager, Oracle Policy Manager est un outil graphique d'administration des droits des Virtual Private Databases.
Oracle Real Application Clusters	Cette option permet le fonctionnement d'Oracle dans un environnement matériel de type « cluster ». Dans ce cas, chaque instance s'exécute sur un nœud particulier tout en accédant à la même base de données.
Oracle Real Application Clusters Guard	Cette option d'Oracle Real Application Clusters assure le redémarrage rapide des nœuds en cas de défaillance.
Oracle Required Support Files	Les « Required Support Files » sont des ensembles de fichiers et programmes nécessaires dans certaines applications tierces pour se connecter à des serveurs Oracle 8.x, 7.x.
Oracle 10g Required Support Files	C'est un ensemble de fichiers et de DLL nécessaires pour se connecter à une version Oracle 10g. Ils peuvent être installés à partir de n'importe quelle option d'installation.
Oracle Servlet Engine	Environnement d'exécution de servlets intégré dans la base Oracle 10g. Il intègre un serveur HTTP et permet d'exécuter des Java Server Pages (JSP) et des servlets directement dans la base. Il peut être appelé par le module <i>mod_ose</i> de la passerelle Oracle PL/SQL Gateway.

Produit	Fonction
Oracle Spatial	Permet le stockage et la manipulation de données multidimensionnelles dans une base relationnelle. Utile pour des applications très spécifiques réservées au domaine de la cartographie ou du dessin assisté par ordinateur.
Oracle SQLJ	Format normalisé pour incorporer des ordres SQL dans des programmes Java. Il regroupe les composants : Oracle SQLJ Runtime : Moteur d'exécution utilisé en sortie du précompilateur Java. Oracle SQLJ Translator : Précompilateur Java permettant d'incorporer facilement des ordres SQL dans des programmes Java.
Oracle Syndication Server	Produit annexe qui permet des services d'affiliation ou d'abonnement dans des sites Web.
Oracle Text	Précédemment Oracle <i>interMedia Text</i> . Permet d'utiliser des ordres SQL pour indexer, rechercher et analyser des documents contenus dans une base. De nombreux filtres permettent d'utiliser pratiquement tous les formats de documents.
Oracle Text Manager	Interface graphique intégrée à Oracle Enterprise Manager qui permet d'administrer Oracle Text.
Oracle Ultra Search	Placé au sommet d'Oracle Text, ce moteur de recherche communique aussi avec des fichiers, des serveurs de messagerie et des pages Web.
Oracle Universal Installer	OUI (<i>Oracle Universal Installer</i>) est l'outil graphique écrit en Java qui permet d'installer, de mettre à jour et de supprimer des produits Oracle. Multi environnement, il est conçu pour offrir une interface graphique identique, quel que soit le système d'exploitation utilisé : Unix, Linux, Windows 98/Me/NT/2000/XP. Il peut être utilisé pour réaliser des installations automatiques en exécutant un fichier de commandes préparé.
Oracle Utilities	Utilitaires liés à l'administration des bases Oracle. Certains sont génériques et ne sont pas liés au système d'exploitation : DBVERIFY, Export, Import, Migration, Server*Manager, SQL*Loader, TKPROF, Password Utility Recovery Manager. D'autres concernent uniquement les systèmes d'exploitation de Microsoft : OCOPY, ORADIM. Ils sont détaillés au chapitre 18, <i>Les outils d'Oracle 10g</i> .
Oracle Wallet Manager	Gestion des clés publiques et privées dans une architecture de sécurité très poussée.
Oracle Workflow	Oracle Workflow est un moteur de gestion de processus, qui assure leur définition et leur exécution. Ce composant a été développé pour servir de support à la gestion des processus métiers d'Oracle Applications.
Oracle XML Developer's Kit	Kit complet pour gérer des données au format XML. Il permet toutes sortes de manipulations depuis plusieurs langages : Java, C, C++, PL/SQL. Ce kit en pleine évolution contient actuellement les composants suivants : • un Parser XML pour Java, C, C++ et PL/SQL • un générateur de classes XML pour Java et C++ • un composant Java, le XML Transviewer Bean • un servlet XSQL
Oracle XML SQL Utility	Cet utilitaire est composé de classes Java permettant de présenter au format XML le résultat d'un ordre SQL.
Oracle 10g Advanced Analytic Services – Data Mining	Cette application Java est intégrée dans Oracle 10g. Elle permet de construire des applications de Data Mining. Les opérations mathématiques relatives au Data Mining sont proposées sous forme d'API Java.

Produit	Fonction
Oracle 10g JVM Accelerator	Permet de transformer des classes Java en bibliothèques directement interprétables par le système.
Oracle 10g JVM Enterprise Edition	Environnement Java situé dans une base Oracle. Cet ensemble très complet comprend une JVM (machine virtuelle Java), un ORB (Object Request Broker) compatible CORBA, des pilotes JDBC intégrés, un environnement Enterprise JavaBeans. Cette grande nouveauté introduite dans Oracle8i et améliorée dans Oracle 10g est abordée au chapitre 20, <i>Java et Oracle 10g</i> .
Oracle 10g JVM Servlet Container	Permet à des JSP (JavaServer Pages) d'être exécutées à l'intérieur de la base.
Oracle 10g Server	Logiciels nécessaires au fonctionnement de la base Oracle 10g, version Oracle 10g Standard Edition ou Oracle 10g Enterprise Edition. Les différences de contenu entre les deux versions se limitent à des produits supplémentaires qu'il est possible d'acheter séparément après avoir acquis la version Enterprise.
Pro*C / C++	Offre la possibilité d'exécuter des requêtes SQL depuis des programmes écrits en C/C++.
SQL*Plus	C'est l'interpréteur SQL historiquement le plus connu chez Oracle. Il permet de travailler en SQL et PL/SQL directement sur vos bases Oracle 10g. Indispensable à tout administrateur, il est capable d'exécuter toutes les commandes SQL d'Oracle. C'est SQL*Plus qui remplace Server Manager (<i>svrmgrl</i>) pour démarrer/arrêter les bases. Il est détaillé au chapitre 18, <i>Les outils d'Oracle 10g</i> .
SQL*Plus Worksheet	Utilitaire graphique comparable à SQL*Plus : plus convivial mais légèrement moins puissant.
TCP/IP Protocol Support	Couche logicielle qui assure l'interface entre le protocole réseau TCP/IP et le middleware Oracle Net.
XML Development Kit	Nécessaire à l'exécution d'application XML dans la base Oracle.

Sites Internet et bibliographie

Si les sites Internet sont indispensables pour obtenir des informations techniques précises, charger les dernières versions, accéder à des forums, etc., le recul proposé par les livres et la diversité des sujets traités sont complémentaires. N'oublions pas que le seul outil multimédia réellement portable, utilisable en toute circonstance et en tout lieu... c'est le livre !

Il est maintenant impossible de séparer les sites Internet de la bibliographie. Nous proposons tout un ensemble de sites pour « surfer ». Ils ont été testés par nos soins, mais étant donné la vitesse d'évolution d'Internet, de nouveaux sites ont dû apparaître depuis ! À vous de les découvrir...

Le site de l'auteur

Le site accompagnant ce livre où vous trouverez les dernières nouveautés Oracle, des ressources techniques et une liste de liens remise à jour :

<http://perso.club-internet.fr/gbriard>

Le site de Digora

Les chapitres dédiés à Java, XML et Oracle Portal ont été écrits par la société Digora (www.digora.com), SSII spécialisée en conseil, formation et support autour de l'ensemble des produits Oracle :

www.digora.com

Recherches de livres techniques

À tout seigneur tout honneur, le site d'Eyrolles, l'éditeur de ce livre :

www.eyrolles.com

Un autre éditeur de choix, Oracle Press (Osborne/McGraw-Hill) publie la collection officielle d'Oracle. Les titres sont de valeur inégale :

www.osborne.com/oracle/index.htm

Pour les recherches bibliographiques, nous vous conseillons un certain nombre de librairies spécialisées en informatique dont les sites proposent une recherche des ouvrages par mots-clés. Parmi ces librairies, en France, il faut citer : *Le Monde en Tique* et *Infothèque* :

www.lmet.com

www.infotheque.com

Un site Web uniquement consacré aux livres traitant d'Oracle :

www.oracle-home.com

Aux États-Unis, la plus grande bibliothèque du Monde en ligne ; tous les sujets y sont traités :

www.amazon.com

Voici une autre librairie spécialisée en informatique :

www.cbooks.com

N'oubliez pas vos libraires locaux, dont certains ont des rayons spécialisés en informatique.

Sites Internet créés par Oracle

Ces sites sont la propriété d'Oracle. Ils traitent de sujets divers. Le premier est l'incontournable site officiel. C'est à partir de lui que sont menées nombre d'opérations, dont le téléchargement de certaines versions d'évaluation :

www.oracle.com

Sites Oracle techniques

Deux sites techniques sont intéressants. Le premier est le support « Metalink » d'Oracle. Vous devez fournir votre numéro de client Oracle (sous contrat de support) afin de créer un compte d'accès personnalisé. Des forums techniques, avec des animateurs du support Oracle, fournissent des réponses à vos questions :

<http://metalink.oracle.com>

Technet est le site Oracle orienté technique qui permet de télécharger les dernières versions des logiciels, d'accéder à des forums techniques, des bibliothèques d'exemples, etc. : un must ! Son accès est actuellement gratuit : créez vite un compte d'accès ! Un lieu construit par et pour des développeurs :

<http://technet.oracle.com>

La revue « Oracle Magazine » propose des articles souvent intéressants ; son abonnement est gratuit. Profitez-en ! De plus, l'intégralité de la revue ainsi que les anciens numéros sont consultables en ligne :

www.oracle.com/oramag

Sites Oracle généraux

Si vous êtes partenaire d'Oracle ou si vous cherchez un progiciel répondant à vos besoins et reposant sur la technologie Oracle, vous pouvez consulter les sites suivants :

<http://alliance.oracle.com>

<http://solutions.oracle.com>

Si l'aventure Oracle vous tente :

<http://jobs.oracle.com>

Sites Oracle locaux et affiliés

Les filiales d'Oracle proposent des sites Internet dans différentes langues. Ces sites sont parfois d'une qualité d'information très relative. Par exemple, le site français souffre de la comparaison avec son homologue britannique :

www.oracle.fr

www.uk.oracle.com

Voici le site français des utilisateurs Oracle. Espérons que le contenu des sessions du club soit plus fourni que son site Web :

www.oracle.fr/aufo.html

www.ioug.org

www.eoug.com

Les sites externes techniques

Les sites suivants sont incontournables : consultez-les absolument !

www.onwe.co.za/frank/faq2.htm

www.jlcomp.demon.co.uk

www.oriola.com

www.oracle-base.com

www.dbazine.com/ch_oracle.html

www.orakle.com

www.orsweb.com

www.oracletuning.com

www.coxee.com

www.evdbt.com

www.bf.rmit.edu.au/~orafaq/contents.html

www.oracleguru.com/scripts.htm

Le site LazyDBA possède (comme son nom l'indique...) de nombreux scripts SQL utiles :

www.lazydba.com

« Oracle Magazine » présente souvent des scripts, retrouvez-les sur le Web :

www.oracle.com/oramag/code/index.html

Un site réservé aux « fans » d'Oracle :

www.orafans.com

Un site est dédié uniquement aux sauvegardes Oracle ainsi qu'aux outils qui peuvent s'y rattacher :

www.backupcentral.com/oraback.html

Interfacer intelligemment Oracle et Excel : un spécialiste peut vous y aider :

www.rci-info.fr/rcitools/

Les sites Windows-Oracle

www.ipass.net/~davesisk/oont.htm

www.orafaq.com/faq2.htm

De bonnes informations Oracle - Windows sur ce site :

www.orapub.com

Une mine de liens sur ce site :

web.singnet.com.sg/~petermag

Les sites Linux-Oracle

L'excellent site de Suse est à la pointe de l'actualité Oracle et il propose une liste de diffusion très active à laquelle je vous conseille de vous abonner. Félicitations à Michaël Haseinstein pour son excellent travail !

www.suse.com

Un autre site, mine d'informations techniques :

www.akadia.com

Pour installer Oracle sur Linux, rendez-vous sur le site :

<http://jordan.fortwayne.com/oracle>

Voici une liste de diffusion (placez SUBSCRIBE ORACLE-LINUX dans votre message) :

listguru@facility.com

Freewares, sharewares et logiciels libres

Ce ne sont pas forcément des logiciels au rabais. Certains sont superbes et très bien réalisés ! Des utilitaires accédant à Oracle apparaissent sous Linux. Pour commencer, nous vous proposons une suite d'outils Linux assez bien conçus :

<http://oracle.mattshouse.com>

Pour découvrir d'autres utilitaires sous Linux, utilisez oracle comme critère de recherche sur le site :

www.freshmeat.net

Parmi de nombreux exemples, le site suivant propose un outil d'administration des bases via des pages HTML utilisant APACHE et PHP :

<http://phporacleadmin.org>

L'outil YAST (Yet Another SQL*Plus replacement)

<http://sourceforge.net/projects/yasql/>

Voici un excellent outil sous Windows, bien qu'il soit nécessaire de le mettre à jour régulièrement (durée de « vie » limitée) :

www.toadsoft.com

Les sites suivants proposent des utilitaires Oracle sous Windows :

www.cais.com/oic/prod/oicora.html

www.dbcool.com

www.benthicsoftware.com/

Sites proposant un large choix de logiciels autour de Windows 95 et Windows NT :

www.geocel.com

www.tucows.com

www.windows95.com

Les forums de discussion

Un certain nombre de forums traitent d'Oracle. Même s'il existe un forum plus ciblé, il est parfois préférable de rester dans un forum « généraliste », plus animé. N'oubliez pas les forums du site *technet.oracle.com* :

[news :comp.databases](#)

[news :comp.databases.oracle.server](#)

[news :comp.databases.oracle.tools](#)

[news :comp.databases.oracle.misc](#)

Les listes de diffusion

Vous pouvez vous inscrire à des listes de diffusion (mailing lists) dédiées aux produits Oracle. Vous trouverez la référence de ces listes sur les sites suivants :

www.onwe.co.za/frank/faq.htm

www.saraswati.com

La liste dédiée à Developer2000 (le L4G Oracle) est très satisfaisante. Pour y souscrire, envoyez un mail à : *listserv@kbs.net* contenant le texte suivant : SUBSCRIBE oracle-l votre_email

Comparer les bases de données

On apprend beaucoup en observant les autres :

www.tpc.org

www.microsoft.com

www.informix.com

www.sybase.com

www.ibm.com/db2

Et sous Linux, n'oubliez pas :

www.mysql.com

Devenir un DBA certifié

Comment s'entraîner, progresser, ou tout simplement se poser des questions :

www.reducation.oracle.com/certification

www.revealnet.com

www.oracle-users.com

Java, SQLJ et JDBC

Java et ses interfaces d'accès aux bases de données vous intéressent ? Le site incontournable est celui de Sun, l'inventeur de Java, mais il en existe beaucoup d'autres :

www.sun.com

<http://java.sun.com>

www.javasoft.com

<http://java.apache.com>

Gamelan est un site impressionnant par la diversité des ressources techniques proposées. De très nombreux exemples téléchargeables sont proposés :

www.gamelan.com

Annuaire de sites Java : on trouve de très nombreux pointeurs vers des sites dédiés à Java (spécifications du langage, tutoriels, exemples d'applets ou de Java Beans, outils de développement, adresse des groupes de news ainsi que des mailing lists).

www.teamjava.com/links/

www.javaworld.com/javaworld/jw-jumps.html

Liste exhaustive des API Java avec date de disponibilité et possibilité de téléchargement des spécifications complètes et des JDK correspondants.

www.javasoft.com/products/api-overview/

Pour avoir plus d'informations sur les accès aux bases de données distantes à partir de Java :

splash.javasoft.com/databases/

splash.javasoft.com/databases/jdbc/jdbc.drivers.html

Groupes de news dédiés à JDBC :

[news :comp.lang.java.databases](http://news.comp.lang.java.databases)

Outils de développement Web/bases de données ou Java ; « DBMS Magazine/Internet Systems » fournit un guide des produits de développement Web/intranet permettant l'accès aux bases de données :

www.dbmsmag.com/pcintnet.html

www.dbmsmag.com/intsys.html

Liste de produits de développement Java :

www.sys-con.com/inetbg/index.html

www.javaworld.com/javaworld/common/jw-toolstable-alphaproduct.html

www.teamjava.com/links/#products

Revue informatiques

Certaines revues sont spécialisées dans le domaine des bases de données ; d'autres, plus généralistes, sont à la pointe de tous les nouveaux produits et architectures.

« Oracle Magazine » est une revue qui présente des articles souvent très intéressants et l'abonnement est gratuit. Profitez-en ! De plus, l'intégralité de la revue ainsi que les anciens numéros sont consultables en ligne :

www.oracle.com/oramag

« Database Programming & Design » (Miller Freeman) :

www.dbpd.com

Autres revues abordant les bases de données et Oracle :

www.pinpub.com/oracle/home.html

www.cobb.com/dbm

www.cobb.com/eod

Installation d'Oracle 10g XE

Dans cette annexe :

- les apports d'Oracle XE (Express Edition) ;
- les principales différences entre Oracle XE et les autres versions Oracle ;
- le parcours de lecture de l'ouvrage pour les utilisateurs d'Oracle XE ;
- l'installation d'Oracle XE sous Windows.

Cette annexe aborde la toute dernière version GRATUITE d'Oracle : Express Edition (XE). L'installation des versions Oracle Database Standard Edition et Oracle Database Enterprise Edition est décrite au chapitre 13, *L'installation d'Oracle 10g sous Windows*.

Les apports d'Oracle Express Edition

À qui s'adresse Oracle XE ?

Oracle XE s'adresse à :

- tous ceux qui désirent économiser le coût d'une licence Oracle pour des besoins « d'entrée de gamme » ;
- aux administrateurs qui ont besoin d'une base de données gratuite, rapide à installer, pour des tests et des déploiements ;
- aux hébergeurs qui bénéficient d'Oracle sans surcoût pour le client final, avec une facilité d'installation comparable à celle de MySQL ;

- aux sociétés éditrices de logiciels, SSII, qui ont besoin d'une base de données gratuite à distribuer avec leurs applications ;
- aux sociétés qui désirent n'avoir qu'un seul éditeur de base de données, pour limiter les surcoûts engendrés par la maintenance d'environnements hétérogènes ;
- aux développeurs d'applications Java, PHP, .NET, qui évitent ainsi de devoir développer artificiellement pour une base de données en logiciel libre des fonctions qui existent déjà dans Oracle ;
- aux écoles, universités et aux étudiants qui peuvent ajouter Oracle sur leur CV sans devoir l'acheter.

Principales différences entre XE et les autres versions Oracle

Le tableau suivant indique les principales différences entre Oracle XE et les autres versions Oracle.

Caractéristiques	Oracle XE (Express Edition)	Oracle Standard Edition ou Oracle Enterprise Edition
Gratuit	OUI	NON
Gratuit pour les hébergeurs ou dans un progiciel	OUI	NON
Installation en moins de 15 minutes	OUI	NON, mais avec une grande finesse de paramétrage.
Windows XP Professionnel, Windows 2000 Service Pack4, Windows 2003	OUI	OUI
Windows XP Standard	NON	NON
Sur une machine avec plusieurs CPU, la puissance d'un seul est utilisée.	OUI	NON, tous les CPU sont utilisés.
Une base peut héberger les données de plusieurs applications	OUI	OUI
Il peut y avoir plusieurs bases sur un serveur	NON, une seule fonctionne à un instant donné.	OUI, plusieurs peuvent fonctionner simultanément.
Les versions XE et Standard ou Enterprise peuvent coexister sans difficulté	OUI	OUI
Taille maximale de la base de donnée	4 Go	Très supérieure, limitée par l'OS.
Taille mémoire utilisable par Oracle	1 Go	Très supérieure, limitée par l'OS.
Administration	Par une interface Web très simple.	Par Oracle Enterprise Manager, très complet.

Si vous souhaitez comparer les options techniques comprises dans Oracle XE avec celles présentes dans Oracle Database Standard Edition et dans Oracle Database Enterprise Edition, consultez mon site Internet <http://gbriard.club.fr> et recherchez les articles publiés comprenant le mot-clé « Express ».

Quelle est la stratégie d'Oracle avec XE ?

Oracle montre avec XE sa volonté de ne pas laisser les bases de données libres empiéter sur son territoire historique.

Au-delà des qualités de son logiciel, la réussite commerciale d'Oracle est venue de sa capacité d'animation d'un réseau de SSII et d'éditeurs de logiciels pour développer leurs logiciels avec Oracle comme base de données. Ces progiciels, s'ils sont destinés à de petites entreprises, se satisferont d'Oracle XE. Pour les moyennes et grandes entreprises, ils seront source de licences, de maintenance et de support technique. Oracle ne voulait pas qu'un acteur des logiciels libres viennent bouleverser cette importante source de revenus, fusse au prix de la perte des licences d'entrée de gamme qui seront *cannibalisées* par XE.

Même avec une base données non libre, Oracle entend combattre les logiciels libres avec un argument de poids : la gratuité pour un vaste champ d'utilisations. C'est une opportunité pour nous, utilisateurs d'Oracle, à mettre au crédit des logiciels libres.

Parcours de lecture pour les utilisateurs d'Oracle XE

Cet ouvrage a été conçu pour couvrir toutes les versions Oracle. La nouvelle version, XE, ne comprend pas certains outils figurant dans les autres versions. Par conséquent, le plan de lecture proposé est légèrement différent pour les utilisateurs d'Oracle XE.

Chapitre s'appliquant à Oracle XE	
Partie I Introduction	
Chapitre 1 : Pourquoi choisir Oracle 10g sous Windows ?	Oui
Chapitre 2 : Les nouveautés d'Oracle 10g	Oui
Chapitre 3 : Comparatif technique Windows/Unix	Oui
Chapitre 4 : Environnement Windows : l'indispensable	Oui
Partie II Principes de fonctionnement d'Oracle 10g	
Chapitre 5 : Fonctionnement d'Oracle 10g sous Windows	Oui
Chapitre 6 : Les clusters Oracle	Non, XE ne peut pas fonctionner en cluster.
Chapitre 7 : Les processus d'Oracle 10g	Oui
Chapitre 8 : L'utilisation de la mémoire par Oracle 10g	Oui
Chapitre 9 : Les fichiers d'une base Oracle 10g	Oui
Chapitre 10 : Démarrer et arrêter une base Oracle 10g	Oui
Chapitre 11 : Transactions et accès concurrents	Oui
Chapitre 12 : Le dictionnaire de données Oracle	Oui

Chapitre s'appliquant à Oracle XE	
Partie III Installer et paramétrer Oracle 10g sous Windows	
Chapitre 13 : L'installation d'Oracle 10g sous Windows	Partiel : l'installateur Oracle Universal Installer n'est pas celui d'XE.
Chapitre 14 : Création d'une base Oracle 10g	Partiel : l'outil Oracle Database Configuration Assistant n'est pas livré avec XE.
Chapitre 15 : Oracle Net, le middleware Oracle	Oui
Chapitre 16 : Accéder à Oracle depuis Windows	Oui
Partie IV Outils et langages de programmation	
Chapitre 17 : Programmer avec PL/SQL	Oui
Chapitre 18 : Les outils d'Oracle 10g	Oui
Chapitre 19 : Pilotes JDBC et SQLJ	Oui
Chapitre 20 : Java et Oracle 10g	Non, le moteur Java intégré dans Oracle n'est pas livré avec XE.
Chapitre 21 : XML et Oracle 10g	Oui
Chapitre 22 : Présentation d'Oracle Application Express	Oui, Oracle Application Express est installé nativement avec XE.
Partie V Administration d'Oracle 10g	
Chapitre 23 : Gestion de l'espace disque et des fichiers	Oui
Chapitre 24 : Stratégie de sécurité sous Windows	Oui
Chapitre 25 : Oracle Enterprise Manager	Non, OEM n'est pas livré avec XE. Une interface plus simple basée sur Oracle Application Express permet d'effectuer l'administration courante d'XE.
Chapitre 26 : La sauvegarde d'une base Oracle 10g	Oui
Chapitre 27 : Configurer les disques pour Oracle	Oui
Chapitre 28 : Optimisation et performances	Oui
Partie VI Annexes	
Annexe 2 : Mémento SQL	Oui
Annexe 3 : Procédures pour le DBA	Oui
Annexe 4 : Extensions des fichiers d'Oracle	Oui
Annexe 5 : NLS : support des langues par Oracle	Oui
Annexe 6 : Description des produits livrés	Oui
Annexe 7 : Sites Internet et bibliographie	Oui
Annexe 8 : Installation d'Oracle 10g Express Edition (XE)	Oui

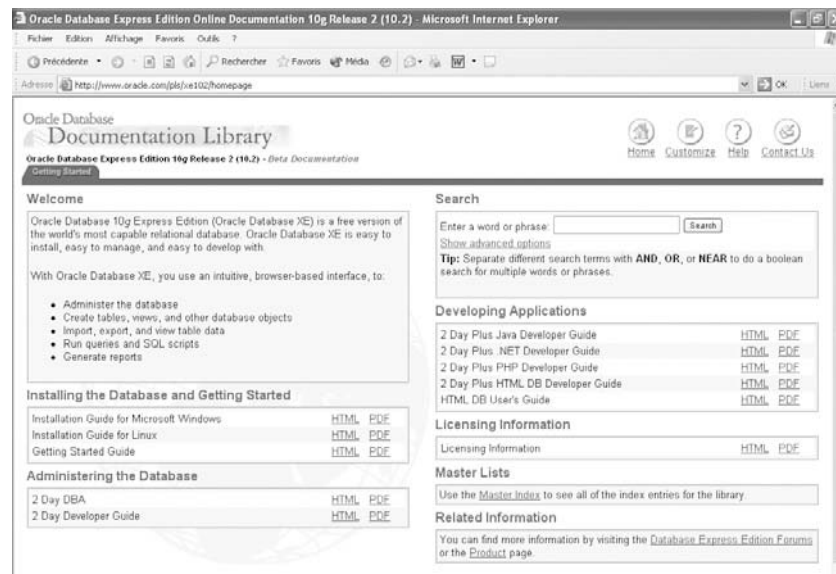
Les ressources Oracle

Les ressources indispensables sont accessibles depuis les menus Windows, une fois XE installé :

- la documentation sur le site Oracle : <http://www.oracle.com/pls/xe102/homepage> ;
- le forum en ligne (inscription préalable nécessaire) ;
- l'aide en ligne locale, installée avec Oracle XE.

Figure A8-1

Documentation
en ligne



J'attire votre attention sur la documentation en ligne disponible à l'adresse <http://www.oracle.com/pls/xe102/homepage>, où vous trouverez tout pour installer et commencer rapidement avec Oracle XE. Figurent aussi des tutoriaux pour installer et développer avec Java, .Net, PHP et Oracle Application Express.

Les principales publications Oracle concernant XE sont :

- **Oracle Database Express Edition 2 Day DBA.** Présente les tâches courantes d'un administrateur.
- **Oracle Database Express Edition 2 Day Developer Guide.** Décrit les principaux apports d'XE pour développer des applications.
- **Oracle Database Express Edition 2 Day Plus PHP Developer.** Excellente introduction pour les programmeurs PHP-Oracle.
- **Oracle Database Express Edition 2 Day Java Developer Guide.** Excellente introduction pour les programmeurs Java-Oracle.

Les autres ressources

Je n'ai jamais vu une nouvelle version Oracle provoquer un tel engouement. Dès sa parution, de nombreux sites Web ou blogs, ont parlé d'Oracle XE.

Parmi ces ressources :

<http://www.rci-informatique.fr>

<http://www.sitepoint.com/blogs/2005/11/01/oracle-10g-xe-and-php/>

<http://www.orablogs.com/sergio/>

<http://www.grallandco.com/blog/>

Et bien sûr mon site : <http://gbriard.club.fr>

Installation d'Oracle XE sous Windows

Pour information, consultez les prérequis indiqués au chapitre 13, *L'installation d'Oracle 10g sous Windows*.

Pour installer Oracle XE, il est indispensable de se connecter sous un compte Windows possédant les droits Administrateur.

L'installation

Une fois téléchargé, Oracle XE s'installe en moins de 15 minutes. Un record pour Oracle !

- Téléchargez la dernière version d'Oracle XE depuis le site <http://www.oracle.com/technology>. Elle se compose d'un seul fichier d'environ 155 Mo.
- Lancez le fichier. Une installation au format InstallShield démarre.

Figure A8-2

Décompactage de fichier
d'installation

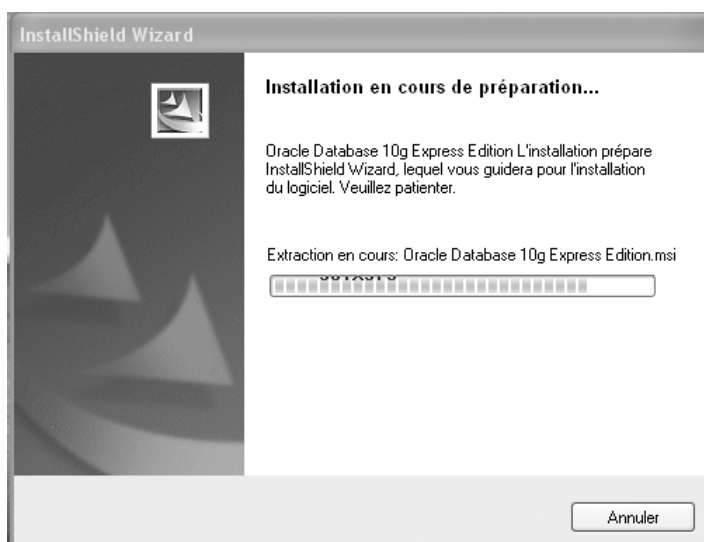
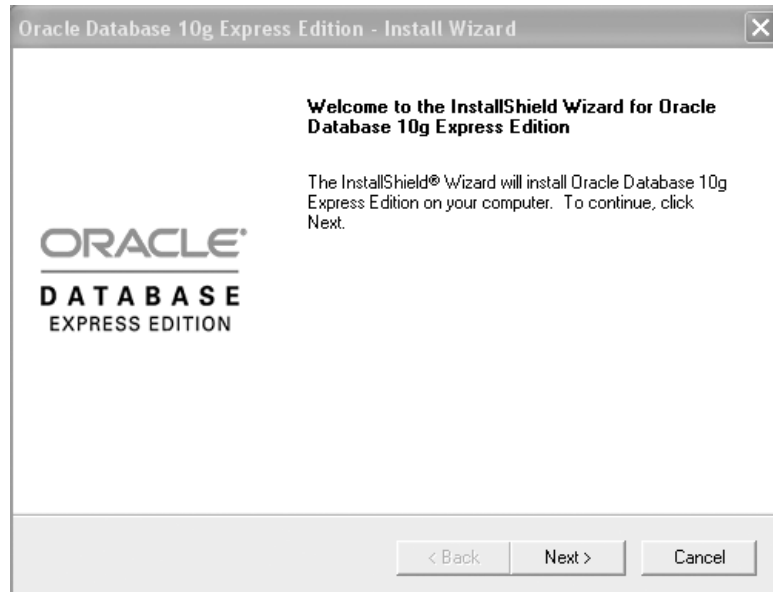


Figure A8-3

Lancement de l'installation

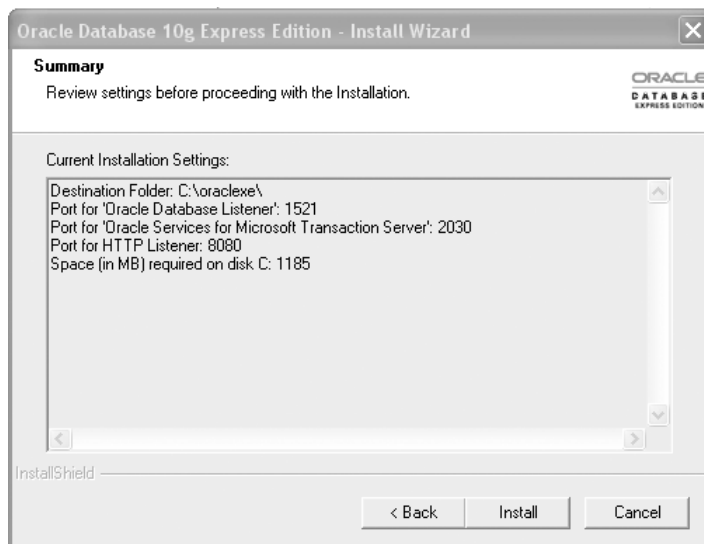
- Lisez attentivement les conditions d'utilisation d'Oracle XE. Ce n'est pas un logiciel libre. Même si la version est gratuite, c'est toujours un logiciel commercial !
- Indiquez le répertoire d'installation d'Oracle XE.
- Indiquez le mot de passe qui sera identique pour les comptes Oracle SYS et SYSTEM. Conservez précieusement ce mot de passe !
- Le résumé suivant s'affiche. Les options à saisir sont limitées au maximum ! Conservez les numéros de ports utilisés par Oracle. Ils seront utiles ultérieurement. Les valeurs habituelles sont :
 - **1521** : pour le listener Oracle
 - **2030** : pour les services Microsoft Transaction Server d'Oracle
 - **8080** : port HTTP de l'interface graphique d'Oracle XE

Si votre machine héberge un firewall (cas de Windows XP Professionnel avec le Service Pack 2), il faudra ouvrir les ports 1521 et 8080 pour accéder respectivement à la base depuis un serveur d'application distant (cas d'un serveur Web) et pour administrer à distance la base via un navigateur Web.

Lancez l'installation.

Figure A8-4

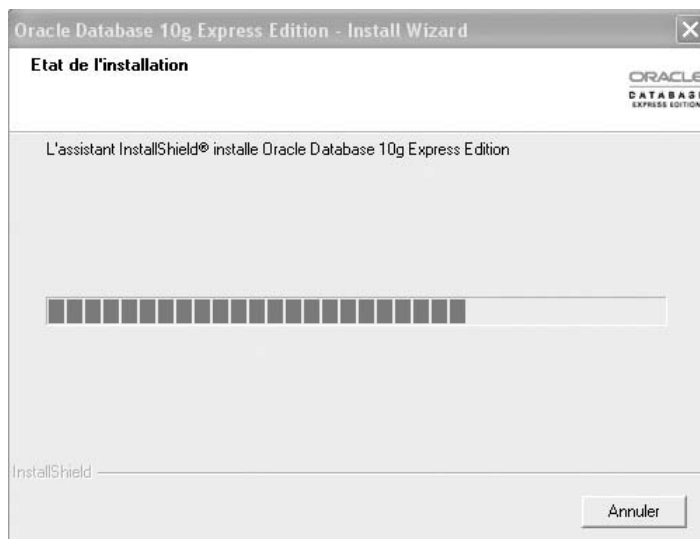
Résumé des paramètres



- Et c'est parti ! En moins de 10 minutes, l'installation est terminée. Un record pour Oracle. Comme beaucoup, j'attendais cela depuis longtemps !

Figure A8-5

Installation



- Vous êtes ensuite automatiquement dirigé vers la page d'accueil d'administration de la base. Pour cela, saisissez SYSTEM et votre mot de passe.

- Si vous vous connectez avec un compte utilisateur différent de SYSTEM ou SYS, vous accéderez ainsi à la toute dernière version d'Oracle Application Express, présentée au chapitre 22, *Présentation d'Oracle Application Express*.

À vous maintenant de découvrir Oracle XE et Oracle Application Express.

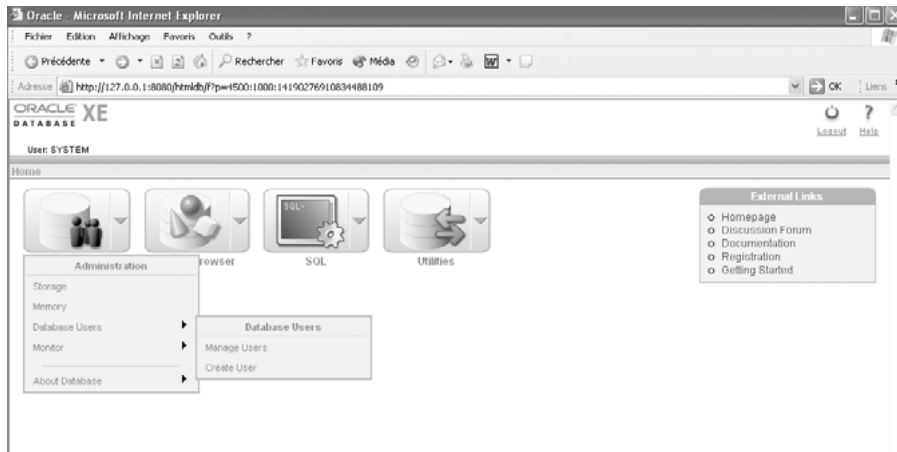


Figure A8-6

Oracle Application Express

Le paramétrage optionnel

Inscrivez-vous sur le forum gratuit Oracle XE, depuis le menu démarrer. Vous trouverez ce forum l'adresse suivante : <http://www.oracle.com/technology>.

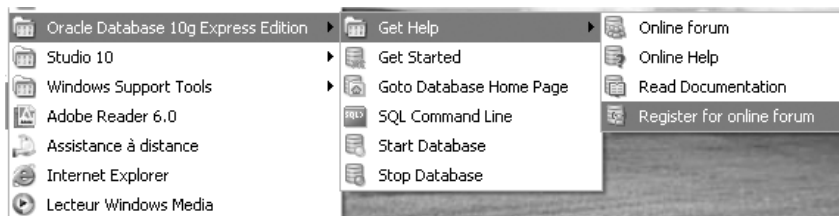


Figure A8-7

Inscription au forum Oracle XE

Si vous êtes un habitué de MySQL, la simplicité des menus d'Oracle XE vous rappellera certaines interfaces graphiques. Cela nous change des menus « usines à gaz » habituels.

Consultez par curiosité les fichiers StartDB.bat et StopDB.bat situés sous C:\oracle\app\oracle\product\10.2.0\server\BIN.

Commandes utiles

Pour accéder à l'outil d'administration d'une base Oracle XE locale ou distante, l'URL est `http://serveur:port/htmldb` avec :

- `serveur` : le nom du serveur Oracle XE ou son adresse IP. Si le serveur est situé sur votre poste, vous pouvez utiliser l'adresse IP locale 127.0.0.1.
- `port` : port TCP que le listener écoute. Par défaut, c'est le 8080.

Par exemple, si le serveur se nomme Cyrille, l'URL sera `http://cyrille:8080/htmldb`.

L'interface d'Oracle Application Express sera proposée en plusieurs langues dans la version de production. Si vous choisissez comme jeu de caractères de la base :

- `WE8MSWIN1252` : l'interface sera uniquement en anglais.
- `AL32UTF8` : l'interface sera multilingue, dont le français.

Pour plus d'informations concernant le jeu de caractères d'une base, consultez l'annexe 5, *NLS : support des langues par Oracle et l'euro*.

Développer en PHP

Oracle a fait de très gros efforts en direction de la communauté des développeurs PHP. Le tutorial mentionné précédemment est représentatif. Si le couple PHP-Oracle vous tente, étudiez le logiciel **Zend Core for Oracle** qui a été développé par Zend (société des créateurs de PHP) à la demande d'Oracle.

Zend Core for Oracle comprend :

- PHP ;
- une version optimisée des OCI (l'interface la plus fine de programmation d'Oracle), qui évite de devoir installer un « client Oracle » ;
- une interface de configuration Apache – PHP.

Résumé

Cette annexe vous a présenté les principaux apports d'Oracle XE. Cette version va sans aucun doute révolutionner le marché des bases de données. Oracle montre ainsi sa volonté de ne pas laisser les bases de données libres venir empiéter sur son territoire de prédilection. Ce fait est très positif pour de nombreuses entreprises. Sachons en profiter avantageusement.

Index

Symboles

.ctl 150
.dbf 150
.log 150
.NET 316, 828
@ 427
@@ 427
procédure stockée 499

Numériques

1521 833
2030 833
8080 833

A

Access 387
Access Advisor 755
accident d'instance 765
ACID 172
Active Directory 765
ActiveX Data Objects XLVII, 312
ADDM 24, 752
administrateur de disques 52
Administration Assistant for Windows 67
ADO XLVII, 312
Advanced Queueing 813
Advanced Replication 275, 813
agent intelligent 683
AL32UTF8 217, 250, 836

alertSID.ora 142
alias 295, 765
ALL (vues commençant par ~) 182
ALL 182
ALTER DATABASE 169
ALTER SYSTEM 656
AMM 24
AMT 24
analyseur de performances 735
Apache 318, 598, 765
Applet 765
Appliance 765
Application Builder Voir Oracle
Application Express 600
ARC 81
ARCH 62
architecture Oracle sous Windows 62
archivage 765
ARCHIVELOG 113
ARCH Voir thread 112
arrêter
 Oracle sous Windows 67
 service OracleServiceSID 86
 une instance 84, 86
ASM 23, 219, 717, 725
ASMM 129
ASP 318, 765
Assistant Common Files 813
Assistant d'administration Oracle
 pour Windows 86

atomicité 172
audit 765
audit_trail 141
authentification 765
AUTOALLOCATE 615
Automatic Database Diagnostic Monitor 24, 752
Automatic Maintenance Task 24
Automatic Memory Management 24
Automatic Shared Memory Management 129
Automatic Storage Management 23, 219, 717, 725
Automatic Workload Repository 23, 679, 752
autorisation 766
AUTOTRACE 746
AWR 23, 679, 752

B

background_dump_dest 140
base de données 61, 766
 arrêt 166
 création 235
 démarrage 162
 démontage 166
 fermeture 166
 montage 162
 ouverture 162

base de registre 43, 44, 65, 766
 base et instance Oracle 10g 159
 batch 457
 bdump 152
 BIGFILES 626
 bloc Oracle 611, 766
 Borland 314
 BUFFER 394
 Business Objects 314

C

C++ 314
 cartographie 381
 Cascading Style Sheet 520
 catalog.sql 269
 catalog.sql script 180
 catblock.sql script 181
 caths.sql script 181
 catio.sql script 181
 catockt.sql script 181
 catparr.sql script 181
 catproc.sql script 180
 catrep.sql script 180, 181
 catman.sql script 181
 cdump 152
 CGI 15, 766
 Character Set Scanner 813
 CHARACTER_SET 250
 CHARACTERSET *Voir* NLS 799
 checkpoint 111, 766
 CJQ 81
 CKPT *Voir* thread 81, 108
 Clé 44
 client léger 766
 client-serveur 10, 282, 766
 cluster XLV, 91, 203, 766
 de tables 766
 nœud 92
 Cluster Ready Service 203
 CLUSTER_DATABASE 128
 Cognos 314
 cohérence 172
 COM 312
 COM Automation 322
 COM/COM+ 766
 commandes
 SQL 775
 Commit 766
 communiquer entre bases Oracle
 11
 Companion CD 203
 COMPATIBLE 128

compatibles 140
 compilation JIT 492
 compilation native 492
 COMPILE 400
 Component Object Model 312
 COMPRESS 394
 concurrence d'accès 767
 CONNECT INTERNAL 421, 668
 Connection Caching 466
 connexion Oracle 767
 CONSISTENT 394
 CONSTRAINTS 394, 400
 CONTROL_FILES 128
 control_files 139
 core_dump_dest 140
 CPU 767
 create 152
 CREATE CONTROLFILE 648
 CREATE DATABASE 267
 CREATE PFILE 655
 CREATE SPFILE 654
 CreateDB.sql 267
 CreateDBCatalog.sql 269
 CreateDBFiles.sql 269
 création
 d'une instance 83
 d'une base de données 235
 CRS 203
 CSS *Voir* Cascading Style Sheet
 520
 CTXXPATH 557
 customScripts.sql 271

D

DAD 598
 daemons 46
 Data Marts 13
 Data Mining 817
 Data Pump 32
 Export 32
 Import 32
 Data Pump Export 68, 405
 Data Pump Import 68, 405
 DataSource 461
 Database Access Descriptor 598
 Database Buffer 127
 Database Configuration Assistant
 67
 Database link 767
 DataWarehouse 13
 db_block_buffers 140
 db_block_size 140

DB_BLOCK_SIZE 128, 249, 258,
 742, 750
 db_cache_size 140
 DB_CACHE_SIZE 750
 DB_CREATE_FILE_DEST 128,
 650
 DB_CREATE_ONLINE_LOG_D
 EST_n 128, 650
 DB_DOMAIN 128, 217, 226, 242,
 267
 db_file_multiblock_read_count
 141
 db_name 139
 DB_NAME 128, 217, 242, 267
 DB_RECOVERY_FILE_DEST
 129
 DB_RECOVERY_FILE_DEST_
 SIZE 129
 DB_UNIQUE_NAME 129
 DBA 160, 666
 l'vues commençant par ~ 182
 procédures d'administration
 781
 DBCA 67, 235
 db-files 139
 DBMS_ALERT 366, 379
 DBMS_AQ 366, 380
 DBMS_DATAPUMP 409
 DBMS_JAVA 480, 489, 490
 DBMS_JOB 366, 678
 DBMS_LDAP 371
 DBMS_LOB 366, 372
 DBMS_LOGMNR 136, 381, 646
 DBMS_LOGMNR, 366
 DBMS_LOGMNR_D 136, 381,
 646
 DBMS_METADATA 376
 DBMS_OUTPUT 366, 367
 DBMS_PIPE 366, 379
 DBMS_SQL 366, 377
 DBMS_STANDARD 366, 367
 DBMS_UTILITY 366, 381
 Dbms_warning 32
 DBMS_XDB 530
 DBMS_XDB_VERSION 533
 DBMS_XMLDOM 567
 DBMS_XMLPARSER 381, 567
 DBMS_XMLSCHEMA 551
 DBMS_XSLPROCESSOR 567,
 568
 DBW0 80

DBWR *Voir* thread 62, 108, 135
 DCOM 322, 767
 DDL (Data Definition Language) 775
 déclencheur Java 507
 Définition d'un type de document 518, 574
 démarrer
 Oracle sous Windows 67
 service OracleServiceSID 86
 une instance 83, 86
 déplacer une base vers un autre serveur 86
 deploync 494, 497
 DICTIONARY 26, 183
 dictionnaire de données 131, 179, 767
 DIRECT 394
 diskperf 735
 Distributed Component Object Model 322
 distribution 134
 DLL 767
 DML (Data Manipulation Language) 779
 DMZ 598
 Dnnnn *Voir* thread 113
 DOM (Document Object Model) 561
 Domaine 767
 dropjava 489
 DTD (Document Tag Définition) 767
 Voir Définition d'un type de document 517
 durabilité 172

E

EJB 767
 EM2go 679
 Email 767
 Enterprise JavaBeans 813
 Enterprise Manager 227
 espace disque 605
 gestion de l'~ 606
 ETL 767
 événement *Voir* Oracle Enterprise Manager 684
 Excel 822

EXP 68
 exp 152
 EXPDP 68
 Expor 68
 Export 388, 679
 Export/Import 388, 704
 Express Edition *Voir* Oracle Express Edition 827
 ext 718
 ext2 717, 718
 eXtensible Markup Language 515
 eXtensible Stylesheet Language 521
 extent ou extension 611

F

FEEDBACK 394
 fetchlets 679
 fichier
 ajouter un fichier à une base 135
 d'administration 59
 d'alerte de la base 142
 d'échange 736
 d'initialisation réseau 141
 d'une base Oracle 133
 de contrôle 59, 138, 767
 de données 59, 134, 149, 768
 de pagination 748
 initialisation initSID.ora 138
 initialisation sfileSID.ora 138
 lien avec les tablespaces 631
 persistant 654
 redo-log 59, 136, 768
 redo-log archivés 137
 trace 768
 trace d'Oracle Net 144
 FILE 394
 FILESIZE 394
 FireWall 768
 flash recovery area 219
 Flashback Query 637, 638
 freeware 822, 825
 FTP 247, 768
 FULL 394

G

GATHER_STATS_PROG 744
 Generic Connectivity 813
 géolocalisation 381
 gestion d'une base Oracle 10g 157

gestionnaire
 de services 46
 de tâches Windows 734
 des tâches 49
 Global Database Name 217, 226
 GLOBAL_DBNAME 242, 297
 GRANTS 394
 Grid 679, 680
 Grid Computing 91
 Grid Control 218, 680
 groupe DBA 160
 GUID 768
 GV\$ 183

H

HARDWARE 45
 HASPATH 557
 HCKU 45
 hébergement 768
 HKCC 45
 HKCR 45
 HKEY_CLASS_ROOT 45
 HKEY_CURRENT_CONFIG 45
 HKEY_CURRENT_USERS 45
 HKEY_LOCAL_MACHINE 45, 88
 HKEY_USERS 45
 HKLM 45
 HKU 45
 HOME_NAME 768
 HOMEID 66, 768
 hpfs 718
 HTML (HyperText Markup Language) 16, 515, 768
 Oracle Application Express 32, 830
 Application Builder 600
 SQL Workshop 601
 workspace 602
 HTTP 247, 768
 HyperText 768
 HyperText Markup Language 515

I

iAS 319
 IDREF, IDREFs 583
 IFS (Internet File System) 372
 Voir Oracle iFS 770

IID 768
 IIS 318, 768
 IMP 68
 IMPDP 68
 Import 68, 388, 679
 Import / Export 388
 Impromptu 314
 INCTYPE 394
 Index 768
 Index Bitmap 768
 INDEXES 394
 infocentre 8, 13
 Informix 387
 INIT.ORA 768
 init.ora 138, 248, 271
 INPATH 555
 instance 61, 159, 768
 arrêter 84
 création 83
 démarrer 83
 déplacer 86
 modifier 85
 SID 769
 supprimer 85
 instance_name 139
 INSTANCE_NUMBER 129
 Instant Client 280
 interMedia 275
 interMedia Locator 275
 Internet 769
 fonctionnement 14
 Application Server 319
 Information Server 318
 intranet 14, 769
 dynamique 15
 statique 15
 IO ou I/O 769
 IPC 292
 iso9660 718
 isolation 172
 iSQL*Plus 227

J

J2EE 769
 Java 17, 318, 769, 815, 824, 828
 Java Accelerator 493
 Java Database Connectivity 441
 ouvrir une connexion 447
 requête 449
 Java J2EE 476
 Java Pool 131

Java Runtime Environment (JRE)
 813
 JAVA\$OPTIONS 490
 JAVA_HOME 493
 JAVA_POOL_SIZE 131, 478, 750
 java_pool_size 140
 JAVA_SOFT_SESSIONSPACE_
 LIMIT 478
 JAVA_SOFT_SESSIONSPACE_
 SIZE 478
 Javascript 769
 JavaServer Pages (JSP) 769
 JDBC 443, 500, 815, 824
 Voir Java Database
 Connectivity 441
 JDBC-OCI 280
 Jdeveloper 594
 jeu de caractères
 UNICODE 805
 Voir NLS 799
 JNDI 463
 JOB_QUEUE_PROCESSES 129
 JVM (Java Virtual Machine) 770

K

Kit de Ressources techniques de
 Windows 53, 54

L

L4G 594
 large_pool_size 140
 LDAP 285, 371, 769
 LDD 173
 lecture consistante des données
 175
 LGWR 81
 Voir thread 108
 Linux versus Windows 2000 35
 Listener 286
 listener 769
 listener.log 307
 listener.ora 288, 289, 769
 listener *Voir* thread 112
 Loader *Voir* SQL*Loader 409
 loadjava 485, 501
 LOCALY MANAGED 26, 258
 Locator 275
 LOCK_SGA 128
 LOG 394
 Log Miner 136, 381, 645

LOG_ARCHIVE_DEST_n 129
 LOG_ARCHIVE_DEST_STATE
 _n 129
 log_archive_start 141
 LOG_BUFFER 751
 log_buffer 141
 log_checkpoint_interval 141
 logiciel libre 828
 lsnrctl 304

M

mainframe 4, 769
 MAPI 769
 max_dump_file_size 140
 mémoire 60
 partagée 60
 utilisation par Oracle9i 123
 messagerie 8
 Microsoft .NET 594, 769
 Microsoft .NET Data Provider for
 Oracle 316
 Microsoft Access 9, 10, 594
 Microsoft Cluster Server 91, 769
 MSCS 97
 Microsoft Management Console
 50, 769
 Microsoft Transaction Server 320,
 769
 Microsoft Visual Studio 314
 middleware 280
 Migration Workbench 204, 321,
 387
 minix 718
 MMC 50, 769
 modifier
 une instance 85
 Mount 769
 MSCS 769
 msdos 718
 MTS (Multi-Thread Server) 114,
 117, 320, 769
 multiprocesseurs 10
 multithreading 309
 MySQL 827

N

NAME 146
 National Language Support *Voir*
 NLS 799
 ncomp 494
 net start 74
 net stop 77

Net8 Voir Oracle Net 279
 NetBEUI 770
 NetBios 770
 netca 288, 290
 netmgr 288, 300
 nfs 718
 NLS 770, 799
 CHARACTERSET 801
 Export/Import 804
 LANG 37
 NLS_CURRENCY 806
 NLS_DATABASE_PARAME
 TERS 802
 NLS_DATE_FORMAT 806
 NLS_LANG 146, 806
 NLS_LANGUAGE 806
 SQL*Loader 804
 NLS_CHARACTER_SET 251
 NLS_LANGAGE 129
 NLS_TERRITORY 129
 NTFS 218, 274

O

Object Linking and Embedding 312
 Object Type Translator 813
 OCCI 280
 OCI XLVII, 280, 312, 317, 770,
 814
 OCOPY 39, 68
 ocopy.exe 707
 ODBC XLVII, 280, 312, 770
 ODBC .NET XLVII, 312
 OEM 33, 218, 227
 OFA (Optimal Flexible Architec-
 ture) 25, 38, 133, 741, 770
 OFFLINE 39
 ojvmjava 491
 OLE 312, 770
 OLE DB XLVII, 312
 OLE DB .NET XLVII, 312, 316
 ONLINE 39
 OO4O XLVII, 312, 315
 open_cursors 139
 Operating System 770
 Optimal Flexible Architecture 25,
 133, 147, 238
 optimisation 729
 optimiseur 743
 OPTIMIZER_MODE 744
 ORA_SID_AUTOSTART 168
 ORA_SID_SHUTDOWN 168

ORA_SID_SHUTDOWN_TIME
 OUT 168
 ORA_SID_SHUTDOWNTYPE
 168
 Oracle
 Database Configuration
 Assistant 235
 fonctionnement 57
 iFS 770
 installation 201
 automatique 230
 JServer Voir Oracle JVM 770
 JVM (Java Virtual Machine)
 770
 logiciels 134
 optimiseur 743
 partitionnement 770
 Oracle Administration Assistant
 for Windows 67, 74, 86, 147, 275
 Oracle Administrative Assistant
 for Windows 663
 Oracle Advanced Replication 275
 Oracle Advanced Security 814
 Oracle Application Server 594
 Oracle Call Interface (OCI) XL-
 VII, 312, 317, 814
 Oracle COM Automation 321, 322
 Oracle COM Automation Feature
 770
 Oracle Connection Manager 814
 Oracle Data Migration Assistant
 814
 Oracle Data Provider for .NET
 XLVII, 312, 316
 Oracle Data Pump Export 405
 Oracle Data Pump Import 405
 Oracle Database Client 203
 Oracle Database Companion CD
 components 203
 Oracle Database Configuration
 Assistant 67, 68, 235, 814
 Oracle DBA Management Pack
 814
 Oracle DBA Studio 814
 Oracle Documentation 814
 Oracle Enterprise Login Assistant
 814
 Oracle Enterprise Manager 33, 67,
 76, 218, 227, 770, 814
 agent 683
 applications standard 685
 console 690

Database Control 680
 Database Grid Control 680
 event 684
 job 683
 présentation 678
 référentiel 685
 repository 685
 Oracle Enterprise Manager Bac-
 kup Wizzard 68
 Oracle Enterprise Manager Confi-
 guration Assistant 814
 Oracle Enterprise Manager Con-
 sole 68, 814
 Oracle Enterprise Manager Grid
 Control 218
 Oracle Enterprise Manager Inte-
 grated Applications 814
 Oracle Enterprise Manager Load
 Wizzard 68
 Oracle Enterprise Manager Mana-
 gement Packs 814
 Oracle Enterprise Manager Quick
 Tour 815
 Oracle Enterprise Manager Reco-
 very Wizzard 68
 Oracle Enterprise Manager Web
 Site 815
 Oracle Express Edition 827
 Oracle Fail Safe
 comparatif avec Oracle Real
 Application Clusters 98
 Microsoft Cluster Server 97
 objectifs 93
 Oracle Failsafe 91, 93
 Oracle Forms 594
 Oracle Home 66, 88, 145
 Oracle HTTP Server 815
 Oracle Integration Server 815
 Oracle Intelligent Agent 815
 Oracle interMedia 275, 815
 Oracle Internet Directory 815
 Oracle Internet Directory Client
 Toolset 815
 Oracle Internet Directory Configu-
 ration Assistant 815
 Oracle Internet Directory Manager
 815
 Oracle Internet File System 815
 Oracle Java Database Connectivi-
 ty (JDBC) 815
 Oracle Java Tools 815
 Oracle JavaServer Pages 815

Oracle Jdeveloper 594
 Oracle Label Security 815
 Oracle Local Builder 815
 Oracle Locator 815
 Oracle Management Server 815
 Oracle Migration Workbench 204, 387
 Oracle Names 285, 816
 Oracle Net 279, 443, 770, 816
 Oracle Net Configuration Assistant 288, 290, 816
 Oracle Net Listener 816
 Oracle Net Manager 288, 300, 816
 Oracle Objects for OLE XLVII, 312, 315, 770
 Oracle OLAP 816
 Oracle OLE DB XLVII, 312
 Oracle Parallel Query 96
 Oracle Parallel Server
 objectifs 93
 Oracle Partitioning 816
 Oracle Performance Monitor pour Windows 735
 Oracle Personalization 816
 Oracle PL/SQL Gateway 816
 Oracle Policy Manager 816
 Oracle Portal 595
 Oracle Real Application Clusters 91, 114, 203, 725, 770, 816
 comparatif avec Oracle Fail Safe 98
 Oracle Real Application Clusters (RAC) 93
 Oracle Real Application Clusters Guard 816
 Oracle Recovery Manager 219
 Oracle Required Support Files 816
 Oracle Server Manager 75
 Oracle Servlet Engine 816
 Oracle Spatial 275, 817
 Oracle SQLJ 817
 Oracle SQLJ Runtime 817
 Oracle SQLJ Translator 817
 Oracle Syndication Server 817
 Oracle Text 275, 817
 Oracle Text Manager 817
 Oracle Ultra Search 817
 Oracle Universal Installer 149, 817
 installation automatique 230
 Oracle Utilities 817
 Oracle Wallet Manager 817

Oracle WebDB Voir Oracl Appli-
 cation Express XLVIII, 593
 Oracle Workflow 817
 Oracle XML Developer's Kit 817
 Oracle XML SQL Utility 817
 oracle.exe 63
 Oracle 10g 543
 contenu du produit 18
 Oracle 10g Unified Scheduler 24
 ORACLE_BASE 36, 149, 770
 ORACLE_HOME 36, 149, 770
 Oracle_XE Voir Oracle Express
 Edition 827
 Oracle9i
 sécurité 659
 tuning 742
 Oracle9i Advanced Analytic Ser-
 vices 817
 Oracle9i JVM Accelerator 818
 Oracle9i JVM Enterprise Edition
 818
 Oracle9i JVM Servlet Container
 818
 Oracle9i Required Support Files
 816
 Oracle9i Server 818
 OracleDBConsoleTEST 70
 OracleOraHome92TNSListener
 304
 OracleService 63
 OracleServiceSID 168
 ORADIM 38, 67, 73, 82
 orapwd 671
 OSDBA 36
 OSOPER 36
 OUI Voir Oracle Universal Instal-
 ler 817
 outil complémentaire
 Process Viewer 54
 outil standard
 analyseur de performances 50
 gestionnaire des tâches 49
 l'observateur d'événements 52
 Regedit 53
 Terminal Server 53
 OWNER 394

P

page fault 749
 pagination 732, 748, 749
 paramètre d'initialisation
 audit_trail 141

background_dump_dest 140
 compatibles 140
 control_files 139
 db_block_buffer 140
 db_block_size 140, 249, 258, 611
 db_cache_size 140
 db_create_file_dest 650
 db_create_online_log_dest_n 650
 db_domain 139, 242, 267
 db_file_multiblock_read_coun-
 t 141
 db_name 139
 global_dbname 242
 instance_name 139, 242, 267
 java_pool_size 131
 large_pool_size 140
 log_archive_dest 641
 log_archive_format 641
 log_archive_start 141, 641
 log_buffer 130, 141
 log_checkpoint_interval 141
 max_dump_file_size 140
 open_cursors 139
 processes 141
 remote_login_passwordfile 667, 672
 rollback_segment 141
 shared_pool_size 131, 140
 sort_area_size 132, 625
 sql_trace 141
 timed_statistics 141
 undo 269
 undo_management 141, 637
 undo_tablespace 141, 637
 user_dump_dest 140
 PARFILE 394
 parse 131
 partitionnement 770
 PATH 771
 PDA 679
 Performance Monitor 736
 Perl 15, 318
 PERMANENT 256
 permissions Java_2 480
 PFILE 655
 pfile 152
 pfmon.exe 736
 PGA 108, 117
 Process Global Area 131
 Program Global Area 131

PGA_AGGREGATE_TARGET
129, 132, 140, 750
PHP 15, 17, 282, 318, 594, 828
Pilote
 JavaSoft 444
 JDBC interne 447
 JDBC OCI 445
 JDBC Thin 446
PL/SQL 31, 327, 598, 771
 %ROWTYPE 331
 %TYPE 331
 bloc 330, 335
 CLOSE 345
 curseur
 %FOUND 348
 %ISOPEN 349
 %NOTFOUND 349
 %ROWCOUNT 349
 CLOSE 347
 déclaration 346
 exemple 349
 FETCH 347
 FOR IN 354
 FOR LOOP 353
 OPEN 346
 simplification d'écriture 351
échange de variables avec
 SQL*Plus 341
EXCEPTION 332, 356
exécuter avec SQL*Plus 337
FETCH 345
fonction 331, 358
FOR-LOOP 343
IF-THEN-ELSE 342
LOOP 342
OPEN 345
procédural 329
procédure 331, 358
Shared Pool 131
variable 330, 337
WHILE-LOOP 344
Platinum Technology 438
PMON 62, 80, 771
 Voir thread 108
Pnnnn *Voir* thread 113
pool de connexions 464
Portal_PL/SQL 329
postDBCreation.sql 270
Powerbuilder 314
précompilateur 771
privlège 771
Pro*C / C++ 818

procédure 771
 externe 771
Process Global Area 108, 117
Process ou Processus 771
Process Viewer 54, 65, 71
PROCESSES 129, 139, 141
processus 59
 Voir thread 108
PRODUCT_USER_PROFILE
 424
Program Global Area 117
protocoles réseau 733

Q

QUERY 394
Quests Software 438
Quota 771

R

RAC 91, 114, 203
RAID 218, 717
RAM 771
Raw Devices 104, 219, 717, 724
read only 176
Real Application Clusters 114,
 203, 725
RECO 81
RECORD 394
RECORDLENGTH 394
Recovery Manager 68, 219, 705
RECO *Voir* thread 112
redo-log 771
 buffer 130
 groupe de ~ 137
 Log Miner 645
référentiel *Voir* Oracle Enterprise
 Manager 685
registre
 clé 44
 sous-clé 44
 valeur 44
registre 43, 766
reiser 717, 718
REMOTE_LISTENER 129
REMOTE_LOGIN_PASSWORD
 FILE 129, 139, 667, 672
replication 12, 275, 771
repository *Voir* Oracle Enterprise
 Manager 685
Required Support Files 816
restauration 771
 d'instance 771

ResultSet 459
rfcfg 735
RMAN 68, 705
rôle 771
Rollback 771
rollback segment 129, 141, 174,
 744
ROWS 394

S

SAM 45
SAN 218
Satan 675
sauvegarde 701, 771
 script de commandes 709
savepoint *Voir* transaction 173
SAX (Simple API for XML) 561
scalability 772
schéma 772
schéma XML 772
Scott 772
Scratchpad 737
script Shell 5, 772
SDO 381
SECURITY 45
segment 611
 UNDO 174
Segment Advisor 628
SELECT 772
 FOR UPDATE 178
Server Generated Alerts 25
Server Manager 75, 162, 388, 419
service 44, 45, 46, 63, 66, 772
 démarrer 47
 démarrer/arrêter
 OracleServiceSID 86
 OracleOraHome92TNSListene
 r 304
 OracleServiceSID 63
service Pack 733
servlet Java 15
servlets 772
SESSIONS 129, 772
sessions 141
SGA 60, 61, 108, 772
SGA System Global Area 125
SGA_TARGET 129, 750
sga_target 129, 140
SGML (Structured Generalized
 Markup Language) 515

Shared Pool 130
 SHARED_POOL_SIZE 140, 477, 750
 SHARED_SERVERS 129
 shareware 822, 825
 SHRINK 628
 SHRINK SPACE 628
 SHUTDOWN 77
 ABORT 166
 IMMEDIATE 166, 169
 NORMAL 166
 TRANSACTIONAL 166
 SID 217, 226, 772
 Simple API for XML 561
 single-task 115
 SMON 81
 Voir thread 108
 snap-ins 50, 769
 snapshot 772
 Snnnn *Voir* thread 113
 SNPnn *Voir* thread 112
 SOFTWARE 45
 Solaris 772
 SORT_AERA_SIZE 140, 751
 souche technologique 7
 souplesse_architecture 11
 sous-clé 44
 Spatial 275
 spécification d'appel 501
 SPFILE 253, 654
 spfile.ora 138, 248
 SQL 772
 memento 775
 procédures d'administration 781
 SQL Access Advisor 755
 SQL Server 387
 SQL Tuning Adviser 24, 745
 SQL Workshop *Voir* Oracle Application Express 601
 SQL*Loader 68, 409
 SQL*Net *Voir* Oracle Net 279
 SQL*Plus 67, 68, 226, 419, 746, 818
 fonction 361
 plan d'exécution SQL 745
 procédure 361
 d'administration 781
 SHOW ERROR 362
 visualiser des variables 363
 SQL*Plus Worksheet 818
 SQL*Worksheet 438

sql.bsq script 180
 sql_trace 141
 SQLJ 441, 448
 ouvrir une connexion 451
 requête 451
 sqlj 454
 SQLLDR 68
 sqlnet.ora 288
 SQL-Programmer 438
 STA 24
 START 427
 START_TRANSFORMATION_ENABLED 129
 startup 76
 STARTUP NOMOUNT 164
 statement caching 465
 STATISTICS 394
 statistics_level 130
 statistiques 744
 STATSPACK 678
 statusnc 494, 498
 stockage
 déstructuré 546
 structuré 546
 STORAGE 617
 initial 617
 next 617
 optimal 639
 pct_increase 617
 Structured Generalized Markup Language 515
 supprimer une instance 85
 swap 718
 Sybase 387
 synonyme 772
 SYS 180, 269, 663, 772
 SYSAUX 26, 256, 257, 608
 SYSDBA 161, 666, 668, 772
 SYSOPER 161, 666, 668, 772
 SYSTEM 26, 38, 45, 180, 256, 269, 608, 663, 772
 System Global Area 60, 108
 System Parameter File 654
 système de fichiers_
 ext 718
 ext2 718
 hpfs 718
 iso9660 718
 minix 718
 msdos 718
 nfs 718
 swap 718
 vfat 718

T

TABLES 394
 TABLESPACE 256, 394, 606, 773
 BIGFILES 626
 EXTENT MANAGEMENT
 DICTIONARY 613
 EXTENT MANAGEMENT
 LOCAL 613
 LOCAL AUTOALLOCATE 615
 LOCAL UNIFORM SIZE 615
 PERMANENT 256, 257, 268
 SYSAUX 26, 608
 SYSTEM 26, 256, 608
 TEMPORARY 257, 625
 UNDO 174, 257, 636
 TCL 684
 TCP/IP 285, 773
 TCP/IP Protocol Support 818
 TEMPORARY TABLESPACE 257, 625
 Terminal Server 53, 773
 Text 275
 thread 40, 59, 62, 64
 ARCH Archiver 112
 CKPT Checkpoint 108
 DBWR Database Writer 108
 Dnnnn Dispatcher 113
 LGWR Log Writer 108
 listener Net 112
 PMON Process Monitor 108
 Pnnnn Parallel Query Slave 113
 RECO Recover 112
 SMON System Monitor 108
 Snnnn Server 113
 SNPnn Snapshot 112
 timed_statistics 141
 TKPROF 752
 TNSNAMES.ORA 288, 773
 transaction
 atomicité 172
 cohérence 172
 COMMIT 173
 début et fin 173
 durabilité 172
 isolation 172
 lecture consistante des données 175
 lecture seule 176

notion de ~ 172
 pose de verrous 177
 read only 176
 ROLLBACK 173
 SAVEPOINT 173
 transaction distribuée 468
 Trigger 382
 Voir déclencheur Java 507
 TSE 53, 773
 tuning 729, 773
 Oracle9i 742
 Windows 730
 two-tasks 115

U

udump 152
 UNDO 174, 257
 UNDO TABLESPACE 129, 141, 636
 UNDO_MANAGEMENT 129, 141
 UNICODE 217, 250, 805
 Unified Scheduler 24
 Unix 3
 URL 773
 US 24
 USER (vues commençant par ~) 182
 USER_ 182
 user_dump_dest 140
 USERID 394
 UTF-16 805
 UTF-8 217, 250, 805
 utilisateur Windows 86
 Utl_compress 32

UTL_FILE 366, 368
 UTL_HTTP 366
 UTL_MAIL 31, 370
 UTL_SMTP 370
 utlbstat 737, 752
 utlbstat.sql script 181
 utlchain1.sql script 181
 utlestat 737, 752
 utlestat.sql script 181
 utlhttp.sql script 181
 utllockt.sql script 181
 utlrp.sql script 181
 utlsampl.sql script 181
 utlxplan.sql script 181

V

V\$ (vues commençant par ~) 182
 V\$LOGMNR_CONTENTS 646
 valeur 44
 verrou 177
 LOCK TABLE 178
 SELECT...FOR UPDATE 178
 vfat 718
 Visual Basic 314
 Visual C++ 314

W

WE8ISO8859P15 217, 250
 WE8MSWIN1252 217, 250, 836
 WebDAV 247, 773
 WebDB *Voir* Oracle Application Express XLVIII, 593
 Windows 2000 versus Linux 35
 Windows Terminal Server 53
 Windows XP Professionnel 828

Windows XP Standard 828
 Windows 2000 Support Tools 53, 54
 Wintel 773
 WITHIN 555
 WORKAREA_SIZE_POLICY 132
 Workload Repository History 678
 workspace *Voir* Oracle Application Express 602
 WSH 773
 WWW 773

X

XA 280
 XDK 558
 XE 827
 Voir Oracle Express Edition 827
 XML (eXtensible Markup Language) 381, 515
 XML Class Generator 574
 XML DB 247, 525
 XML Development Kit 818
 XML SQL Utility 582
 XMLQUERY 773
 XPATH 773
 XQL 773
 XSL (eXtensible Stylesheet Language) 521, 773
 XSLT 773

Z

Zend Core for Oracle 836