CHAPITRE 1 -concepts de base

SGBDs et SGBDRs

Un SGBD est un système complexe permettant de gérer de manière efficace, un volume important de données structurées, accessible par des utilisateurs simultanés locaux ou non. Un SGBDR (ou RDBMS en Anglais) est un SGBD basé sur le modèle relationnel.

Une relation est un sous-ensemble de données caractérisé par des attributs et visualisable sous forme de table. Bien que cela ne soit pas vraiment formalisé, envisageons d'un point de vue pragmatique quelles doivent être les caractéristiques d'un 'vrai' SGBD par opposition à des systèmes de gestion de fichiers plus ou moins évolués comme Paradox, Omnis, fileMaker, Access…

Caractéristique d'un vrai SGBD

Un SGBDR digne de ce nom doit disposer d'un certain nombre de caractéristiques minimales :

* gestion de gros volumes de données (en consultation et en mise à jour),

* Sécurité des données, qui se décline en :

* disponibilité

Un SGBDR se doit d'offrir une bonne disponibilité des données. Une disponibilité totale des données est possible (temps de reprise nul) il suffit de s'en donner les moyens logiciels et matériels…

* fiabilité

Des mécanismes de sauvegarde variés (physique, logique, off-line, on-line, totale, partielle, incrémentale), ainsi que des mécanismes de journalisation, et de reprise permettent de restaurer une information sans pratiquement aucune perte, dans tous les cas de problème matériel ou logiciel.

* confidentialité

Tout n'est pas accessible à tout le monde! Se connecter à la base de données, donne un certain nombre de droits et de ressources en fonction d'un profil défini et maintenu par un administrateur. La granularité d'accès peut aller jusqu'à la vision unique d'un champ d'un enregistrement d'une table particulière. Encore une fois on ne manipule plus des fichiers…

* cohérence

Que les données soient réparties ou non –dans ce dernier cas les mécanismes mis en jeux seront plus complexes– elles doivent être cohérentes. Cela sous entend, d'une part que les accès concurrents d'utilisateurs, notamment lors de mises à jour, ne doivent pas compromettre l'intégrité des données et d'autre part que ces dernières satisfassent aux contraintes d'intégrité du modèle, mais aussi aux règles de gestion de l'entreprise.

* traçabilité

On peut, notamment en cas de problème important ou en cas d'attaque du système, recourir à l'analyse de traces ou de logs du SGBD. Le niveau de détail de ces traces est paramétrable, et concerne soit les aspects système, soit réseau, soit l'accès aux données élémentaires elles-mêmes.

* concurrence d'accès en lecture et écriture (avec une bonne granularité),

* gestion (efficace) des transactions,

* portabilité sur différents OS, des données et du code

* administrabilité :

Existence d'outils d'administration généraux : gestion des données, des utilisateurs, des fichiers physiques, des espaces logiques, des droits, des profils, des ressources systèmes, etc.

- existence d'outils de surveillance

En temps réel grâce à un moniteur, si possible graphique ou en temps différé grâce à des journaux ou à des traces paramétrables

- possibilité d'export import :

De, ou vers des fichiers "texte", des logiciels bureautiques ou des fichiers gros systèmes par exemple

- optimisation de performances :

Offrir de bonnes performances et des outils permettant de les mesurer et de les contrôler via des paramètres de configuration. Des processus d'optimisation en temps réel des requêtes complexes sont également souvent présents. Les données peuvent être indexées, de manière souple, dynamique et complète (index simples, concaténés, multiples, tables de hashage, recherche textuelle, etc.). Un nombre important d'utilisateurs, ainsi qu'un volume conséquent de données peuvent être pris en compte.

Caractéristiques du relationnel

* modèle sous jacent simple, bien formalisé et éprouvé,

* accès simple via un langage de requêtes SQL

* normalisé : Comme dans toute norme, elle est enrichie par les différents fournisseurs de logiciels. Mais si l'on se contraint à écrire des procédures respectant la norme ISO/ANSI 92 par exemple, on est quasiment garanti de pouvoir porter ce code sur Oracle, Informix ou Sybase…

* opérateur de projection (ou sélection), restriction, produit cartésien, union, différence,

* indépendance entre stockage physique et vision logique des données :

Les données (le plus souvent des tables) sont référencées de manière logique. Un dictionnaire de données permet de retrouver la correspondance avec l'objet physique désiré. Ceci est bien sûr très utile dans les environnements ouverts, et offre une grande souplesse aussi bien lors du développement, que lors de la mise en production ou dans la phase de maintenance des applicatifs. La conséquence est que l'on pourra déplacer physiquement des données, par exemple les changer de serveur, renommer ou retailler un fichier sans pour autant retoucher le code des applications. Il n'y a pas de correspondance bijective entre un fichier et une table.

* existence de contraintes d'intégrité

- intégrité de domaines (contrôle du type de la donnée)

- intégrité de relation (existence d'une clé primaire : unique et toujours définie (non nulle))

- intégrité de référence (contrôle de la cohérence d'attributs de tables différentes lors des mises à jour

Architectures locales et réparties

On peut distinguer essentiellement 3 types d'architecture globale de systèmes d'information basés sur Oracle :

* architecture locale

Tout est sur le même serveur matériel, programme client et serveur de données Oracle

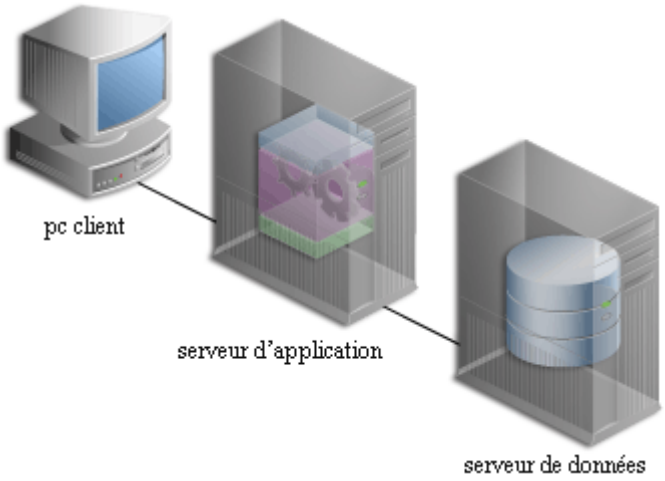
* architecture client/serveur

On a un programme client (un exécutable) sur le poste client, dit alors 'client lourd'. Le PC communique avec le serveur de données sur un serveur distant, via le réseau, et la couche Oracle Net.

* architecture 3 tiers

Pas de programme client. Un navigateur suffit sur le poste de travail (dit alors 'client léger'). Il dialogue avec le serveur de données distant via http.

Note : il existe également des architectures n- tiers plus anecdotiques, mais



conceptuellement intéressantes

Exemple d'architecture 3-tiers

Bases, instances et serveurs de données

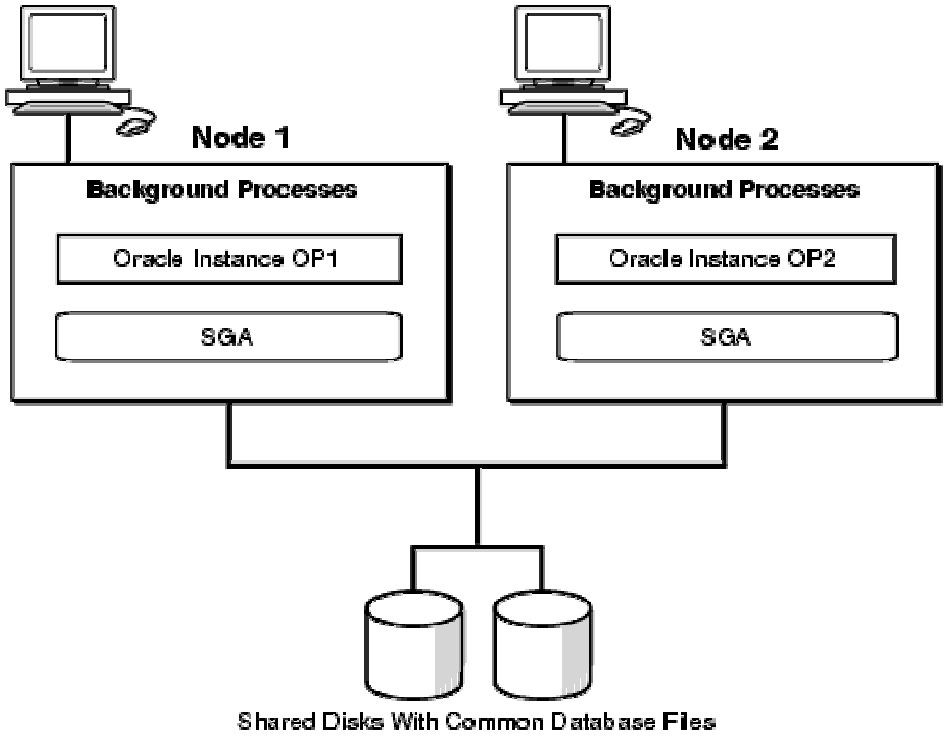
Une base de données Oracle est un objet riche et complexe composé physiquement d'une INSTANCE (zone mémoire partagée + processus de fond) et d'un certain nombre de FICHIERS PHYSIQUES, totalement indépendants du nombre de table (vrai SGBDR!).

Une base de données peut héberger des centaines, voire des milliers, d'applications, d'utilisateurs et de tables.

Un ensemble de tables (et de structures de données connexes) fonctionnellement dépendantes et appartenant au même utilisateur s'appelle us SCHEMA (parfois un COMPTE) Oracle.

Pour fixer les idées une 'DATABASE' MySQL correspond à un SCHEMA Oracle.

Note : dans la pratique on confond généralement la notion d'instance Oracle avec la notion de base de données Oracle. En théorie, une base de données peut être partagée par plusieurs instances, mais cela est assez rare. Cela permet de faire une répartition de la charge ou d'introduire de la redondance, pour pallier à des problèmes éventuelle de panne de CPU…



Exemple de 2 instances qui se partagent une base de données physique

Un serveur de données 'au sens Oracle) est l'ensemble : instance(s) + base de données physique

Typologie des utilisateurs et des bases

Les différentes catégories d'utilisateurs

En pratique il existe 5 catégories d'utilisateurs, qui utilisent différents outils en fonction de leur profil et du SGBD ciblé. Ceci est résumé dans le tableau suivant :

Type d'utilisateur	**profil**	**outil**	**ex Oracle**	**exemple MySQL**
Utilisateur final	non informaticien	application client serveur ou web, ou progiciel, ou appli bureautique, outil d'aide à la	Excel, appli maison, Oarcle application , B.O	appli web

		décision		
utilisateur final évolué	non informaticien éclairé	client QBE (Query by Example)	MS QUery	PHPMyAdmin
Développeur	Technicien ou ingénieur	Bloc note, environnement de développement, Atelier de développement	TOAD, designer, webdb	textpad
Administrateur application	Utilisateur final hiérarchiquement privilégié	l'application elle même, via un accès privilégié	NA	NA
DBA	Technicien ou ingénieur	langage SQL, console d'admin texte, console graphique, console web	sqlplus, OEM CS ou web, TOAD,	PhpMyAdmin

Outre le moteur du SGBD lui même, les différents fournisseurs proposent toute une suite de logiciels connexes permettant ces interactions, qui représente une part non négligeable de leur activité.

Les différentes catégories de base de données

La base de données, à l'instar d'un programme classique s'inscrit dans un cycle de vie, et différentes occurrences de la base permettent de 'coller' aux différentes phases du projet.

On distingue principalement :

* la phase de conception, qui produira le modèle conceptuel et le modèle logique de la base

* la phase de développement, qui verra la création et l'évolution de la base de développement (Base n°1)

* les tests, avec une base de test ou de pré production (Base n°2) qui permettra outre de tester les programmes de manière approfondie, de tester la montée en charge et les performances en terme de volume de données, et de concurrence d'accès

* la phase de production / exploitation et de maintenance, avec une base de production (Base n°3), initialisée avec des données réelles et dimensionnée de manière adéquate

Le DBA sera fréquemment amené à transférer des données d'une base à une autre :

* de la base de développement à la base de Test, pour effectuer les dits tests. Un ensemble de données cohérentes (jeu d'essai) est alors nécessaire, ce qui n'est pas toujours facile à obtenir vu la complexité croissante des applications et peut nécessiter des outils spécifiques d'extraction de données

* de la base de test à la base de production, pour la mise en exploitation réelle

* de la base de production vers la base de test ou de développement, pour être sûr d'avoir des données cohérentes et pertinentes (puisque réelles).

Ce dernier transfert de données est VIVEMENT déconseillé, d'abord parce qu'il manipule des volumes de données inutiles (la totalité des données réelles) et ensuite car il offre d'importantes failles de sécurité, les systèmes de test et de développement étant par définition moins protégés que les systèmes en production…

Note : le passage de la base de test à la base de production, nécessite de réinitialiser la base, avec les scripts SQL originaux (création et chargement initial). Le passage au serveur réel peut s'avérer délicat et contraignant (environnement matériel et logiciel différent).

CHAPITRE 2 -Outils utiles

Logiciels utiles

Pour pouvoir faire les exercices, et pratiquer à minima l'administration Oracle, il vous faudra installer le logiciel Oracle 10g, et créer une base par défaut.

Attention!

Avant d'installer un de ces logiciels sur votre poste, lisez très attentivement le début de la doc d'installation et notamment les prérequis matériels et logiciels.

Il vous faudra suivant le cas 1GO sur disque et de 512MO à 1GO de mémoire pour que le logiciel s'installe et fonctionne.

Oracle Database 10g Express

Une version light ET GRATUITE d'Oracle pour Linux et Windows. Totalement fonctionnelle et assez peu gourmande en ressources. Données limitées à 4GO !

S'installe facilement, Idéale pour démarrer.

Toutes les infos sur Oracle Database 10g Express et le logiciel à télécharger, sont disponibles ici :

http://www.oracle.com/technology/products/database/xe/index.html

Oracle Database 10gR2 Enterprise Edition

La version professionnelle standard du serveur de données version 10gR2.

Toutes les infos (et le téléchargement) sont ici :

http://www.oracle.com/technology/products/database/oracle10g/index.html

Oracle SQL developper v 1.2.1

Un outil de développement graphique, gratuit.

Il vous permet de faire du SQL, du PL/SQL, de naviguer dans la base de données et de produire des rapports personnalisés.

Toutes les infos concernant ce produit, et le logiciel à télécharger, sont disponibles ici :

http://www.oracle.com/technology/products/database/sql_developer/index.html

CHAPITRE 3 -Le 'D.B.A' ?

Un Administrateur de bases de données ?

D.B.A. = Data Base Administrator = Administrateur de bases de données. Il est responsable du bon fonctionnement des bases de données de l'entreprise (bases de développement, test et production).Les tâches du DBA :

* Installer le logiciel, faire les mises à jour (patchs, changement de version du noyau mais aussi des autres produits),

* Créer les bases de données physiques et gérer l'espace physique,

* Gérer les utilisateurs et leurs droits d'accès,

* Valider les schémas de données (cohérence, non redondance, optimisation)

* créer les objets du schéma : tables, index, vues, …

* Assurer la sécurité de la base (sauvegarde, restauration, confidentialité d'accès),

* surveiller le système, gérer les ressources et optimiser les performances,

* Faire les transferts de données de et vers d'autres systèmes,

* contacter le support technique…ou Google

Son travail peut être assimilé à celui d'un responsable système, Oracle pouvant être perçu comme un 'sous' système d'exploitation.

Les utilisateurs d'une base Oracle :

* le dba lui même

* le développeur d'application

* l'administrateur d'application

L'utilisateur final (normal ou infocentre / web)

Les interlocuteurs du DBA :

* les développeurs

* les administrateurs système (unix/linux, windows, etc.)

* le support technique Oracle

* le support technique interne

* l'administrateur sécurité (s'il existe)

et plus rarement …les utilisateurs

Les 'privilèges' DBA :

Un DBA pour pouvoir travailler doit posséder un certain niveau de privilèges

* au niveau OS

* au niveau de la base de données

Ils seront détaillés dans un autre chapitre…

CHAPITRE 4 -Installation

Environnement système Oracle minimal

Au minimum et ce quel que soit le système d'exploitation, on doit positionner le répertoire de référence d'Oracle : ORACLE HOME DIRECTORY et le nom ou l'identificateur de l'instance sur laquelle on veut travailler : SID ORACLE.

Ceci est donné par 2 variables d'environnement : ORACLE_HOME et ORACLE_SID

Note : L'installation du logiciel se fait par défaut dans un répertoire normalisé compatible avec l'architecture flexible d'Oracle (ORACLE FLEXIBLE ARCHITECTURE ou OFA)

Sous windows : disque:\Oracle\product\<no_version>\db_n

Variables d'environnement système Unix / Linux Oracle

Pour vérifier les variables positionnées dans l'environnement courant :

$> env|grep ORA

Pour positionner les variables d'environnement utiles :

$> export ORACLE_HOME = /oracle

$> export ORACLE_SID = TEST

$> $ORACLE_HOME/bin/sqlplus scott/tiger

Remarque : EXPORT est obligatoire si l'on veut que ces variables soient aussi accessibles aux process fils du process courant, ce qui peut être utile quand on lance une commande SHELL comme 'sqlplus' puisque cela déclenche…un process fils !

Variables d'environnement système Windows / Oracle

Lors de l'installation d'Oracle 10g ou 11g, la base de registre Windows est normalement mise à jour et permet de se connecter à la base par défaut sans préciser de variables d'environnement.

Les variables de configuration d'Oracle sont visibles par 'regedit' dans la clé :

HKEY_LOCAL_MACHINE / SOFTWARE / ORACLE / KEY_<ORACLE_HOME>

On peut éventuellement positionner des variables le temps d'une session :

<ORACLE_HOME> ayant la valeur de la home directory d'Oracle choisie lors de l'install…

C:>SET ORACLE_HOME = /oracle

C:> SET export ORACLE_SID = TEST

C:> $ORACLE_HOME/bin/sqlplus scott/tiger

Note : la base de registre est normalement mise à jour à l'installation d'Oracle 10g et permet de se connecter à la base par défaut sans préciser ces variables d'environnement.

Attention ! Certains outils Oracle comme 'emctl' par exemple ont parfois besoin d'avoir le ORACLE_SID défini dans l'environnement en plus d'être défini dans le registre dans la sous clé 'Oracle'

Rappel sur les variables d'environnement WINDOWS :

Il y a les variables 'User' connues uniquement du user courant, et les variables 'systèmes' connues de tous les utilisateurs.

Les variables USER sont initialisées ou consultées

* automatiquement au démarrage de) Windows (Windows récupère les valeurs définies dans la base de registre dans la clé : HKEY_CURRENT_USER / Environment

* manuellement en utilisant les 'propriétés' du poste de travail (clic droit sur poste de travail / Propriétés /Avancées / Variables d'environnement

Les variables système peuvent être initialisées (ou consultées) de différentes manières

* automatiquement au démarrage de Windows (Windows récupère les valeurs définies dans la base de registre dans la clé : HKEY_LOCAL_MACHINE / SYSTEM / CurrentControlSet / Control / SessionManager / Environment

* lors de l'installation de logiciels qui utilisent alors les fonctions de l'API Windows

* manuellement en utilisant les 'propriétés' du poste de travail (clic droit sur poste de travail / Propriétés /Avancées / Variables d'environnement

Que ce soit à travers la base de registre ou via les propriétés du poste de travail, les modifications sont pérennes (ie après un reboot)

Par contre les variables qui seraient définies dans une fenêtre de commande par une commande SET (SET ORACLE_HOME=c: \oracle , par exemple) ne durent pas au dela de la session.

Autres variables utiles

* le chemin d'exécution (PATH)

Pour plus de confort, on étendra la variable PATH courante pour accéder directement aux exécutables d'Oracle, qui se trouvent dans le sous répertoire BIN de la HOME DIRECTORY d'Oracle, sans spécifier de chemin absolu.

Sous Unix : $> export PATH=$PATH:$ORACLE_HOME/bin

Sous Windows : C:> PATH = %PATH%;%ORACLE_HOME%BIN

*l'identificateur de terminal X (DISPLAY)

Sous Unix / Linux, cette variable est nécessaire pour pouvoir utiliser des outils graphiques.

L'installeur d'Oracle ou la console d'administration en sont des exemples vitaux!

$> export DISPLAY = @IP_terminal:0.0

Note : en cas de problème vérifier / positionner les autorisations d'accès au serveur X (commande xhost) et pour les distributions récentes de Linux, vérifier que le serveur X est en mode 'listen' !!

Fichier d'environnement Unix / Linux

Sous Unix, le script $ORACLE_HOME/bin/oranenv permet de positionner l'environnement de manière semi-automatique. Il suffit de renseigner la variable ORACLE_SID et le reste est mis en place par le script.

Par défaut une valeur de ORACLE_SID est demandée par le script, même si une valeur courante est déjà définie, sauf la la variable 'ORAENV_ASK' est positionnée à 'NO'.

Autres fichiers / exécutables utiles

nom	nom unix	nom windows	fonction

SQL+	sqlplus	sqlplusw.exe sqlplus.exe	Interpréteur SQL/PL standard
noyau oracle	oracle	oracle.exe	le sgbd lui même
oradim		oradim.exe	création d'instance
Oracle Net	lsnrctl	LSNRCTL.EXE	le gestionnaire de serveur Oracle Net
export	exp	exp.exe	export de données de base a base
import	imp	imp.exe	import de données de base a base
loader	sqlldr	sqlldr.exe	import de fichiers texte
tnsping	tnsping	tnsping.exe	test de configuration Oracle Net
oraenv	oraenv		affectation de l'environnement
orapwd	orapwd	orapwd.exe	gestionnaire de fichier d'authentification
ora error	oerr		description des messages d'erreurs
startup	dbstart		démarrage de la base
shutdown	dbshut		arrêt de la base
assistant creation	dbca	dbca.bat	utilitaire de création semi-automatique

Installation Oracle 10g

Toutes les documentations officielles concernant Oracle 10g sont disponibles sur Oracle Technical Netwoirk (OTN) ici :http://www.oracle.com/pls/db102/homepage

L'essentiel de l'installation d'Oracle 10g pour Windows XP est disponible ici dans le **Oracle 10g Quick installation guide for windows (format PDF)**

L'essentiel de l'installation d'Oracle 10g pour Linux est disponible ici dans le **Oracle 10g Quick installation guide for windows (format PDF)**

Une Installation réussie d'Oracle tient en 1 phrase:

- Vérifier et respecter strictement les pré-requis matériels (disque temporaire, disque d'installation, place pour la base à créer) et logiciels (compatibilité de l'OS, , librairies dynamiques et packages obligatoires).

Il est donc très important par exemple de vérifier la matrice de compatibilité fournie par Oracle, pour votre Linux et la 10g.

A priori la version recommandée est un 'advanced Application Server' de Redhat. Installer sous debian, Mandrake ou FEDORA est en pratique possible mais peut s'avérer délicat et nécessite une bonne maîtrise de Linux.

CHAPITRE 5-Outils d'administration

Console Enterprise Manager (em)

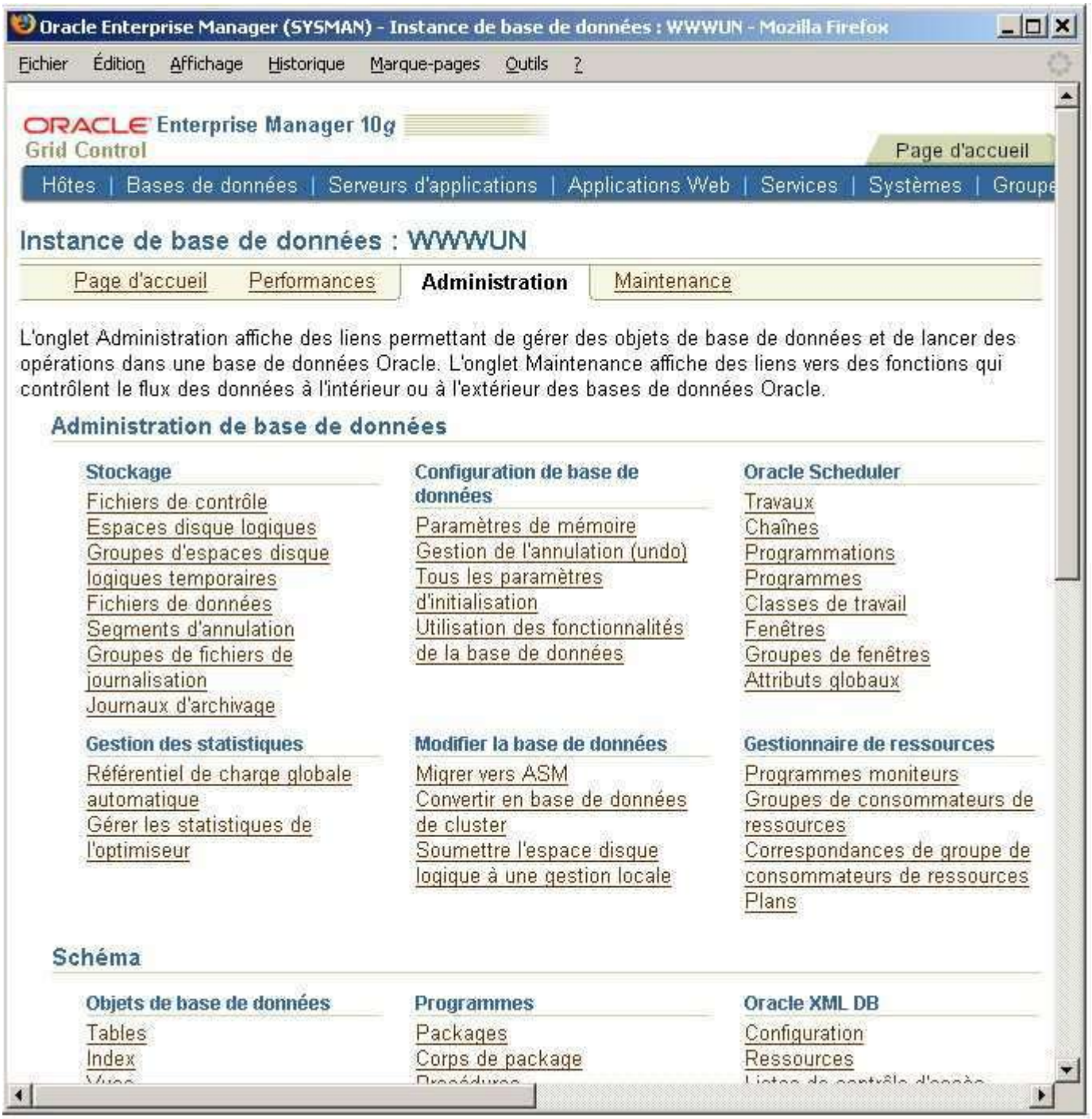
Oracle 10g est livré avec une console d'administration graphique : Oracle Enterprise Manager (OEM ou EM), accessible depuis un client léger, PC ou poste avec navigateur et TCP/IP.

Le port d'écoute par défaut est 5500, ou 1158 suivant les version
On peut vérifier cette information dans ORACLE_HOME/install/portlist.ini
exemple d'URL :
http://localhost:5500/em

Note : La base et le serveur Oracle Net (même en local!) doivent être démarrés, avant de lancer la console.

Note : lorsque la base ou le serveur Oracle Net n'est pas démarrée on obtient un écran général d'information d'OEM, qui indique la non disponibilité de la base, mais pas de fenêtre de login… OEM est attaché à une base particulière. On dit qu'elle fonctionne en STANDALONE, à la différence de consoles centralisées multi-bases comme Oracle Server Manager ou la GRID console.

OEM utilise cependant un référentiel dans la base, à travers 2 schémas SYSMAN et DBSNMP.



Exemple de page d'accueil de la console GRID

Gestion en mode commande (Unix ou Windows)

La commande emctl permet de gérer la console. Elle est de la forme
emctl dbconsole
option = start …démarre la console
= stop …l'arrête
= status…donne son statut

exemple sous Windows
C:\\ORACLE10G\\BIN>emctl status dbconsole

Oracle Enterprise Manager 10g Database Control Release 10.1.0.2.0
Copyright (c) 1996, 2004 Oracle Corporation. All rights reserved.
http://DAREDEVIL:5500/em/console/aboutApplication
Oracle Enterprise Manager 10g is running.
——————————————————————
Logs are generated in directory C:\\oracleproduct10.1.0Db_1\\DAREDEVIL_ORCL\\sysman/log

Gestion via les services Windows

Sous Windows le service s'appelle OracleDBConsole et correspond à l'exécutable nmssvc.exe.
On peut gérer ce service avec l'icône service du groupe 'Outils d'administration' ou bien avec la commande NET :

net start OracleOraDb10g_home1TNSListener
net start OracleServiceORCL
net start OracleDBConsoleorcl

Note : sous XP notamment la gestion via les services peut s'avérer plus délicate. On préféra plutôt le mode commande…

SQL pour le DBA

Un certain nombre de commandes SQLPlus, voire quelques SELECT sont très pratiques pour le travail quotidien du DBA.
Ils peuvent être lancés sous SQLPlus, ISQLPLus , ou pour les SELECT sous n'importe quel Outil client et ce quelle que soit la plate forme.

SQL> — consulter les tables du DBA
SQL> SELECT * FROM DBA_%…
SQL> — rechercher des informations dans le dictionnaire
SQL> SELECT * FROM DICT WHERE TABLE_NAME LIKE '%…'

SQL> — connaître les colonnes d'une table
SQL> DESCRIBE nom_table
SQL> — envoyer les resultats d'un SELECT dans un fichier
SQL> SPOOL nom_fic | OFF

SQL> -formater l'affichage des champs LONG (DBA_VIEWS par exemple)
SQL> SET LONGWIDTH n
SQL> — formater l'affichage d'une colonne tronquée
SQL> COL nom_col FOR An

SQL> — executer un script SQL ou PLSQL
SQL> @nom_script_sql

SQL> — modifier les paramètres de la session courante
SQL> ALTER SESSION SET
SQL> — et notamment…prendre l'identité d'un utilisateur
SQL> ALTER SESSION SET CURRENT_SCHEMA = nom_schema
SQL> — vérifier les erreurs d'un script ou d'une procédure stockée
SQL> SHOW ERRORS

ou
SQL> SELECT * FROM USER_ERRORS

SQL> — vérifier le USER courant
SQL> SHOW USER
SQL> — voir les paramètres de la base
SQL> SHOW PARAMETER nom_parametre
SQL> –modifier les paramètres dynamiques de la base
SQL> ALTER SYSTEM SET param = valeur
SQL> — ou l'état de la base…
SQL> ALTER DATABASE…

Lorsque le DBA veut automatiser des scripts, et les rendre dynamiques par rapport au référentiel il peut utiliser du SQL qui génère du SQL (META SQL)

l'Outil iSQL*Plus

SQL et PL/SQL

Oracle 10g propose deux langages (plus ou moins) standards pour manipuler les données de la base : SQL et PLSQL

SQL (structured Query Language)
est LE langage standard et donc normalisé d'interrogation et de mise à jour de tous les SGBDRs. Plus précisément il permet de créer / modifier des structures de données (LDD), mais aussi de consulter / interroger, modifier, supprimer les données contenues dans ces structures (LMD).

PL/SQL (Procedural language)
PL/SQL est un langage qui intègre SQL et permet de programmer de manière procédurale. Il est spécifique à Oracle. Pour SQL Server il existe par exemple un équivallent : TRANSAC SQL
Globalement, avec PL/SQL on aura à notre disposition un vrai langage de programmation structuré :
- gestion de variables, et typage avancé,
- structures conditionnelles (IF / THEN / ELSE),
- structures itératives (WHILE, FOR, LOOP),
- gestion d'erreurs sophistiquée (EXCEPTION),

PL/SQL est certes moins intuitif mais est plus puissant que le SQL. Ces 2 langages sont éminemment complémentaires.

Note : le code PL/SQL peut être stocké (et compilé) dans la base de données sous forme de fonction, procédure, package ou trigger.

Les interpréteurs (PL)SQL d'Oracle
Pour pouvoir utiliser ces langages, les DBAs mais aussi les développeurs ont besoin d'un outil client spécial : l'interpréteur.
Cet interpréteur permet :
- d'exécuter interactivement des ordres SQL et PL/SQL,
- d'exécuter des scripts SQL et PL/SQL
- de personnaliser l'environnement et formater les résultats

SQL*Plus
est l'interpréteur le plus simple. Il existe en mode commande sous Unix/Linus ou MSDOS, et en mode semi-graphique sous Windows.

exemples de commandes :
$> sqlplus SYSTEM/XYZ
$> sqlplus SYSTEM/XYZ@ma_base_distante
$> sqlplus /nolog
$QL> connect SYS/XYZ AS SYSDBA

sous Windows : Menu Démarrer, Programmes, Orahome10g_db1/ Développement d'application / SQLPlus

iSQL*Plus
est le (grand) frère de SQL*Plus. Il est utilisable par un client léger, avec donc un simple navigateur sans besoin d'aucun environnement Oracle client.
Outre le formatage automatique des sorties au format HTML, iSQL*Plus présente le gros avantage de proposer un historique des commandes…

Il est démarré sur un serveur local ou distant, et accessible en http. Le port TCP/IP standard d'écoute est le 5560.

exemples de commande de contrôle

lancement Windows :
BINisqlplussvc.exe -start
arrêt Windows :
BINisqlplussvc.exe -stop
utilitaire Windows (appelle le précédent…) :
BINisqlplusctl.bat
URL d'accès : http://mon_serveur:5560/isqlplus
Rappel des ports utilisés par les outils

Voir ORACLE_HOME/install/**portlist.ini**

Par défaut :
Numéro de port HTTP iSQL*Plus =5560
Port HTTP de la console Enterprise Manager (orcl) = 1158
Port du Listenet TCP/IP : 1521

CHAPITRE 6 -Création de base

Assistant de création 'dbca'

Un assistant de création de base de données est fourni avec Oracle 10g. Celui ci est lancé automatiquement, si on le souhaite lors de l'installation du logiciel.
Cet utilitaire se trouve dans ORACLE_HOME/bin.

Pour le lancer :
- sous Windows : Menu Démarrer, Programmes , Oracle-OraDb10g_home1 , Configuration and Migration Tool
- sous Unix / Linux : $> dbca&

(Attention, un environnement Xwindows opérationnel est nécessaire)

A chaque étape de la création, Oracle propose des valeurs de paramètres par défaut, et il est possible de 'court-circuiter' tous les écrans qui suivent en cliquant sur 'Terminer'.

Les modèles

Les modèles de BDs ou templates, servent à créer, dupliquer, voire cloner facilement des bases de données physiques..
Oracle fournit 3 templates prédéfinis qui satisfont la plupart des cas d'utilisation des BDs en entreprise. Templates pour :
- des entrepôts de données (datawarehouse),
- des bases standards,
- le transactionnel.

Il est bien sûr possible de plus de définir et personnaliser son propre modèle.

Concrètement les modèles sont des fichiers XML, d'extension 'dbc' localisés dans le répertoire ORACLE_HOME/assistants/dbca/templa.
Ci-après un exemple d'extrait de fichier pour une base standard :
General_Purpose.dbc

Ils contiennent essentiellement des valeurs pour :
- les options de la BD,
- les paramètres d'initialisation,
- les attributs d'allocation de fichiers et de tablespaces.

Note : les templates peuvent également contenir les fichiers de données et redolog de la base, dans un format compressé ('.dfj') ou non ('.dfb')

Principales options de création

Outre les templates, dbca permet de faire un certain nombre de choix importants :
- le type d'administration : locale (database control) ou centralisée (grid control)
- le type de stockage : via le gestionnaire de fichier OS (file system) ou géré automatiquement par une instance spécifique Oracle (Automatic Storage Management / ASM)
- le type de restauration : 'Flash Recovery' ou via des archives de journaux
- le jeu de caractère utilisé pour le stockage des données

CHAPITRE 7 -DBAs et privilèges

DBAs et privilèges d'exploitations

Certains 'administrateurs' : les opérateurs et techniciens d'exploitation, ou 'exploitants' pour faire court, n'ont pas forcément besoin d'un niveau de privilège DBA.

Les opérations concernées sont par exemple :

* les démarrages / arrêts,

* les sauvegardes / restaurations,

* la planification et l'exécution de batch

Oracle fournit 2 niveaux de privilèges, qui peuvent être assimilés à des niveaux de connexion, qui satisfont ces besoins : les privilèges d'exploitation 'SYSDBA' et 'SYSOPER'.
Comme tout accès privilégié il s'acquiert via un processus d'identification / authentification.

Authentification locale au niveau du système d'exploitation

C'est une forme d'authentification externe, en ce sens que ce n'est pas Oracle qui contrôle la connexion grâce à son référentiel interne. On se connecte directement (via telnet ou ssh par exemple) au système qui héberge le serveur de données, puis à la base locale, sans plus faire intervenir le réseau.
Ce type de connexion originale ne nécessite pas d'identifiant ni de mot de passe Oracle, mais d'être un utilisateur privilégié au niveau O.S.

Note : un utilisateur quelconque, non privilégié de la base, peut aussi être défini avec une authentification externe, et se connecter localement avec une commande du type : sqlplus /

On peut dire que dans ce cas la sécurité est déportée au niveau O.S. et qu'Oracle accorde sa 'confiance' aux mécanismes d'identification / authentification de ce dernier.

exemples de connexion avec le client SQL standard :
connexion 'normale'
$sqlplus scott/tiger
connexion avec authentification externe
$sqlplus / as sysdba
$sqlplus / as sysoper

Pour obtenir un 'privilège' d'exploitation il suffit d'appartenir au groupe utilisateur correspondant au niveau système :

privilege	gpe unix	groupe windows
SYSDBA	dba	ORA_DBA
SYSOPER	oper	ORA_OPER

Ces groupes sont créés lors de l'installation, et un administrateur système en 'hérite' automatiquement

Note : Le 'CONNECT INTERNAL' des versions précédentes est définitivement obsolète et a été remplacé par le 'CONNECT SYS AS SYSDBA. Parallèlement il n'est plus possible de se connecter SYS 'tout court' sans préciser 'AS SYSDBA'.

Authentification distante au niveau du système d'exploitation

Elle présente les mêmes caractéristiques que précédemment sauf que la base est située sur une machine distante de la connexion système courante.

La syntaxe de connexion devient donc :
$ sqlplus /@nom_base_distante AS SYSDBA (ou SYSOPER)

Un paramètre d'initialisation de la base : REMOTE_OS_AUTJENTICATION=TRUE autorise cette fonctionnalité.

Note importante : il est vivement conseillé pour des raisons de sécurité d'invalider cette possibilité.

Authentification via fichier de mots de passe
Dans ce cas de figure, les privilèges seront contrôlés à partir d'un fichier de mot de passe cryptés local.
exemple de création du fichier :
$ ORAPWD FILE=monfic PASSWORD=monpasse ENTRIES=100
avec
PASSWORD : le mot de passe de SYS
ENTRIES : le nb mas d'utilisateurs référencables dans le fichier.

On peut ensuite créer un utilisateur TOTO avec mot de passe TUTU et que le DBA lui donne le privilège oracle (et non pas système cette fois) nécessaire : SYSDBA ou SYSOPER :

$> sqlplus / AS SYSDBA
SQL> CREATE USER TOTO IDENTIFIED BY TUTU;
SQL> GRANT CREATE SESSION TO TOTO (qu'il ait le droit de se connecter quand même…)
$ GRANT SYSDBA TO TOTO (et lui donner le privilège d'exploitation qui va bien)

Note : le paramètre d'initialisation REMOTE_LOGIN_PASSWORDFILE doit être à EXCLUSIVE (c'est le défaut) pour pouvoir utiliser et modifier le password file

DBAs et privilèges Oracle

Un utilisateur Oracle (déclaré au sein de la base, à distinguer de l'utilisateur au niveau OS) peut être DBA.
Il a tous les 'privilèges système' AU SEIN DE LA BASE, et le droit de les transmettre (ADMIN OPTION)
Grace à quoi, il peut essentiellement :
* consulter et mettre à jour (SELECT, UPDATE, INSERT, DELETE) toutes les données utilisateur de la base
* créer, modifier des structures de données utilisateur (CREATE, ALTER, DROP) n'importe ou (ANY TABLESPACE)
* gérer des utilisateurs et des droits (CREATE/DROP USER, GRANT, REVOKE)
* consulter la totalité du dictionnaire
* exécuter des ordres d'administration purs (CREATE DATABASE, DATAFILE, TABLESPACE)

Il y a 2 utilisateurs privilégiés prédéfinis, SYS et SYSTEM (dont les mots de passe sont définis à la création de la base ou par 'ORAPWD')
Ils sont tous les 2 DBA, mais SYS est plus privilégié en ce sens qu'il est propriétaire des tables et des vues du dictionnaire.

Il existe un ensemble de privilèges (ROLE) prédéfinis nommés 'DBA' qui donne les privilèges nécessaires à un DBA.

Après avoir créé un utilisateur 'normal' il suffit de lui donner ce rôle pour en faire un DBA :
SQL> GRANT DBA TO

Note : Il existe un autre rôle prédéfini, parmi quelques dizaines, qui est également intéressant c'est le rôle 'SELECT_CATALOG_ROLE'. Il est souvent utilisé par des progiciels ou applicatifs utilisant Oracle pour récupérer des méta-données.

Pour plus d'infos sur les droits, rôles et privilèges voir les chapitres correspondants.

DBAs et privilèges système

La fonction de DBA, nécessite des privilèges au niveau Système d'exploitation :
- pour l'installation,
- la maintenance,
- la gestion et l'exécution de batchs, de scripts (SQL ou shell),
- les sauvegardes / restauration

Sur Unix / Linux :
Il existe un user Unix nommé 'oracle' et un groupe associé nommé 'dba'.
Tous les fichiers Oracle, appartiennent à l'utilisateur Oracle.

On peut (doit ?) créer autant d'utilisateur Unix que de DBAs dans l'entreprise ; dba1, dba2, appartenant au groupe 'dba'.Ceci permet d'éviter les recouvrements et d'avoir une meilleure traçabilité.
On évitera de travailler connecté en tant qu'utilisateur 'oracle' pour éviter toute erreur de manipulation des fichiers Oracle.

Note : les programmes et processus, qui constituent le cœur d'Oracle, s'executent en tant qu'oracle, et ont conséquemment les droits nécessaires pour écrire dans les fichiers de données, journaux, archives, etc.

Le DBA et le super utilisateur 'root' :
lors de l'installation, il est nécessaire d'écrire dans certains répertoires protégés du système (/etc par exemple) ou d'exécuter certaines taches privilégiées.
Cependant l'installation se fait bien en tant qu'Oracle, et l'installeur demande simplement le privilège root pendant la période nécessaire à ces opérations.
Il exécute 2 scripts autonomes root.sh et rootpre.sh, en tant que root.
Il retourne ensuite en mode 'normal'.
En production les no de ports TCP/IP utilisés par Oracle 10g sont tous > 1024, et ne nécessitent donc pas de privilèges particuliers.

Sur Windows :
Le principe est plus simple. L'install se fait en général en tant qu'administrateur système.

CHAPITRE 8 -Démarrage/arrêt base

Fichier d'initialisation

Lors de son démarrage, une instance Oracle lit ses paramètres dans un fichier, dit de démarrage ou d'initialisation.
Ce fichier peut être de 2 types :
- un 'init.ora', fichier "texte" simple en lecture seule
- un 'sp file', fichier de paramètres serveur, binaire, en lecture / écriture.

Le fichier INIT.ORA

Il est éditable avec un éditeur de texte standard. Les modifications faites en son sein sont prises en compte lors du redémarrage de la base. Son contenu est de la forme suivante :

commentaire
exemple de fichier init.ora
#
=
= (, , …)
ou
exemple de init.ora référençant simplement un spfile :
#
SPFILE='spfile.ora'

Les noms et emplacements standards de ce fichier INIT.ORA sont les suivants :
sur Unix/Linux $ORACLE_HOME/dbs/init$ORACLE_SID.ora
sur Windows %ORACLE_HOME%\\database\\init%PRACLE8SID%.ora

Note : il existe un fichier init.ora d'exemples (sample init.ora file) abondamment commentés sur chaque plate-forme installée.
Il se trouve
sur Unix/Linux dans $ORACLE_HOME/dbs/
sur Windows dans %ORACLE_HOME%\\admin\\samplepfile

Le fichier SPFILE

Le fichier spfile, est aussi un fichier externe localisé sur le serveur de données, qui contient les paramètres.

Ceux ci sont persistants, c'est a dire qu'on peut les modifier à n'importe quel moment, et que ces changements resteront pérennes au delà du re démarrage de la base. Ceci se fait avec la commande ALTER SYSTEM.

Exemple :

SQL> ALTER SYSTEM SET DB_FILES=500 SCOPE = SPFILE;

On peut créer un SPFILE ex nihilo avec l'assistant de création de base de données 'dbca' ou à partir d'un 'init.ora' existant avec la commande 'CREATE SPFILE'.

exemple :

SQL> CREATE SPFILE = '' FROM PFILE = ''

Les noms et emplacements standards de ce fichier SPFILE sont les suivants :
sur Unix/Linux $ORACLE_HOME/dbs/spfile$ORACLE_SID.ora
sur Windows %ORACLE_HOME%databasespfile%ORACLE_SID%.ora

Pour savoir quel SPFILE est effectivement utilisé on peut faire sous SQL+ :

SQL> SHOW PARAMETER SPFILE

NAME TYPE VALUE

spfile	string	F:ORACLEPRODUCT10.2.0DB_2 DBSSPFILEORCL.ORA

Paramètres d'instance et de session

Les paramètres d'initialisation

Les ressources utilsées par une base, essentiellement ressources mémoires, buffers, processus et fichiers sont paramétrables, et permettent d'ajuster son fonctionnement, voire de faire de l'optimisation (tuning).
Certains paramètres ont des valeurs libres (dans un domaine de valeurs, bien sûr), d'autres plus restrictifs sont dépendants des systèmes d'exploitation, ou dérivés d'autres paramètres.

Lors de son démarrage, une instance Oracle lit ses paramètres dans un fichier, dit de démarrage ou d'initialisation.
La totalité des paramètres d'initialisation est décrite dans la documentation Oracle Oracle 10gR2 référence (attention PDF de 30MO!).

Note : augmenter les ressources allouées à la base en augmentant les valeurs de certains

Paramètres est une bonne idée…jusqu'à un certain point. La mémoire ne peut pas,on s'en doute, être entièrement dévolue à la SGA…
Quelle que soit la base il est obligatoire de spécifier explicitement au strict minimum 2 paramètres :

'DB_NAME' : le nom de la base de données
'CONTROL_FILES' : la liste des fichiers de contrôle utilisés

Les autres paramètres les plus fréquemment utilisés sont les suivants (S:statique, M:modifiable, D: dérivé d'un autre paramètre)

COMPATIBLE	S	Compatibilité arrière avec les versions précédentes (>= 9.2.0)
DB_DOMAIN	S	Nom de domaine, qui associé au DB_NAME donne un nom de base uniqueNom de domaine,

		qui associé au DB_NAME donne un nom de base unique
DB_RECOVERY_FILE_DEST	M	Répertoire de destination de la zone de FLASH recovery
LOG_ARCHIVE_DEST_n	M	Répertoire des fichiers REDO LOGS archivés
NLS_LANGUAGE	D	Langue utilisée (fonction de NLS_LANG dans l'environnement Unix ou windows)
NLS_TERRITORY	M	le territoire (France par ex). Impacte les dates, et la monnaie
OPEN_CURSORS	M	nb max de cursor ouvrable par une application
PGA_AGGREGATE_TARGET	M	si 0, dimensionnement auto de la PGA, pour les proccess utilisateur server
PROCESSES	S	nb max de process connectés à Oracle (background et user)
SESSIONS	D	= 1,1 * PROCESSES
SGA_TARGET	D	si 0 auto dimensionnement de la SGA
SHARED_SERVER	M	0 si serveur dédié, n = nb de serveurs partagés lancés au startup instance
SP_FILE	S	emplacement du SP File
UNDO_MANAGEMENT	S	Manual | Auto, par defaut Manual indique qu'on utilise des Rollback segments

Paramètres statiques et dynamiques

Certains paramètres peuvent être modifiés en temps réel, ils sont dits dynamiques, d'autres seront pris en compte au prochain redémarrage de la base, ils sont dits statiques.
Les paramètres dynamiques peuvent être modifiés pour la durée de la session,, pour la durée de l'instance ou de manière permanente avec la commande 'ALTER SESSION' ou 'ALTER SYSTEM'.
Voici les syntaxes :

```
ALTER SESSION SET =
ALTER SYSTEM SET = [SCOPE = MEMORY | SPFILE | BOTH] [DEFERRED] [COMMENT = '']
```

exemples

```
SQL> ALTER SESSION SET NLS_LANGUAGE='FRENCH';
```

Informations sur les paramètres

On peut utiliser les commandes SQL*Plus SHOW :

SQL> SHOW PARAMETER

active_instance_count	integer	
aq_tm_processes	integer	0
archive_lag_target	integer	0
asm_diskgroups	string	
asm_diskstring	string	
asm_power_limit	integer	1
audit_file_dest	string	F:ORACLEPRODUCT10.2.0ADMIN ORCLADUMP
audit_sys_operations	boolean	FALSE
audit_trail	string	NONE
background_core_dump	string	partial

background_dump_dest	string	F:ORACLEPRODUCT10.2.0ADMIN ORCLBDUMP
backup_tape_io_slaves…	boolean…	FALSE…

ou en filtrant avec un mot clé :

SQL> SHOW PARAMETER AUDIT

audit_file_dest	string	F:ORACLEPRODUCT10.2.0ADMIN ORCLADUMP
audit_sys_operations	boolean	FALSE
audit_trail	string	NONE

ou utiliser des vues virtuelles du dictionnaire :

V$PARAMETER : les paramètres en cours d'utilisation (de la session donc…)
V$PARAMETER2 : les mêmes, mais mieux présentés…
V$SPPARAMETER : les paramètres du SPFILE (valable au (re)démarrage de l'instance)
V$SYSTEM_PARAMETER : paramètres de l'instance en cours en mémoire

```
SQL> select a.name, a.value courante, sp.value sp_file, mem.value memoire
from v$parameter a, v$spparameter sp, v$system_parameter mem
where a.name=sp.name
and mem.name=a.name
and a.name = 'open_cursors'
```

NAME	COURANTE	SP_FILE	MEMOIRE
open_cursors	1100	1100	1100

```
SQL> alter system set open_cursors=1200;
```
– on modifie la valeur courante de l'instance, ce qui a un impact sur la session
– mais ne sera plus valable au redemarrage de la base (spfile non modifié…)

NAME	COURANTE	SP_FILE	MEMOIRE
open_cursors	1200	1100	1200

```
SQL> — infos sur parametres, par defaut ou non ?
SQL> SELECT NAME , VALUE, ISDEFAULT, DESCRIPTION
FROM V$PARAMETER
```

NAMEtracefile_identifier	VALUE	ISDEFAULTTRUE	DESCRIPTIONtrace file custom identifier
lock_name_space		TRUE	lock name space used for generating lock names for standby/clone database
processes	150	FALSE	user processes
sessions	170	TRUE	user and system sessions
timed_statistics	TRUE	TRUE	maintain internal timing statistics
timed_os_statistics	0	TRUE	internal os statistic gathering interval in seconds

```
SQL> — combine de parametres spécifiés dans le SPFILE ?
SQL> SELECT count(*)
FROM V$SPPARAMETER
WHERE ISSPECIFIED IS TRUE
COUNT(*)
———-
23
```

Démarrage d'une base

Etapes pour démarrer une base :

- se connecter au serveur (niveau OS)
- positionner le 'ORACLE_SID'
- lancer SQL+ sans se connecter (on se connecte à quoi si la base n'est pas démarrée ??

$> sqlplus /nolog
- s'affecter les privilèges SYSDBA
SQL> connect /as sysdba
- lancer la commande de démarrage ou de modification adéquate :
SQL> STARTUP…
ou
SQL> ALTER DATABASE…

Attention !!!! Sous Windows pour se connecter à une base 'Idle' (non démarrée) le Service associé doit obligatoirement être démarré (sans l'instance bien sûr ici…)

Dans le cas contraire, on a une erreur

ERROR:
ORA-12560: TNS : erreur d'adaptateur de protocole

exemple Unix / Linux

```
$ sqlplus /nolog
SQL> connect toto as sysdba
Enter password:
Connected to an idle instance.
SQL>
SQL> show user
USER is "SYS"
SQL>
SQL> startup
ORACLE instance started.
Total System Global Area 285212672 bytes
Fixed Size 1218992 bytes
Variable Size 239076944 bytes
Database Buffers 41943040 bytes
Redo Buffers 2973696 bytes
Base de donnees montee.
Base de donnees ouverte.
SQL>
```

La base dans tous ses états

Une base peut se trouver dans différents 'états', ou niveau de disponibilité qui correspondent généralement à des modes de démarrage :
- 'non montée' (NOMOUNT) : l'instance seule est démarrée. On en peut déclencher que des pseudo connexions de type 'SYSDBA' ou 'SYSOPER'
Les opérations sont assez limitées. Surtout utile (et pour cause) lors de la création de la base.
- 'montée' (MOUNT) : l'instance est lancée, la base n'autorise pas de transactions, ni les connexions utilisateurs normales, mais des opérations d'exploitations portant généralement sur les fichiers
- 'ouverte' (OPEN) : c'est l'état classique, de la base (comme on l'aime) disponible à tous les utilisateurs…
- 'en veille' (QUIESCED) : c'est un état particulier, qui permet au DBA de travailler

tranquillement, sans transactions concurrentes, mais sans avoir à redémarrer la base pour la passer en accès restreint. Dans ce mode, les transactions actives sont momentanément suspendues, jusqu'a l'éveil de la base.

Les options de démarrage :

PFILE = : permet de spécifier un fichier d'initialisation autre que le défaut,
SPFILE = : permet de spécifier un fichier SPFILE autre que le défaut,
FORCE : permet de forcer un démarrage quel que soit l'état de la base, en forçant un arrêt brutal puis un startup
RESTRICT : démarre la base en accès restreint seulement pour les DABS. Aucune autre connexion n'est acceptée. Diffère du mode 'veille' qui autorise d'autres utilisateurs, mais les faits 'hiberner'.
READ : en lecture seule (CQFD) pas d'update, delete, insert, …

Quelques Exemples d'option de démarrage :

```
# un démarrage (très) progressif en 3 temps :
SQL> STARTUP NOMOUNT PFILE=mon_fic.ora;
SQL> ALTER DATABASE MOUNT;
SQL> ALTER DATABASE OPEN;
# un démarrage en comité restreint
SQL> STARTUP RESTRICT
Démarrage automatique
```

Il est bien sûr utile de pouvoir démarrer automatiquement la base lors du démarrage (ou du reboot intempestif de la machine)

Démarrage de la base sous Windows

Sous Windows le + simple est d'utiliser le gestionnaire de services du panneau de configuration pour démarrer le service correspondant à la base désirée.
Le service a pour nom générique : OracleService<ORACLE_SID>

Démarrer le service démarre normalement l'instance également mais ceci dépend en fait d'un paramètre de la base de registre :

Name: ORA_ORCL_AUTOSTART
Type: REG_EXPAND_SZ
De même pour l'arrêt on peut dissocier service et instance :

Name: ORA_ORCL_SHUTDOWN
Type: REG_EXPAND_SZ
Data: FALSE

Remarque : en général le serveur Oracle est accédé via Oracle*Net en client-serveur, dans ce cas ne pas oublier de démarrer également le service 'TNSLISTENER' correspondant au serveur Oracle Net.

Alternativement on peut écrire un script de démarrage (.BAT) faisant appel à l'utilitaire de gestion d'instance d'Oracle : ORADIM
exemple

```
C:\> ORADIM -STARTUP -SID ma_base
– démarre uniquement l'instance mais pas le service…

C:\> oradim -startup -sid -starttype srvc,inst
– démarre le service et l'instance
```

Démarrage par script sous Unix

Oracle fournit un script shell pour le démarrage automatique lors de la procédure de boot du système :
$ORACLE_HOME/bin/dbstart
Ce script s'appuie sur le fichier /etc/oratab normalement créé automatiquement à l'installation (avec création de base) ou manuellement.
/etc/oratab liste les bases présentes sur la machine, et précise si elles doivent être démarrées ou non.
Ce fichier texte contient le ou les nom(s) de base, le ORACLE_HOME correspondant et l'option de démarrage : 'Y' pour 'oui', 'N' pour 'non'.
Une ligne a donc la forme suivante :
|*::Y|N
le caractère '*' veut dire ici toutes les bases…

exemple de fichier oratab généré

```
# /etc/oratab
#
# This file is used by ORACLE utilities. It is created by root.sh
# and updated by the Database Configuration Assistant when creating a database.
# A colon, ':', is used as the field terminator. A new line terminates
# the entry. Lines beginning with a pound sign, '#', are comments.
# Entries are of the form: $ORACLE_SID:$ORACLE_HOME::
#
# The first and second fields are the system identifier and home
# directory of the database respectively. The third filed indicates
# to the dbstart utility that the database should , "Y", or should not,
# "N", be brought up at system boot time.
#
# Multiple entries with the same $ORACLE_SID are not allowed.
#
TEST:/oracle/10gDB2:N
# *:/oracle/10gDB2:N
```

Note : Ce script ne doit pas être exécuté interactivement.

Démarrage du listener

Même remarque que précédemment : comme en général le serveur Oracle est accédé via Oracle*Net en client-serveur, ne pas oublier de démarrer également le process serveur du LISTENER Oracle Net.
Soit par la commande :

```
$> lsnrctl start
```

ou soit en positionnant le paramètre ORACLE_HOME_LISTNER= $ORACLE_HOME avant l'appel de dbstart (dans ce cas dbstart démarre les bases spécifiées dans /etc/oratab ET le listener)

Ou dernière possibilité

```
$dbstart $ORACLE_HOME
# le premier paramètre passé au script sert a renseigner ORACLE_HOME_LISTNER….
```

Démarrage automatique sous linux / Unix

Oracle (et le listener s'il n'est pas protégé par un mot de passe) peuvent être démarrés automatiquement comme un service. L'implantation des fichiers et les 'run level' utilés au boot dépendent des distributions.

Oracle fournit un script générique qui fonctionne pour les principales plateformes : AIX, HP-UX, Linux.
Voici les principales étapes de configuration :

1. se positionner dans le répertoire d'initialisation qui va bien

Platform	Initialization File Directory
AIX	/etc
Linux and Solaris	/etc/init.d
HP-UX	/sbin/init.d

2. créer un fichier /etc/oraclectl comme suit avec les bonnes valeurs de ORACLE_HOME et de compte propriétaire oracle

```
3.  # les 2 commentaires 'chkconfig' et 'description' suivants sont OBLIGATOIRES
4.  # on fournit  liste_run_level priorite_start priorite_stop, ici 2,3,5 et 80 , 20
5.  # chkconfig: 235 80 20
6.  # description: ajout de service auto pour start/stop oracle#!/bin/bash
7.  ORACLE_OWNER="oracle"
8.  ORACLE_HOME="/oracle/db11a"
```

```
9.    case "$1" in
10.         start)
11.              echo -n $"Starting Oracle DB:"
12.              su - $ORACLE_OWNER -c "$ORACLE_HOME/bin/dbstart
    $ORACLE_HOME"
13.              echo "OK"
14.              ;;
15.         stop)
16.              echo -n $"Stopping Oracle DB:"
17.              su - $ORACLE_OWNER -c "$ORACLE_HOME/bin/dbshut
    $ORACLE_HOME"
18.              echo "OK"
19.              ;;
20.         *)
21.              echo $"Usage: $0 {start|stop}"
    esac
```

22. changer le groupe (dba en général) et les permissions du fichier

```
23.   $> sudo chgrp dba oraclectl
    $> sudo chmod 750 oraclectl
```

24. tester le script a la main

    $> ./oraclectl start

25. ajouter le service oraclectl
    Le moyen le plus simple pour ce faire plutôt que de bidouiller les rc0.d , rc1.d, rc2.d …et autres liens symboliques est d'utiliser la commande chkconfig qui va bien (cf paramètres mlis en commentaires en début du script précédent

```
$> sudo chkconfig --add oraclectl
# ou sans utiliser les commentaires du script (on précise explicitement les levels :
$> chkconfig --level 235 oraclectl on
$> #verifier :$> chkconfig -l oraclectl
oraclectl 0:off 1:off 2:on 3:on 4:off 5:on 6:off
$> # Verifier si nécessaire le runlevel courant
$> runlevel
N 2
```

CHAPITRE 9 -architecture physique

Instance et processus

Instance ?
Rappel : Une instance est caractérisée par son identificateur : SID, généralement une variable ORACLE_SID positionnée dans l'environnement.

Une instance active en mémoire ce sont :
- des programmes de fond, services ou processus, qui assurent la maintenance du serveur de données et les entrées / sorties fichiers
- des process server (dédiés ou non à un utilisateur)
- une zone globale partagée : la SGA, qui contient essentiellement du cache de buffers
- des zones mémoires dédiées aux utilisateurs : les PGAs (Private Global Area)

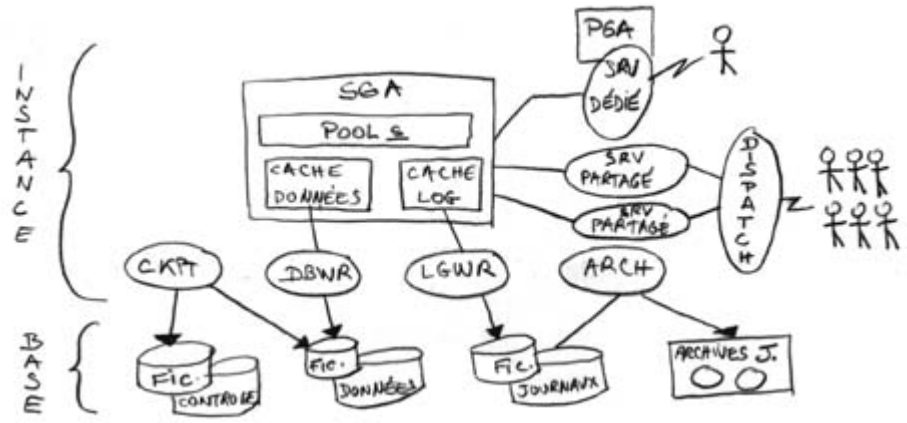


Schéma de l'architecture générale en mémoire

La SGA

La SGA est essentiellement un cache mémoire qui contient des infos partagées de la base.

Voir l'article concernant '**les zones mémoires**'

Les process de fond

Ils permettent de paralléliser et désynchroniser les accès multi-utilisateur à la base.
Nous allons lister les principaux :

DBWn Database Writer
Ecrit les données modifiées, du cache de données de la SGA vers les fichiers de données. On peut en avoir plusieurs (max 20) suivant la valeur du paramètre DB_WRITER_PROCESSES

LGWR Log Writer
Ecrit les mises à jour, du cache de LOG de la SGA vers les ou le fichier REDOLOG, suivant qu'on utilise des LOGs multiplexés ou non.

CKPT Checkpoint
Signale l'occurrence d'un point de reprise (flush de tous les buffers de données), à DBWR

PMON Process Monitor
Libère les ressources et nettoie le cache en cas d'échec d'un process utilisateur

SMON System Monitor
Surveille les process de l'instance et assure les restaurations d'instance

RECO Recoverer
Gère les transactions distribuées (commit à 2 phases)

et optionnellement :

ARCn Archiver
sauvegarde le REDOLOG qui vient d'être terminé

Dnnn Dispatcher
Distribue les accès utilisateurs, vers les serveurs partagés, en architecture multiplexée (Shared Server)

Snnn Shared Server
Process utilisateurs partagé, en architecture multiplexée

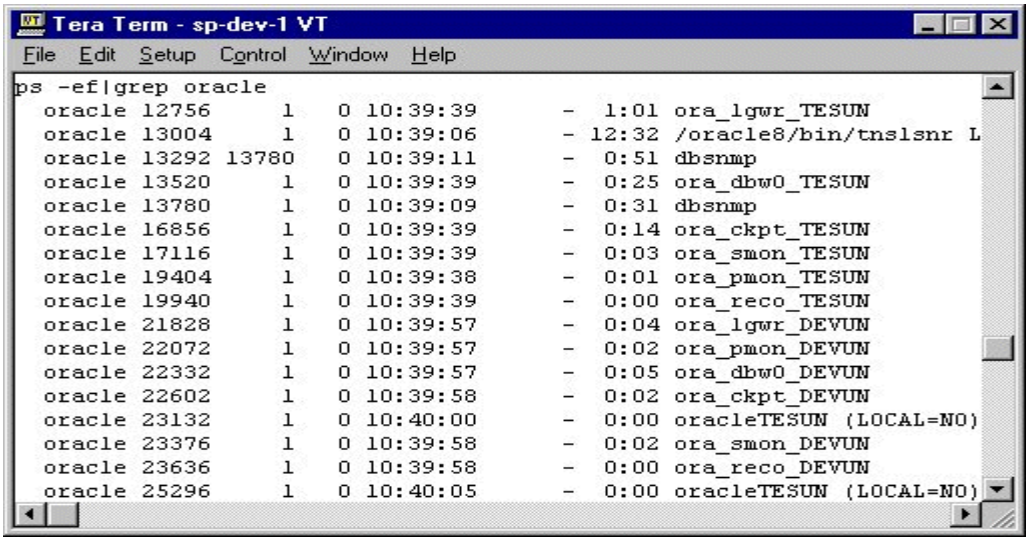
CJQn Coordinateur de jobs batch
Jnnn Process fils dédiés aux Jobs

QMNn - Queue monitor
gestionnaire de file d'attente, pour l'option Oracle Advanced QUeuing

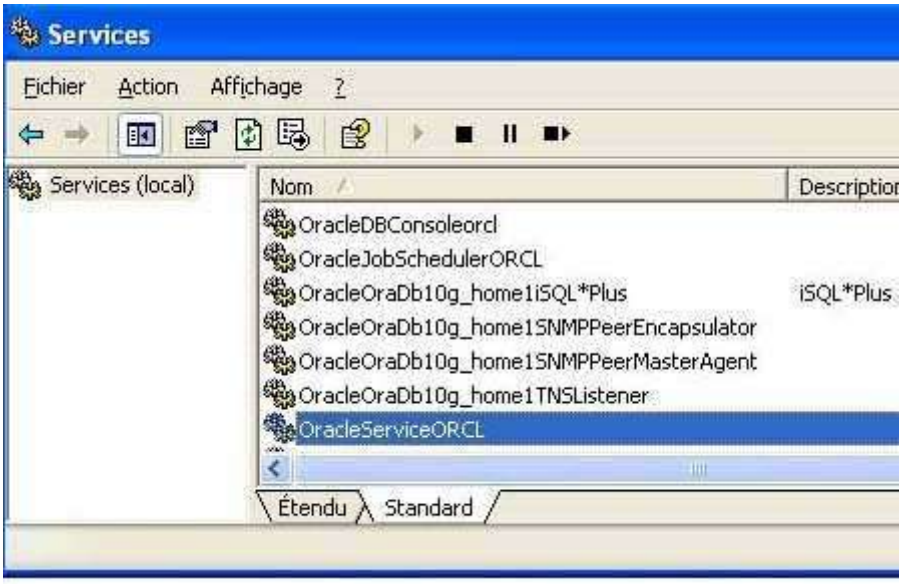
Pnnn Esclave d'exécution Parallèle
Exécution des requêtes parallèles. Le nb max de process est donné par :
PARALLEL_MAX_SERVERS

LCKn Lock monitor
verrouillage des ressources utilisées par plusieurs instances
LMS Global cache service
Gestion des ressources inter-instances en architecture cluster (RAC)

Note : sous Unix/Linux ces process correspondent à des process Unix, qui appartiennent à l'utilisateur Oracle et s'exécutent de manière 'déconnectée' (background).
On peut facilement en avoir la liste avec la commande suivante :
$> ps -ef |grep oracle



Sous Windows ils correspondent à des services



Zones mémoire

La SGA

(System Global Area) est une zone mémoire partagée par tous les utilisateurs de la base, et utilisée comme cache par le code Oracle pour stocker des infos partagées.
Elle est allouée au démarrage de l'instance et libérée à l'arrêt de la base.

Sa taille et ses modalités d'utilisation, sont pilotées par des paramètres d'initialisation de la base.

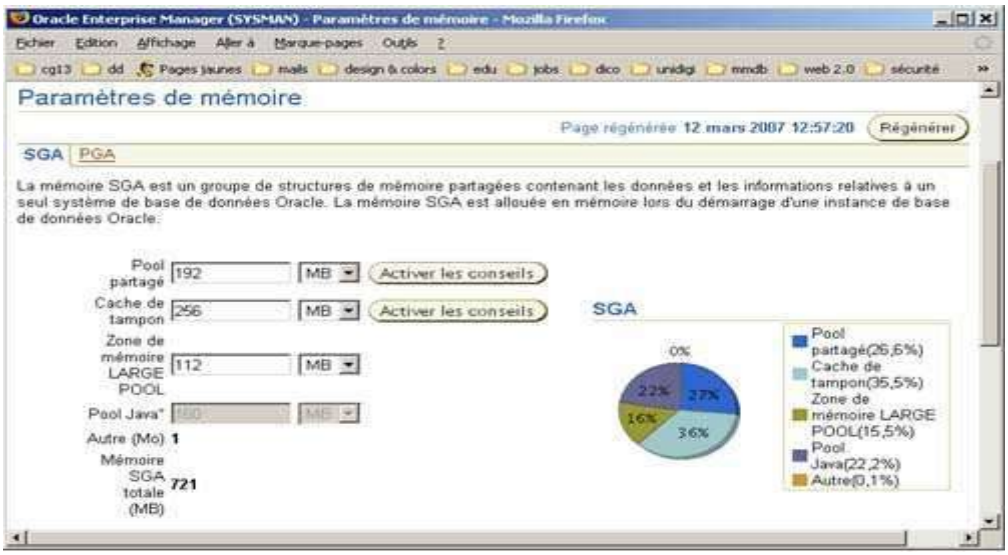
Elle contient :

- des données (lues ou écrites dans le fichier de données)
- des données modifiées à journaliser
- des instructions SQL
- des meta-données (contenu du dictionnaire)
- des programmes stockés (PLSQL ou java)

Les fichiers de données, mais aussi les redo logs ont par exemple une zone d'antémémoire spécifique en SGA : le (DATA) BUFFER CACHE et le LOG BUFFER CACHE.
Des informations résumées sur la SGA courante sont disponibles dans la vue V$SGA.

Les composant de la SGA

Des informations sur la taille et l'utilisation des différents composants de la SGA sont disponibles dans la vue V$SGAINFO du dictionnaire.



Modes de gestion de taille de la SGA

Il existe deux modes de gestion de taille de la SGA, la gestion manuelle et la gestion automatique.

* dans le mode manuel,

c'est le DBA qui analyse le fonctionnement de la base et détermine quelle valeur affecter à chacun des paramètres de taille des composants.

* dans le mode automatique,

c'est Oracle qui calcule et ajuste automatiquement, en fonction de règles et de résultats statistiques, les tailles des différents composant de la SGA et conséquemment sa taille globale.
Pour passer en mode automatique le paramètre SGA_TARGET doit être positionné à une valeur différente de 0, et les autres paramètres ajustables automatiquement mis à 0 ou à une valeur minimale.
Il est possible la taille cible de la SGA, grâce à la vue V$SGA_TARGET_SIZE
ou en calculant la taille réellement utilisé à l'instant T :
SQL SELECT SUM(value) FROM v$SGA
- SELECT current_size FROM V$SGA_DYNAMIC_FREE_MEMORY

Paramétrage des principaux composants de la SGA

Le tableau suivant montre les différentes zones de la SGA et les paramètres d'initialisation associés.

COMPOSANT SGA PARAMETRE AJUSTABLE BUT
Large pool LARGE_POOL_SIZE O
Shared pool SHARED_POOL_SIZE O
Java pool JAVA_POOL_SIZE O
Cache de données DB_CACHE-SIZE (1) O
Cache de redolog DB_LOG_BUFFER N
Cache personalisé DB_(n)K_CACHE_SIZE N

CHAPITRE A -Oracle Net

Oracle Net

Pour que la communication puisse s'établir entre un programme client et un serveur de données Oracle, il faut effectuer une configuration minimale.
Les fichiers de configuration coté client et/ou serveur se trouvent dans le répertoire ORACLE_HOME/network/admin
Ils peuvent être en dernier ressort édités à la main (fichiers "texte") mais on utilisera préférablement les Outils 'Enterprise Manager' ou 'Net Manager'

Le programme client doit notamment pouvoir identifier le serveur de données Oracle sur le réseau.
Il utilise pour ce faire une résolution de nom de serveur.

Les méthodes de résolution de nom

Avec Oracle 10g, il existe 4 modes de nommage :
* local :
on utilise un fichier de configuration situé sur le poste client (et donc sur chaque poste client!), baptisé TNSNAMES.ORA
* via un annuaire :
on utilise un serveur d'annuaire compatible LDAP : Oracle Internet Directory par exemple…
* 'facile' (easy ou EZ connect) :
basé uniquement sur TCP/IP (exceptionnellement pas de couche spécifique Oracle Net nécessaire coté client…c'est pour cela que c'est 'facile')
On utilise dès lors une chaîne de connexion explicite, qui devra préciser le serveur (machine) cible, et éventuellement le port d'écoute du logiciel serveur Oracle Net, et un nom de service de données.
Exemple : CONNECT X/Y@serveur_Z:1521/ma_base
note : le port 1521 est le port par défaut du listener Oracle Net.
* externe :
on utilise un service de nom externe, non Oracle mais compatible avec ce dernier (NIS, DCE)

Utilitaires de configuration

Graphiques : Oracle Enterprise Manager et Oracle Net Manager (client lourd)

Pour accéder à l'utilitaire de configuration, à partir de la console EM, cliquer sur 'Processus d'écoute' ou 'listener' sur la page d'accueil, puis cliquer sur le lien 'Administration des services réseaux'.

Vous pouvez alors :

- configurer le LISTENER
- configurer une résolution de nom locale (tnsnames.ora)
- configurer une résolution de nom centralisé par un annuaire (OID/LDAP)
- gérer les emplacements de fichier de config du serveur

lignes de commande Oracle net Listener
Si le PATH et ORACLE_HOME sont correctement positionnés sur Unix / Linux, il suffit de taper pour…
démarrage : $> lsnrctl start
arrêt : $> lsnrctl stop
statut : $> lsnrctl status

Remarque : sur Windows il suffit d'utiliser l'outil de gestion des services, pour le service 'oracle listener' …

Fichier de configuration client : TNSNAMES.ORA

```
# tnsnames.ora Network Configuration File:
# C:oracleproduct10.1.0Db_1networkadmintnsnames.ora
# Generated by Oracle configuration tools.
# lors de la création de la base par défaut 'ORCL'
ORCL = (DESCRIPTION =
(ADDRESS = (PROTOCOL = TCP)(HOST = DAREDEVIL)(PORT = 1521))
(CONNECT_DATA = (SERVER = DEDICATED)
(SERVICE_NAME = orcl)
)
)
```

Fichier de configuration serveur : LISTENER.ORA

```
# listener.ora Network Configuration
# File: C:oracleproduct10.1.0Db_1NETWORKADMINlistener.ora
# avec ajout d'un Service NGORCL
SID_LIST_LISTENER =
(SID_LIST =
(SID_DESC =
(GLOBAL_DBNAME = NGORCL)
(ORACLE_HOME = C:oracleproduct10.1.0Db_1)
(SID_NAME = ORCL)
)
) LISTENER =
(DESCRIPTION_LIST =
(DESCRIPTION =
(ADDRESS = (PROTOCOL = TCP)(HOST = DAREDEVIL)(PORT = 1521))
)
)
```

CHAPITRE B -Objets d'un schéma

Données et Indexs

Les Tables

Les tables sont des segments physiques. Elles contiennent les données utilisateur. Elles sont composées de colonnes typées. Elles présentent des contraintes d'intégrite (unique, clé primaire, clé étrangère, 'check').
Décrites dans le dictionnaire dans USER_TABLES, ALL_TABLES et DBA_TABLES.

exemple

```
SQL> CREATE TABLE salaires
( no_emp NUMBER , salaire NUMBER)
TABLESPACE USERS
STORAGE (INITIAL 50K NEXT 50K MINEXTENTS 1 MAXEXTENTS 25 PCTINCREASE 0)

SQL> describe salaires
Name Null? Type
——————————- ——- —-
NO_EMP NUMBER
SALAIRE NUMBER
```

on peut vérifier les caractéristiques de la table dans le dictionnaire :

```
SQL> select TABLE_NAME, TABLESPACE_NAME, INITIAL_EXTENT,
NEXT_EXTENT, BLOCKS from user_tables
where TABLE_NAME='SALAIRES'

TABLE_NAME TABLESPACE_NAME INITIAL_EXTENT    NEXT_EXTENT    BLOCKS
---------- --------------- -------------- ----------- ------
SALAIRES   USERS            53248          53248
```

Il peut être judicieux comme ici de préciser la clause STORAGE au niveau de chaque table plutôt qu'au niveau du tablespace, les volumes pouvant être très différents d'une table à l'autre….

```
SQL> CREATE TABLE mini_emp (empno NUMBER CONSTRAINT pk_emp PRIMARY KEY,
ename VARCHAR2(10) CONSTRAINT upper_ename CHECK (ename = UPPER(ename)),
hiredate DATE DEFAULT SYSDATE,
deptno NUMBER(2) CONSTRAINT fk_deptno REFERENCES scott.dept(deptno) )
```

on voit ici un certain nombre de contraintes d'intégrité qui faciliteront les développements

Les clusters

Les clusters sont des segments spéciaux qui contiennent plusieurs tables fusionnées, suivant en général une colonne de jointure. On peut les voir comme des jointures physiques.

rem : un certain nombre de tables du dictionnaire sont organisées en cluster.
Voir dba_clusters

Les indexs

Ce sont des accélérateurs, externes aux tables. Peuvent être créés / détruits à tout moment. Se remplissent dynamiquement au fur et à mesure des mises à jour de la table indexée. Organisés en B-TREE. Description dans dba_indexes.

On peut créer autant d'index qu'on veut sur une table (mais il ne faut pas en abuser…)

Remarque : un index n'est efficace que si la colonne indexée est utilisée dans la clause WHERE d'un SELECT !!

SQL> create [unique] index i1 on t1(col1)

Remarque : un index peut ne pas être unique. Une clé primaire dans une table est représentée sous forme d'index unique par Oracle.

Toutes les informations concernant les tables (description, paramètres de stockage, contraintes, index, …) peuvent être visualisées dans la console : Onglet 'Administration', rubrique 'Schema', Table / Visualiser Table

Vues, synonymes, liens, et séquences

Aucun de ces objet n'est un 'segment'. C'est à dire qu'ils ne consomment pas de blocs dans les fichiers de la base. Ils sont simplement définis dans le dictionnaire de données.

Les vues

Une vue est une fenêtre sur une table. Elle ne contient pas de données. Stockée dans le DD sous forme de 'select nommé'.
La mise à jour d'une vue est en fait la mise à jour de la table 'A TRAVERS' la vue. Il n'y a pas de duplication de données.
Une vue peut porter sur plusieurs tables, éventuellement distantes !

Les vues sont décrites dans la vue dba_views du dictionnaire de données.

create view emp_10 as select empno numero, ename nom
where deptno = 10

L'une des colonnes de dba_views est de type LONG. Si on veut visualiser la totalité des informations sous SQL*Plus, utiliser la commande 'SET LONG 2000' par exemple pour ne pas tronquer l'affichage !

Il existe des vues paramétrées qui peuvent rendre de grands services pour restreindre certains type d'accès

Si on veut restreindre l'accès à certaines périodes horaires par exemple :
CREATE VIEW emp_ouvrable
AS SELECT * FROM EMP
WHERE TO_CHAR(SYSDATE,'HH') BETWEEN '08' AND '17'

en dehors de la période 8H - 17H le predicat est faux et la vue ne renvoie aucune données !

Si on veut restreindre l'accès a certains postes de travail :
CREATE VIEW EMP_RESTREINTE
AS SELECT * FROM EMP
WHERE USERENV('TERMINAL') IN ('PC1', 'PC3')

Seuls les postes clients dont l'identification réseau est 'PC1' ou 'PC3' seront autorisés !

Les Synonymes

Un synonyme est un nom logique (un alias) référençant un objet d'un schéma.

Un synonyme peut être privé (visible seulement dans le schéma du propriétaire) ou public (visibles par tous).

Syntaxe :
SQL> create synonym nom_syn for [propr.]objet
SQL> create public synonym nom_syn for [propr.]objet

Remarque: il faut posséder le privilège 'CREATE PUBLIC SYNONYM' ou être DBA pour créer ou détruire un synonyme public.

Un synonyme public doit être unique dans la base.

Le fait que le synonyme soit public ne veut pas dire que l'objet pointé est accessible par tout le monde, il faut en plus donner des droits si nécessaire!

exemple :

SQL> CONNECT SCOTT/TIGER
SQL> GRANT SELECT ON EMP TO DUPONT
SQL> CONNECT DUPONT/JEAN
SQL> CREATE SYNONYM EMP FOR SCOTT.EMP
SQL> SELECT * FROM EMP…

Les liens de base de données (database links)

Les database links sont des références à des comptes utilisateurs distants.
Ceci permet au sein d'une même session SQL d'accéder à différents objets de bases réparties sur le réseau. On peut par exemple définir un database link compta_bordeaux qui référence le schéma compta qui se trouve sur la base distante localisée sur le serveur de bordeaux. Ensuite on pourra accéder à une table ou vue distante via ce database link. Une description des database links se trouve dans la vue USER_DB_LINKS du dictionnaire.

exemple
SQL> CREATE DATABASE LINK compta_bordeaux CONNECT TO compta IDENTIFIED BY xyz USING 'la_base_de_bordeaux' ;
SQL> SELECT * FROM balance@compta_bordeaux ;
– on peut le rendre transparent grace aux synonymes
SQL> CREATE SYNONYM balance FOR balance@compta_bordeaux ;
SQL> SELECT * FROM balance ;

Les séquences

Une séquence est un compteur programmable stocké en mémoire par Oracle et utilisable de manière partagé.
Il est en général utilisé pour fournir des no de clé d'enregistrements.
Pour créer une séquence :

SQL> create sequence nom_seq start with no_debut
increment by saut
maxvalue valeur_max cycle|nocycle cache|nocache

Pour utiliser une séquence :
la valeur courante de la séquence est donnée par nom_seq.currval et la suivante par nom_seq.nextval.

Exemple

SQL> create sequence seq_no_cli
start with 1000 increment by 100;
SQL> insert into clients (no, nom) values (seq_no_cli.nextval, 'Martin');

Pour vérifier rapidement la valeur à suivre : select seq_no_cli.nexval from dual

Programmes stockés

Les programmes stockés (procédures, fonctions ou packages) sont des programmes codés en PL/SQL, nommés, et stockés dans un schéma de la base.
Nécessite le privilège 'Create procedure' ou 'create any procedure' suivant qu'on est dans son propre schéma ou dans un autre schéma.

Il existe des outils comme 'TOAD' (de chez Quest Software) ou 'SQL*developper' (Oracle) qui simplifient la vie du développeur et du DBA pour la gestion de ces objets.

Ces programmes stockés doivent être valides. Dans certains cas notamment quant les objets référencés (tables, vues,etc) dans les programmes ont été modifiés, les procédures peuvent devenir invalides.
Après résolution du problème, les programmes doivent être recompilés :

SQL> ALTER PROCEDURE | PACKAGE | FUNCTION COMPILE

Procédures

Exemple de procédure de crédit d'un compte (qui prend en entrée le no du compte et le montant à créditer)

SQL> create or replace procedure bidon
is
begin
null;
end;

SQL> CREATE PROCEDURE credit (no IN NUMBER, montant IN NUMBER)
AS
BEGIN
UPDATE compte SET solde = solde + montant
WHERE no_compte= no;
END;

Pour exécuter une procédure :

SQL> execute nom_procedure (ses_parametres);
ou
SQL> BEGIN
nom_proc (ses_parametres)
END;
/

Attention ! pour voir le résultat d'une procédure à l'écran, il fuat que la sortie écran soit valide, avant de lancer son exécution (sinon on ne voit rien!!!) :

SQL> SET SERVEROUTPUT ON

Fonctions

Une fonction est une procédure qui retourne une (ou des) valeur(s).

SQL> CREATE FUNCTION lit_solde (no IN NUMBER)
RETURN NUMBER
IS solde NUMBER (11,2);
BEGIN
SELECT balance INTO solde FROM compte
WHERE no_compte = no;
RETURN (solde);
END;

no est un paramètre d'entrée et solde la valeur du solde du compte fournie en sortie

Packages

Un package est un ensemble encapsulé de procédures et ou de fonctions, ainsi que de données dépendant fonctionnellement.
Il est constitué de 2 parties : la spécification qui déclare les objets publics du package et le corps (body) qui définit ces objets.
Pour créer ces 2 entités on utilisera respectivement les ordres :
create package et create package body

Triggers

Un trigger est un morceau de code PL/SQL, stocké dans la base, déclenché lors de l'occurrence d'un événement particulier. Il permet notamment de synchroniser des opérations entre plusieurs tables.
La plupart du temps les triggers sont déclenchés par la modification du contenu d'une table.
La liste des événements déclencheurs apparaît ci-après :

Type d'ordre ordre déclencheur : DELETE, INSERT, UPDATE, CREATE, ALTER, DROP, SERVERERROR, LOGON, LOGOFF, STARTUP, SHUTDOWN

exemple
– trigger déclenché lors d'une insertion
– ou d'une modification de la table client
SQL> CREATE OR REPLACE TRIGGER aff_discount BEFORE INSERT OR UPDATE ON clients

```
FOR EACH ROW WHEN (new.no_cli > 0)
DECLARE evol_discount number;
BEGIN evol_discount := :new.discount - :old.discount;
DBMS_OUTPUT.PUT_LINE(' evolution : ' || evol_discount); END;
/ — FOR EACH ROW signale qu'une modification de 4 lignes
– par un seul UPDATE déclenche 4 fois le trigger.
– Si on ne souhaite qu'un seul déclenchement,
– on omet simplement la clause FOR EACH ROW.
```

CHAPITRE C -Meta données

Les meta données

Définition et accès au dictionnaire

D'une manière générale il est utile de connaître un minimum du contexte des données que l'on souhaite consulter ou mettre à jour : la liste de mes tables, les noms et types des colonnes d'une de mes tables, les tables accessibles, etc. Ces données qui concernent les données, sont appelées méta-données (ou META DATA). Suivant les bases de données, on peut y avoir accès soit via une interface graphique, soit via des commandes spécifiques simples, ou + simplement encore via des SELECT si les méta données sont stockées dans des tables relationnelles.

Pour se faire une idée on a sur http://zqlzoo.net un aperçu de l'accès aux méta données des principaux SGBDs du marché.

Le dictionnaire de données (DD) est le référentiel unique qui décrit tous les objets physiques et logiques d'une base de données Oracle. Il est accédé implicitement par le noyau Oracle et explicitement par les utilisateurs DBA ou non. Par exemple lorsque l'on crée une table le noyau insère automatiquement une ligne descriptive dans la table des tables. Le dictionnaire est organisé en tables et en vues, ce qui implique que son interrogation peut se faire en SQL !
Remarque : il est interdit à un utilisateur même DBA d'écrire explicitement dans le dictionnaire.

Contenu du dictionnaire

Le dictionnaire est composé de tables appartenant à SYS, et dont le nom se termine en général par un '$ (SYS.TAB$ la table des tables, SYS.TS$ la table des tablespaces , etc.) . Pour des raisons de confort et de sécurité, on n'accède pratiquement jamais à ces tables en direct, on passe par l'intermédiaire de vues plus lisibles. Les vues les plus couramment utilisées sont des vues statiques. Elles décrivent les : tablespaces, fichiers physiques, tables, contraintes, clusters, vues, index, synonymes, procédures, fonctions, packages, triggers, utilisateurs, droits d'accès, rôles, profils, audits, etc… Il existe également des vues dynamiques, concernant essentiellement les ressources système en cours d'utilisation, les ressources Oracle en cours d'utilisation, les sessions connectées, les verrous, etc. C'est ce qu'on appelle les vues et tables virtuelles. Toutes les tables et vues du dictionnaire appartiennent à SYS. Elles sont consultables suivant le profil de l'utilisateur.

Les vues statiques

Il existe 3 catégories de vues (reconnaissables par leur préfixe) :

USER_XXX : décrit les objets appartenant à l'utilisateur connecté

ALL_XXX : décrit les objets accessibles à l'utilisateur connecté

DBA_XXX : décrit tous les objets (vues autorisées aux DBAs seulement…)

Chaque XXX est en général remplacé par un nom (en anglais) significatif. Ainsi USER_TABLES est la vue de toutes MES tables, DBA_SYNONYMS est la vue de TOUS les synonymes du système. Une (méta) description du dictionnaire est donnée par la vue DICT.

CHAPITRE D -Gestion du stockage

Tablespaces et fichiers

De l'utilité des tablespaces

Un tablespace ou espace disque logique, est une partition logique de la base contenant un ou plusieurs fichiers.
Un fichier appartient à 1 et 1 seul tablespace.
Par défaut un tablespace à la création est ON LINE et donc accessible, il peut être mis OFFLINE (et les fichiers qu'il contient par conséquent) pour en interdire l'accès ou pour certaines opérations de maintenance

Il existe toujours deux tablespace baptisés SYSTEM et SYSAUX .
- SYSTEM : contient le dictionnaire de données et segment d'annulation SYSTEM
- SYSAUX : contient les informations nécessaires aux composants et outils supplémentaires et traditionnellement on créera également
- 'TEMP' : pour les données 'swappées' sur disque lors d'opération de tri ou de fusion trop volumineuses en mémoire
- 'UNDO' : pour les segments d'annulation, qui stockent les images avant, lors des ROLLBACKS

Outre ces tablespaces 'système' qui servent en quelque sorte à la cuisine interne d'Oracle, il faudra bien tout de même stocker quelques données (et indexs)
Ici plusieurs stratégies sont possibles :
- séparation des indexs et des datas,
- séparation des différents domaines fonctionnels

Note : Il serait possible également de stocker les datas, les index dans ces SYSTEM ou SYSAUX.
Ceci est vivement déconseillé, car on aurait ainsi une base minimale peu structurée.

Les tablespaces sont donc utiles pour répartir les données, les index, mais aussi les segments d'annulations et les espaces temporaires sur plusieurs espaces logiques et disques.
Ils permettent :
- performance (répartitions des accès disques),

- souplesse (séparation fonctionnelle ou métier, meilleure granularité des sauvegardes),
- sécurité (séparation des infos systèmes des données utilisateurs)

SQL de base pour la gestion des TBS

SQL> CREATE TABLESPACE …
SQL> DROP TABLESPACE…
SQL> ALTER TABLESPACE…

exemples
SQL> CREATE TABLESPACE COMPTA DATAFILE
'E:orantdatabaseTESTcompta1TEST.ora' SIZE 100M;
SQL> ALTER TABLESPACE COMPTA OFFLINE;
SQL> ALTER TABLESPACE COMPTA ADD DATAFILE
'E:orantdatabaseTESTcompta2TEST.ora' SIZE 100M;
SQL> DROP TABLESPACE COMPTA INCLUDING CONTENTS AND DATAFILE;
SQL> ALTER TABLESPACE CHARGEMENT_BATCH NOLOGGING;
SQL> ALTER TABLESPACE INFOCENTRE READ ONLY;

Description des tablespaces et fichiers de la base courante dans les vues DBA_TABLESPACES, DBA_DATA_FILES, DBA_FREE_SPACE du dictionnaire.

SQL> SELECT TABLESPACE_NAME "Nom TBS", CONTENTS "Type de contenu", STATUS "EN ligne?", LOGGING "Journalise?", BIGFILE FROM DBA_TABLESPACES;

Nom TBS Type de contenu EN ligne? Journalise?
——— ———— ——— ———-
BIGFILE SYSTEM PERMANENT ONLINE LOGGING NO
UNDOTBS1 UNDO ONLINE LOGGING NO
SYSAUX PERMANENT ONLINE LOGGING NO
TEMP TEMPORARY ONLINE NOLOGGING NO
USERS PERMANENT ONLINE LOGGING NO
EXAMPLE PERMANENT ONLINE LOGGING NO
6 rows selected.

Tablespaces et fichiers

Un tablespace contient AU MOINS un fichier. Celui-ci est créé lors de la création du tablespace, de manière automatique par
Oracle, en fonction des paramètres donnés par la commande CREATE ou ALTER tablespace (emplacement du fichier, nom, taille, et mode d'extension).

Note : Lors de la suppression du tablespace (DROP TABLESPACE…) les fichiers correspondant ne sont PAS SUPPRIMES par Oracle par défaut. Utilisez la clause 'AND DATAFILE'…

exemples
SQL>
Nom Tbs Nom Fic. Taille en MO AUTOEXTENSILE
USERS C:ORACLEPRODUCT10.1.0ORADATAORCLUSERS01.DBF 5 YES
SYSAUX C:ORACLEPRODUCT10.1.0ORADATAORCLSYSAUX01.DBF 230 YES
UNDOTBS1 C:ORACLEPRODUCT10.1.0ORADATAORCLUNDOTBS01.DBF 30 YES

SYSTEM C:ORACLEPRODUCT10.1.0ORADATAORCLSYSTEM01.DBF 440 YES
EXAMPLE C:ORACLEPRODUCT10.1.0ORADATAORCLEXAMPLE01.DBF 150 NO
EXAMPLE C:ORACLEPRODUCT10.1.0ORADATAORCLEXAMPLE012.DBF 10 NO

Extension et gestion d'espace des tablespaces et des fichiers

La taille d'un tablespace est la taille de son (ses) fichier(s) d'origine.
Pour augmenter la taille d'un tablespace, il y a 2 solutions :
* Ajouter un fichier au tablespace, qui sera chainé au premier (ALTER TABLESPACE toto ADD DATAFILE…)
* mettre le fichier du tablespace en AUTO extension (ALTER DATABASE DATAFILE toto.dbf AUTOEXTEND ON)
Une table (et tout segment en général) , peut "s'étaler" sur plusieurs fichiers. Ainsi le fait qu'une table sature un tablespace n'est pas bloquant il suffit d'augmenter la taille du tablespace.

autoextension des fichiers

ATTENTION : la clause AUTOEXTEND spécifie la taille d'extension du fichier d'un tablespace.
La clause STORAGE INITIAL, NEXT, MINEXTENTS … spécifie la taille d'extension d'UN SEGMENT du tablespace par exemple une table. Ces 2 paramètres sont totalement indépendants.
La preuve en est qu'une table (un segment de données) est forcément en allocation dynamique alors qu'un fichier peut avoir une taille fixe (AUTOEXTEND OFF)

Note : Le changement de mode AUTOEXTEND se fait avec la commande 'ALTER DATABASE' pour les 'SMALLFILE' et 'ALTER TABLESPACE' pour les 'BIGFILE'

exemples
SQL> — passage en AUTO extension d'un fichier de tablespace existant
SQL> ALTER DATABASE DATAFILE 'E:orantdatabaseTESTUsr1TEST.ora' AUTOEXTEND ON;
SQL> ALTER DATABASE DATAFILE
'C:ORACLEPRODUCT10.1.0ORADATAORCLEXAMPLE01.DBF'
AUTOEXTEND OFF
SQL> — ajout d'un ficheir auto extensible jusqu'a 100 MO
SQL> ALTER TABLESPACE toto ADD DATAFILE 'E:orantdatabaseTESTTEST.ora' SIZE 10M
AUTOEXTEND ON NEXT 5M MAXSIZE 100M;

extension des segments

* clause LOCAL : Tablespaces gérés localement (Locally managed tablespaces)
Anciennement les tablespaces étaient gérés au niveau du dictionnaire de données, la gestion de l'espace physique (allocation / libération de blocs) se fait désormais dans l'entête du fichier(s) du tablespace. Une table binaire d'allocation (bitmap) y
est maintenue. C'est le fonctionnement par défaut (sauf pour le tablespace SYSTEM)
Avantages :
* pas de contention en mise à jour au niveau du dictionnaire
* et conséquemment pas d'utilisation de Rollback segment pour ces transactions
* pas de soucis de gestion de l'espace (calcul d'un storage adéquat)
* "coalesce" automatique (fusion des espaces libres contigus pour optimiser l'espace libre)
Evidemment la clause "DEFAULT STORAGE" est invalide pour les tablespaces gérés localement.

* Clause AUTOALLOCATE
C'est Oracle qui gère !

* Clause UNIFORM
Les extents ont tous la même taille, par défaut 1MO, sinon elle est précisée par le paramètre 'SIZE'

* clause STORAGE
Les règles et les statistiques d'allocations sont gérées au niveau du dictionnaire.
Pour plus d'informations voir le chapitre sur les 'segments et extents'
changement des paramètres d'un tablespace existant
ALTER TABLESPACE SYSTEM
DEFAULT STORAGE (INITIAL 100K NEXT 100K MINEXTENTS 1 MAXEXTENTS 300
PCTINCREASE 1);

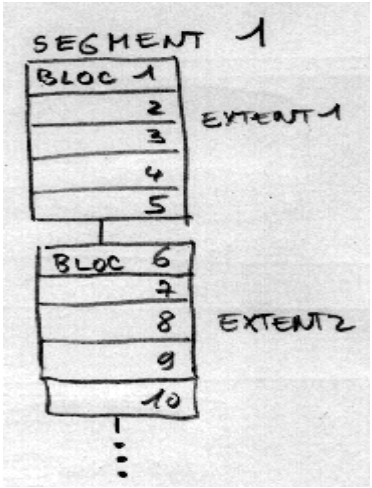
Quelques exemples de syntaxe

```
SQL> CREATE TABLESPACE COMPTA DATAFILE
'E:orantdatabaseTESTcompta1TEST.ora' SIZE 100M
EXTENT MANAGEMENT LOCAL AUTOALLOCATE;
SQL> CREATE TABLESPACE COMPTA DATAFILE
'E:orantdatabaseTESTcompta1TEST.ora' SIZE 100M
EXTENT MANAGEMENT LOCAL UNIFORM SIZE 500K;
SQL> CREATE TABLESPACE COMPTA DATAFILE
'E:orantdatabaseTESTcompta1TEST.ora' SIZE 100M
EXTENT MANAGEMENT DICTIONARY;
```

Segments, extents et blocs

Segments

Un segment est composé d'extents. Un extent est composé de blocs contigus dont la taille dépend de l'OS. Le segment s'étend dynamiquement au sein du tablespace (éventuellement sur plusieurs fichiers donc…).



Il existe plusieurs types de segments :
- segment de données = table

- segment d'index = index
- segment d'annulation = rollback segment, qui stocke l'image avant modification des données
- segment temporaire, utilisé en interne par Oracle, si la zone mémoire de tri est insuffisante

Remarque : seuls les objets 'physiques' peuvent être des segments. Ainsi une vue ou un synonyme n'est pas un segment…

On peut forcer les segments de données et d'index à s'implanter dans un tablespace particulier :
- explicitement à la création du segment
- implicitement en affectant un tablespace par défaut à l'utilisateur qui va créer le segment.

```
SQL> create table credit (n number, …)
tablespace TBS_COMPTA;
ou bien
SQL> create user Appli_comptable default tablespace COMPTA;
SQL> create table credit (n number, …);
```

Description dans la vue dba_segments du dictionnaire.

Principe d'allocation des tablespaces

Allocation des Tablespaces standards

L'allocation obéit à des règles définies par une clause STORAGE. Oracle applique d'abord la clause STORAGE du segment (définie explicitement dans un create table par exemple), sinon il utilise la clause du tablespace (définie explicitement lors du create tablespace), sinon il utilise les valeurs par défaut (implicites) du tablespace.

Au départ, lors de la création du segment (table, index ou autre) Oracle alloue MINEXTENTS extents de taille INITIAL. Ensuite, lorsque le segment se remplit, quand le(s) premier(s) extent(s) est (sont) plein(s), il alloue un extent de taille NEXT. Ensuite il augmente la taille des extents d'un pourcentage fixé par PCTINCREASE. La limite est définie par MAXEXTENTS.

SQL> create table credit (n number,…)
storage (initial 10K next 10K pctincrease 50 maxextents 100) ;

voici ce que donne le segment après création, puis 2 allocations supplémentaires :

Voir dba_extents dans le dictionnaire.

Allocation des Tablespaces gérés localement

Il y a deux types d'allocation pour ces tablespaces :

* "SYSTEM" : le DBA précise la taille de l'extent INITIAL puis Oracle détermine au mieux la taille des extents suivants (minimum 64K)
* "UNIFORM" : tous les extents auront la même taille, précisée par le DBA, ou la taille par défaut : 1MO

La gestion de l'espace libre

Lorsqu'un tablespace est créé (et le fichier minimum associé) Il est pratiquement vide hormis l'entête du fichier. L'espace libre (FREE SPACE) diminue au fur et à mesure de la creation ou de l'augmentation des segments dans ce tablespace et se fragmente lors de la libération de blocs (DELETE, TRUNCATE, etc).
Une "carte des trous" est donnée dans DBA_FREE_SPACE. Cette table possède une ligne par trou et la colonne BYTES donne la taille de chaque trou en octets.
Pour avoir l'espace libre total d'un tablespace il faudra donc sommer tous les espaces libres de tous les fichiers du tablespace.

SELECT SUM(BYTES)/1024 "Taille en KO" FROM DBA_FREE_SPACE
WHERE TABLESPACE_NAME= 'TBS_TOTO

Pour récupérer l'espace libre d'un segment, on peut utiliser
TRUNCATE table_toto DROP STORAGE
ou
ALTER TABLE table_toto DEALLOCATE UNUSED

L'espace libre est défragmenté de temps en temps par Oracle. En clair les espaces libres contigus sont fusionnés (COALESCED).

Voir DBA_FREE_SPACE_COALESCED dans le dictionnaire.

On peut faire un COALESCE manuel :
ALTER TABLESPACE toto COALESCE

Tablespaces spéciaux

Il existe différents types de tablespaces spéciaux…

Tablespaces en lecture seule (READ ONLY tablespaces)

Ces tablespaces sont utilisés (on s'en serait douté) en lecture seule. Ils permettent de stocker des données statiques (ou variant très peu souvent, éventuellement sur des CDROMS, et ne rentrent pas en ligne de compte dans les sauvegardes / restaurations.
Pour modifier les données d'un Tablespace READ ONLY il est évidemment obligatoire de modifier préalablement son statut.

SQL> ALTER TABLESPACE toto READ ONLY;
SQL> ALTER TABLESPACE toto READ WRITE;

Tablespaces temporaires (temporary tablespaces)

Ces tablespaces apparus avec la 9i remplacent les segments temporaires placés précédemment dans des tablespaces standards.
On peut (et doit) créer un tablespace temporaire par défaut autre que SYSTEM, où seront stockées toutes les données temporaires (utilisées lors des tris, création d'index, jointures, etc).
Ils sont définis lors de la création de la base :
SQL> CREATE DATABASE ma_base…
DEFAULT TEMPORARY TABLESPACE mon_temp;

ou a posteriori :
SQL> ALTER DATABASE DEFAULT TEMPORARY TABLESPACE tempts2;

En plus de ce tablespace temporaire par défaut, chaque utilisateur peut se voir assigner un tablespace temporaire particulier
SQL> CREATE TEMPORARY TABLESPACE mon_temp TEMPFILE '/oracle/data/temp01.dbf'
SIZE 20M
EXTENT MANAGEMENT LOCAL UNIFORM SIZE 10M
SQL> CREATE USER toto
IDENTIFIED BY tutu
DEFAULT TABLESPACE data
QUOTA 100M ON data
TEMPORARY TABLESPACE temp_ts

Voir les vues dynamiques V$TEMP_EXTENT_MAP et V$TEMP_SPACE_HEADER pour des infos précises sur l'utilisation en temps réel de ces tablespaces.

Tablespace d'annulation (undo tablespaces)

Les UNDO tablespaces sont exclusivement réservés au stockage de segments d'images avant modification des données pour des annulations éventuelles (ROLLBACK).
Dans les versions précédentes d'Oracle, ces structures n'existaient pas et on utilisait des ROLLBACK SEGMENTS implantables dans n'importe quel tablespace. Oracle peut désormais fonctionner avec des UNDO tablespaces (gestion Automatique, préconisée) ou avec des rollback segments (gestion manuelle)

Paramètres de l'INIT.ORA associés
UNDO MANAGEMENT AUTO | MANUAL
utiliser les UNDO tbs ou les rollback segments
UNDO TABLESPACE nom tablespace
précise le tablespace d'UNDO à utiliser par la base
UNDO_SUPPRESS_ERRORS TRUE | FALSE
évite les erreurs lorsque l'on tente d'utiliser explicitement les Rollback Segments (ALTER ROLLBACK…, SET TRANSACTION USE ROLLBACK…) alors que la base utilise des UNDO tbs

SQL > CREATE UNDO TABLESPACE undo_1

Ou dès la création de la base
SQL> CREATE DATABASE test…
UNDO TABLESPACE undo_1
DATAFILE '/tmp/undo1.dbf' SIZE 10M AUTOEXTEND ON

Tablespaces BIGFILE

Ces tablespaces comme leur nom le laisse supposer sont réservés aux bases très volumineuses. Ils contiennent un fichier UNIQUE dont la taille peut atteindre jusqu'à 4 milliards de blocs, soit pour un bloc minimum de 2KO, une taille respectable de 8TO!
Le type de tablespace par défaut SMALLFILE ou BIGFILE peut être spécifié pour toute la base à la création ou par une commande 'ALTER DATABASE'

SQL> CREATE BIGFILE TABLESPACE gulliver
DATAFILE '/oracle/data/gulliver1.dbf' SIZE 100GO;

Note : si toutes les données sont sur un seul disque on aura tout intérêt à ce que ce dernier soit 'strippé'

CHAPITRE E -Droits d'accès

Accès aux objets

Implicitement le créateur d'un objet (TABLE, VUE, INDEX, etc.) est son propriétaire, et possède tous les droits dessus : consultation, mises à jour, mais aussi suppression complète.
GRANT et REVOKE permettent respectivement de donner ou de supprimer les droits explicites d'accès en lecture ou mise à jour à un utilisateur particulier, pour un objet particulier.
C'est en général le propriétaire de l'objet qui peut donner des droits d'accès à un autre utilisateur.

exemples:
SQL> GRANT SELECT ON TAB_CLIENTS TO MARTIN
SQL> GRANT UPDATE, INSERT ON TAB_CLIENTS TO DUPONT

Un utilisateur non propriétaire peut donner des droits sur un objet ne lui appartenant pas : le DBA bien sûr, mais aussi un utilisateur qui a reçu des droits et le droit de transférer ce droit (GRANT OPTION)

exemple :

SQL> connect DBA1/DBA1
SQL> GRANT SELECT ON scott.emp TO martin;
SQL> GRANT SELECT, UPDATE, INSERT ON scott.emp TO dupont WITH GRANT OPTION;
SQL> CONNECT dupont/dupont
SQL> GRANT UPDATE ON scott.emp TO martin;

Note : 'GRANT OPTION' est à utiliser avec parcimonie…

Le type de privilège dépend évidemment de l'objet sur lequel il s'applique. Ceci est résumé dans le tableau suivant :

TABLE	VUE	VUE MAT.	SEQUENCE		PROCEDURE	
ALTER		X			X	
DELETE		X	X	X		
EXECUTE						X
INDEX		X				
INSERT		X	X	X		
REFERENCES (1)		X	X			
SELECT		X	X	X	X	
UPDATE		X	X	X		
ON COMMIT REFRESH		X				
QUERY REWRITE		X				

(1) possibilité de créer une clé étrangère sur la table

Une erreur très répandue consiste à référencer un objet (dont on n'est pas proprétaire), sur lequel on a des droits et qu'on oublie de préfixer…

SQL> SELECT * from EMP

donne 'Table or view doesn't exists' !!!

alors que SELECT * FROM SCOTT.EMP donne un résultat !!!!

Les rôles

Il n'existe pas de notion de groupe d'utilisateur sous Oracle, mais la notion de rôle, qui permet de nommer un groupe de privilèges. On peut affecter un rôle à un ou n utilisateurs, voir à un rôle.

Rôles prédéfinis
Il existe un certain nombre de rôles prédéfinis, fournis avec Oracle 10g. Voici une liste de ces rôles avec les

privilèges système qu'ils offrent :
CONNECT :
Se connecter ! Équivalent du privilège système 'CREATE SESSION'

Attention !!! en version 10gR1 et antérieures, ce rôle donnait beaucoup + de privilèges, à savoir :
ALTER SESSION, CREATE CLUSTER, CREATE DATABASE LINK, CREATE SEQUENCE,
CREATE SESSION, CREATE SYNONYM, CREATE TABLE, CREATE VIEW
RESOURCE :
créer des données avec des quotas sur tous les tablespaces ->
CREATE CLUSTER, CREATE INDEXTYPE, CREATE OPERATOR, CREATE PROCEDURE,
CREATE SEQUENCE, CREATE TABLE, CREATE TRIGGER, CREATE TYPE

DBA :
Tous les privilèges système avec ADMIN OPTION

EXP_FULL_DATABASE :
export complet ou incrémental ->
SELECT ANY TABLE, BACKUP ANY TABLE, EXECUTE ANY PROCEDURE, EXECUTE ANY TYPE,
ADMINISTER RESOURCE MANAGER, and

INSERT, DELETE, and UPDATE on the tables SYS.INCVID, SYS.INCFIL, and SYS.INCEXP.
Also the following roles:

EXECUTE_CATALOG_ROLE and SELECT_CATALOG_ROLE.

IMP_FULL_DATABASE :
import full + tous les privilèges système + role

DELETE_CATALOG_ROLE :
droit DELETE sur la table SYS.AUD$

EXECUTE_CATALOG_ROLE : privilège EXECUTE sur les objets du dictionnaire + HS_ADMIN_ROLE.

SELECT_CATALOG_ROLE : accès en consultation au dictionnaire + HS_ADMIN_ROLE.

RECOVERY_CATALOG_OWNER : privilèges pour le propriétaire du catalogue de restauration -> CREATE SESSION, ALTER SESSION, CREATE SYNONYM, CREATE VIEW, CREATE DATABASE LINK,
CREATE TABLE, CREATE CLUSTER, CREATE SEQUENCE, CREATE TRIGGER, and CREATE PROCEDURE

HS_ADMIN_ROLE
protection des accès (SELECT et EXECUTE) au référentiel HS (Heterogeneous Services) data dictionary tables

(grants SELECT) and packages
AQ_ADMINISTRATOR_ROLE
privilèges d'administration de l' Advance Queuing. Notamment : ENQUEUE ANY QUEUE, DEQUEUE ANY QUEUE, et MANAGE

ANY QUEUE + SELECT sur les tables AQ + EXECUTE sur les packages AQ.

Rôles applicatifs

Des rôles définis pour les applicatifs, en fonction des besoins et à minima !
Ainsi on peut créer un rôle qui donne des droits de consultation sur les tables d'un schéma :

– creation du role
SQL> create role consult_finance;
– affectation des privileges au role
SQL> grant select on budget to consult_finance;
SQL> grant select on fournisseur to consult_finance;
SQL> grant select on client to consult_finance;
– cession du role aux utilisateurs
SQL> grant consult_finance to user_finance_1;
SQL> grant consult_finance to user_finance_2;

Note : un user ne peut pas hériter de plus de 'MAX_ENABLED_ROLES' paramètre d'initialisation de la base, dont la valeur par défaut est 30 en 10g. Cela parait peu probable pour un applicatif classique (voire déraisonnable).
Dans le cas d'une console d'administration centrale (console 10g GRID ou server manager console). L'user d'administration connecté hérite de tous les droits qu'il crée…ce qui est beaucoup plus fréquent ! et la limite MAX_ENABLED_ROLES peut être facilement atteinte et … empêcher une connexion à la console !
Informations sur les rôles dans le référentiel (dictionnaire) Oracle 10g
————————————————————————

DBA_ROLES
Tous les roles !
DBA_ROLE_PRIVS, USER_ROLE_PRIVS
roles donnés aux users et/ou aux roles

ROLE_ROLE_PRIVS
roles donnés aux roles
ROLE_SYS_PRIVS
privilèges systèmes donnés aux roles (accessible au user)
ROLE_TAB_PRIVS
privilèges objets donnés aux roles (accessible au user)
SESSION_PRIVS
privilèges de la session courante
SESSION_ROLES
roles de la session courante

Les rôles et la console GRID 10g
La console donne par défaut le role CONNECT, cela ne pose pas de problème de sécurité sur une base 10g…mais si on s'en sert pour administrer une base 9i (avec l'agent qui va bien) le role CONNECT offre presque tous les privilèges de création/suppression de données dans le schéma.

Problèmes classiques sur les droits : ORA-00942 , ORA-01031

————————————————-
– une vue de BLAKE sur une table de SCOTT
————————————————-
connect scott
(scott a les droits sur la table)
create view vscott as select * from emp

connect blake
select * from scott.vscott
*
ERROR at line 1:
ORA-00942: table or view does not exist

connect scott
grant select on emp to blake

…le droit sur la table sert pour accéder …a la table
connect blake
select * from scott.vscott
*
ERROR at line 1:
ORA-00942: table or view does not exist

…Le droit sur la vue suffit
connect scott
grant select on vscott to blake

connect blake
select * from scott.vscott
-> OK

————————————————————
—KING accede a une vue de BLAKE sur une table de SCOTT

——————————————————
connect SYSTEM
grant select on scott.emp to blake;
create view blake.vblake as select * from scott.emp
connect SYSTEM
SQL> grant select on blake.vblake to king;
Autorisation de privilèges (GRANT) acceptée.
connect KING
select * from blake.vblake
*
ERROR at line 1:
ORA-01031: insufficient privileges

Si on avait tenté de donner les droits en tant que BLAKE
cela aurait été plus clair :
connect blake
grant select on blake.vblake to king
*
ERROR at line 1:
ORA-01720: grant option does not exist for 'SCOTT.EMP'

LA SOLUTION
connect SCOTT (ou SYSTEM)
grant select on scott.emp to blake WITH GRANT OPTION

Privilèges 'système'

Les privilèges systèmes sont des privilèges qui à la différence des privilèges d'accès aux objets s'intéressent plutôt au contenant qu'au contenu. Ils concernent principalement des ordres de création, de modification de structures et de suppression d'objets.
Ce sont donc des privilèges d'assez haut niveau, que l'on réservera par exemple aux développeurs mais qui seront utilisés avec beaucoup de parcimonie en phase de production…

Attention notamment à l'option 'ANY'

Quelques exemples :

ALTER SNAPSHOT	Modifier tous les snapshots dans tous les schémas.
DROP ANY SNAPSHOT	Supprimer tous les snapshots dans tous les schémas.
SYNONYM	
CREATE SYNONYM	créer un synonyme dans son schéma.
CREATE SYNONYM	Créer tous les synonymes dans tous les schémas.

DROP ANY SYNONYM	Supprimer tous les synonymes dans tous les schémas.
SYSTEM	
ALTER SYSTEM	faire des ALTER SYSTEM .
TABLE	
CREATE TABLE	Créer des tables ou des indexs dans son propre schéma
CREATE ANY TABLE	Créer des tables dans tous les schémas.
ALTER ANY TABLE	Modifier toutes les tables dans tous les schémas et compiler toutes les vues dans tous les schémas.
BACKUP ANY TABLE	Réaliser des exports incrémentaux.
DROP ANY TABLE	Supprimer ou vider toutes les tables dans tous les schémas.
LOCK ANY TABLE	Verrouiller toutes les tables ou vues dans tous les schémas.
COMMENT ANY TABLE	Commenter toutes les tables, vues, ou colonnes dans son schéma.
SELECT ANY TABLE	Interroger toutes les tables, vues, ou clichés dans tous les schémas.

CHAPITRE F -Transferts de données

Data pump

Datapump est un utilitaire qui permet de transférer des données entre une base Oracle…et une base Oracle ! (il correspond aux anciens utilitaires export/import d'Oracle V9 et précédentes)

DataPump export : extrait des données
DataPump import : injecte des données (préalablement exportées)

Les 2 exécutables correspondants 'expdp' et 'impdp' se trouvent comme les autres dans le répertoire bin sous 'ORACLE_HOME' (indifféremment sous Unix ou Windows)

syntaxe :
expdp KEYWORD=value or KEYWORD=(value1,value2,…,valueN)

Exemple:
expdp scott/tiger DUMPFILE=scott.dmp DIRECTORY=dmpdir SCHEMAS=scott

Options d'export

Keyword Description (Default)
————— ————————————————-
ATTACH Attach to existing job, e.g. ATTACH [=job name].
CONTENT Specifies data to unload where the valid keywords are:
(ALL), DATA_ONLY, and METADATA_ONLY.
DIRECTORY Directory object to be used for dumpfiles and logfiles.
DUMPFILE List of destination dump files (expdat.dmp),
e.g. DUMPFILE=scott1.dmp, scott2.dmp, dmpdir:scott3.dmp.
ESTIMATE Calculate job estimates where the valid keywords are:
(BLOCKS) and STATISTICS.
ESTIMATE_ONLY Calculate job estimates without performing the export.
EXCLUDE Exclude specific object types, e.g. EXCLUDE=TABLE:EMP.
FILESIZE Specify the size of each dumpfile in units of bytes.
FLASHBACK_SCN SCN used to set session snapshot back to.
FLASHBACK_TIME Time used to get the SCN closest to the specified time.
FULL Export entire database (N).
HELP Display Help messages (N).
INCLUDE Include specific object types, e.g. INCLUDE=TABLE_DATA.
JOB_NAME Name of export job to create.
LOGFILE Log file name (export.log).
NETWORK_LINK Name of remote database link to the source system.
NOLOGFILE Do not write logfile (N).
PARALLEL Change the number of active workers for current job.
PARFILE Specify parameter file.
QUERY Predicate clause used to export a subset of a table.
SCHEMAS List of schemas to export (login schema).
STATUS Frequency (secs) job status is to be monitored where
the default (0) will show new status when available.
TABLES Identifies a list of tables to export - one schema only.
TABLESPACES Identifies a list of tablespaces to export.
TRANSPORT_FULL_CHECK Verify storage segments of all tables (N).
TRANSPORT_TABLESPACES List of tablespaces from which metadata will be unloaded.
COMPRESSION Réduction de la taille du contenu du fichier de vidage, les mots-clés valides étant : (METADATA_ONLY) et NONE.
ENCRYPTION_PASSWORD Clé de mot de passe pour la création de données de colonne cryptées.
VERSION Version of objects to export where valid keywords are:
(COMPATIBLE), LATEST, or any valid database version.

Options d'import

Keyword Description (Default)
————————————————————
ATTACH Attach to existing job, e.g. ATTACH [=job name].
CONTENT Specifies data to load where the valid keywords are:

(ALL), DATA_ONLY, and METADATA_ONLY.
DIRECTORY Directory object to be used for dump, log, and sql files.
DUMPFILE List of dumpfiles to import from (expdat.dmp),
e.g. DUMPFILE=scott1.dmp, scott2.dmp, dmpdir:scott3.dmp.
ESTIMATE Calculate job estimates where the valid keywords are:
(BLOCKS) and STATISTICS.
EXCLUDE Exclude specific object types, e.g. EXCLUDE=TABLE:EMP.
FLASHBACK_SCN SCN used to set session snapshot back to.
FLASHBACK_TIME Time used to get the SCN closest to the specified time.
FULL Import everything from source (Y).
HELP Display help messages (N).
INCLUDE Include specific object types, e.g. INCLUDE=TABLE_DATA.
JOB_NAME Name of import job to create.
LOGFILE Log file name (import.log).
NETWORK_LINK Name of remote database link to the source system.
NOLOGFILE Do not write logfile.
PARALLEL Change the number of active workers for current job.
PARFILE Specify parameter file.
QUERY Predicate clause used to import a subset of a table.
REMAP_DATAFILE Redefine datafile references in all DDL statements.
REMAP_SCHEMA Objects from one schema are loaded into another schema.
REMAP_TABLESPACE Tablespace object are remapped to another tablespace.
REUSE_DATAFILES Tablespace will be initialized if it already exists (N).
SCHEMAS List of schemas to import.
SKIP_UNUSABLE_INDEXES Skip indexes that were set to the Index Unusable state.
SQLFILE Write all the SQL DDL to a specified file.
STATUS Frequency (secs) job status is to be monitored where
the default (0) will show new status when available.
STREAMS_CONFIGURATION Enable the loading of Streams metadata
TABLE_EXISTS_ACTION Action to take if imported object already exists.
Valid keywords: (SKIP), APPEND, REPLACE and TRUNCATE.
TABLES Identifies a list of tables to import.
TABLESPACES Identifies a list of tablespaces to import.
TRANSFORM Metadata transform to apply (Y/N) to specific objects.
Valid transform keywords: SEGMENT_ATTRIBUTES and STORAGE.
ex. TRANSFORM=SEGMENT_ATTRIBUTES:N:TABLE.
TRANSPORT_DATAFILES List of datafiles to be imported by transportable mode.
TRANSPORT_FULL_CHECK Verify storage segments of all tables (N).
TRANSPORT_TABLESPACES List of tablespaces from which metadata will be loaded.
Only valid in NETWORK_LINK mode import operations.
VERSION Version of objects to export where valid keywords are:
(COMPATIBLE), LATEST, or any valid database version.
Only valid for NETWORK_LINK and SQLFILE.

Datapump et PLSQL

Il existe un package SYS.DBMS_DATAPUMP, fourni avec Oracle 10g, qui offre un certain nombre de procédure pour exécuter / automatiser les transferts de données en PLSQL.
Voici un tout petit exemple :

```
DECLARE
hand NUMBER;
BEGIN
hand := Dbms_DataPump.Open(operation => 'EXPORT',
job_mode => 'FULL',
job_name => 'FULLEXPJOB',
version => 'COMPATIBLE');
Dbms_DataPump.Add_File(handle => hand,
filename => 'expdp_plsql.log',
directory => 'DMPDIR',
filetype => 3);
Dbms_DataPump.Add_File(handle => hand,
filename => 'expdp_plsql.dmp',
directory => 'DMPDIR',
filetype => 1);
– Dbms_DataPump.Set_Parameter(handle => hand,
– name => 'ESTIMATE',
– value => 'STATISTICS');
Dbms_DataPump.Start_Job(hand);
END;
```

Chargement de données avec SQL*Loader

SQL*Loader est, comme son nom l'indique un utilitaire de chargement spécifique pour les bases Oracle. Il permet d'initialiser une base de données, ou plus précisément une ou plusieurs tables avec des données issues d'un fichier texte.
Ainsi si l'on souhaite migrer des données d'un fichier Mainframe vers Oracle, on pourra extraire les données du fichier d'origine pour produire un fichier texte et ensuite utiliser SQL*loader pour effectuer le chargement automatique de la (ou des) table(s).

Principales caractéristiques

- charge des fichiers texte externes dans Oracle
- format des fichiers d'entrée fixe ou variable (avec séparateur)
- utilisation de fonctions SQL
- génération de clés uniques
- mode "direct" optimisé
- gestion des logs, des erreurs et possibilité de reprise

Principe général

La (ou les) table(s) destination sont créées dans le schéma cible.
On précise le format des entrées et des sorties dans un fichier de paramétrage, appelé fichier de contrôle, créé avec un éditeur de texte. Un fichier log donnant les résultats du chargement est généré. En cas d'erreur, l'enregistrement rejetés sont stockés dans un fichier '.bad', pour être éventuellement retraités.

Commande minimale :
sqlldr nom_user/mot_de_passe@base control=nom_fic.ctl

la structure de la table cible doit être créée avant le chargement, SQL*Loader à la différence d'autres outils ne crée pas la table.

options de la ligne de commandes

option description Defaut
userid username et mot de passe
control nom du fichier de controle du loader
data nom du fichier de données d'entrée
log nom du fichier de trace
bad nom du fichier des enregistrements rejetés
parfile nom du fichier contenant les paramètres de la commande…
skip n nombre d'enregistrement logiques à sauter 0
load n nombre à charger all
errors n nombre max d'erreurs autorisées 50
rows n nb de lignes du tableau utilisé pour les entrées 64
bindsize taille du tableau précédent en bytes OSdep
silent n'affiche plus les messages pendant l'exécution
direct utilise l'accès direct (direct path) false
parrallel chargement en mode parallélisé false
file fichier d'allocation pour les chargement parallèles
discard fichier des enregistrements non chargés intentionnellement (saut conditionnel)
discardmax nombre maximums de ces enregistrements non chargés all

Le contenu du fichier de contrôle, par étapes

Spécification des entrées

load data infile nom_fic | *
nom du fichier d'entrée ou * si les datas sont à la suite :
Exemples
1) load data infile c:tempentrees.dat
2) load data *
…
begindata
1000;Martin;Essonne
2000;Dupont;Marne
…
mode de chargement

insert insère les datas dans une table vide
append insère les datas à la suite des données existantes
replace insère les datas en remplaçant les données existantes
truncate insère les datas après un TRUNCATE
table cible

into nom_table (TRAILING NULLCOLS) spécifie la ou les tables (si plusieurs INTO) à charger

Description générale des champs
fields spécifie les délimiteurs

exemple : field terminated by ';' optionally enclosed by ' " '
specif colonnes

col par défaut de type CHAR
col [POSITION debut:fin | *] type_col [longueur] [format] [NULLIF col = BLANKS]
exemple
nom CHAR
no INTEGER EXTERNAL

Les types numériques (INTEGER, DECIMAL, FLOAT, …) sans EXTERNAl sont des types binaires !
La plupart du temps les numériques en entrée sont des caractères !

Remarque : le type de colonne peut être un type spécial
- RECNUM : no de ligne du fichier d'entrée
- SYSDATE : la date du jour
- CONSTANT valeur : une constante
- SEQUENCE (début, incrément) : un compteur automatique
clauses spéciales

continueif This permet de fusionner plusieurs enregistrement physique en cas d'incomplétude
exemple : continueif this (1) = '*' concatene la ligne" suivante si la ligne courante a '*' en colonne 1

WHEN condition specif colonne
fichier de contrôle avec entrées au format variable

– param.ctl
LOAD DATA INFILE 'monfic' –fichier à charger (.dat)
REPLACE –ecrase la table
INTO TABLE matable
FIELDS TERMINATED BY ','
OPTIONALLY ENCLOSED BY '"' –car. de séparation
(NUMERO, NOM, PRENOM , DATE_NAISSANCE date "DDMMYY", SEXE)

fichier de contrôle avec entrées au format fixe

LOAD DATA INFILE 'monfic.fix'
REPLACE
INTO TABLE matable (NUMERO position(1-5) char,
– position de début et de fin du champ
NOM position(6-35) char, PRENOM position (36-50) char,
DATE_NAISSANCE position (*) date "DDMMYY",
–indique la position suivante
SEXE position(*) char
–longueur implicite pour char : 1)

Chargement de 2 tables à partir d'un fichier

Un fichier (3 champs) éclaté dans 2 tables (chacune 2 colonnes) une pour les employés, l'autre pour les projets avec une colonne de jointure sur le no d'employé…
fichier d'entrée :

1000-deleglise——-10
2000-martin———-20
3000-dupont———-10
…

fichier de contrôle…
into table employes
(no_emp position(1:4) integer external,
nom position (6:20) char)
into table projet
(no_emp position (1:4) integer external,
no_projet position (22:25) integer external)
…

Compte rendu d'execution

sqlldr ch/ch control=param.ctl
SQL*Loader: Release 10.2.0.1.0 - Production on Wed Mar 15 15:52:18 2005 Copyright (c) Oracle Corporation 2005. All rights reserved.
Commit point reached - logical record count 2
Fichier log :
$>more param.log
SQL*Loader: Release 10.2.0.1.0 - Production on Wed Mar 15 15:52:18 2005 Copyright (c) Oracle Corporation 2005. All rights reserved.
Control File: param.ctl Data File: monfic.fix Bad File: monfic.bad Discard File: none specified (Allow all discards)
Number to load: ALL Number to skip: 0 Errors allowed: 50 Bind array: 64 rows, maximum of 65536 bytes
Continuation: none specified Path used: Conventional
Table MATABLE, loaded when SEXE = 0X46(character 'F')
Insert option in effect for this table: REPLACE
Column Name Position Len Term Encl Datatype
NUMERO 1:5 5 CHARACTER NOM 6:35 30 CHARACTER PRENOM 36:50 15 CHARACTER DATE_NAISSANCE NEXT 6 DATE DDMMYY SEXE NEXT 1 CHARACTER Table TATABLE, loaded when 57:57 = 0X4d(character 'M') Insert option in effect for this table: REPLACE Column Name Position Len Term Encl Datatype
NUMERO 1:5 5 CHARACTER NOM 6:35 30 CHARACTER PRENOM 36:50 15 CHARACTER DATE_NAISSANCE NEXT 6 DATE DDMMYY Table MATABLE:
1 Row successfully loaded. 0 Rows not loaded due to data errors.
1 Row not loaded because all WHEN clauses were failed.
0 Rows not loaded because all fields were null

Transfert avec les tables externes (external tables)

Les 'tables externes' sont des structures Oracle qui pointent sur un fichier externe a la base et sur lesquelles on peut faire du SQL
Voici un exemple de script de chargement de emp a partir d'un fichier externe CSV
– creer un repertoire oracle de travail
– associe au repertoire ou se trouve le fichier d'entree
–

```
create or replace directory dossier_temp as '/tmp'
/

drop table table_externe_emp
/

– creer une 'table' externe attachee au fichier d'entree
–
create table table_externe_emp
(EMPNO NUMBER(4) ,
ENAME VARCHAR2(10),
JOB VARCHAR2(9),
MGR NUMBER(4),
HIREDATE DATE,
SAL NUMBER(7, 2),
COMM NUMBER(7, 2),
DEPTNO NUMBER(2)
)
ORGANIZATION EXTERNAL
( type oracle_loader
default directory dossier_temp
access parameters
( records delimited by newline
badfile   'charge_emp.bad'
logfile   'charge_emp.log'
fields terminated by ';'
)
location ('emp.csv')
)
– attention c'est important sinon ca marche moins bien !
reject limit unlimited
/

– et on insere dans la table destination si elle existe
insert into mon_emp
select * from table_externe_emp
/
commit
```

Extraction de données SQL avec Oracle

Bizarrement Oracle ne fournit pas de commande pour transférer des données vers l'extérieur mais vous propose par contre des commandes pour charger des données d'un système extérieur VERS Oracle

Le plus simple pour produire du SQL est donc d'utiliser un outil client graphique tiers .

J'en citerai principalement 2, qui permettent d'extraire des données à partir d'une table ou d'un SELECT en quelques clics de souris :

- Oracle SQL Developer téléchargeable gratuitement sur otn.oracle.com
- T.O.A.D l'outil de développement et d'administration de Quest Software

Remarque : pour ce qui des ordres DDL (CREATE TABLE, etc…) oracle fournit des moyens standards comme datapump ou le package PLSQL DBMS_METADATA, mais cela ne concerne pas le contenu des tables ;

CHAPITRE G -Sauvegarde

Sauvegarde et restauration Oracle

Présentation

Les Sauvegarde et restauration, permettent de se prémunir plus ou moins parfaitement de la perte accidentelle de données physique, fichier de données ou autre.
Principaux cas de figure : corruption de fichier, perte de fichier, perte de disque.

Elles assurent la *fiabilité des données* un domaine parmi d'autres de la sécurité des données.

Remarque : de manière détournée, les sauvegardes logiques ou physiques peuvent être utilisées pour transférer / dupliquer des données pour un tout autre but que la fiabilité des données.
Par exemple :

- maintien de base de test à jour
- copie de BD pour infocentre ou Business Intelligence
- mise en production
- etc…

Elles obéissent à une stratégie :

- Quoi sauvegarder : totalité, tablespace, uniquement les données sensibles, etc.
- Quand : fréquence pluri quotidienne, quotidienne, hebdomadaire, etc.
- Comment : à froid, à chaud, physiquement, logiquement

et répondent à des contraintes :

- disponibité des données : haute, moyenne, basse
- importance relative de certaines données
- temps de reprise
- volume maximum de perte supporté
- économie (par exemple la très haute disponibilité coute cher…)

Les grandes catégories de sauvegarde

- **Sauvegarde physique totale**

Elle se fait 'à froid', c'est à dire base fermée. Elle est simple et rapide (dépend seulement de la taille des fichiers et de la vitesse des disques). Elle n'est pas 'divisible' par définition.

On sauvegarde les fichiers physiques (au niveau OS donc) de la base de données. TOUS les fichiers physiques : les fichiers de données bien sûr mais aussi les fichiers de paramétrage (SPFile et init.ora), de contrôle, fichiers journaux (REDO Logs)…

- **Sauvegarde physique partielle**

On sauvegarde uniquement certains tablespaces (et donc tous les fichiers physiques qu'il contient). SI un tablespace correspond à une application particulière, on sauvegarde donc toutes les données de cette application. La liste des fichiers d'un tablespace est donnée dans le dictionnaire.
Peut se faire à chaud ou à froid.

- **Sauvegarde logique totale ou partielle**

On sauvegarde soit du SQL (CREATE TABLE + INSERT en général),
soit dans un format spécifique, grâce à l'utilitaire EXPORT ou aux procédure DATAPUMP.
L'intérêt de cette méthode est sa granularité. On peut sauvegarder une table, qq tables, les données d'un schéma, ou une certaine catégorie d'objets connexes (vues, index, droits…)

Remarque :

1) l'export et le datapump Oracle ne sont utilisables…que par l'IMPORT ou le datapump Oracle
2) les fichier SQL peut être utilisé par un autre SGBD (MySQL, SQLServer) à la conformité au standard SQL près…
3) un utilisateur Lamda peut faire ses propres extractions

Voir **le chapitre sur les transferts de données** pour plus d'infos

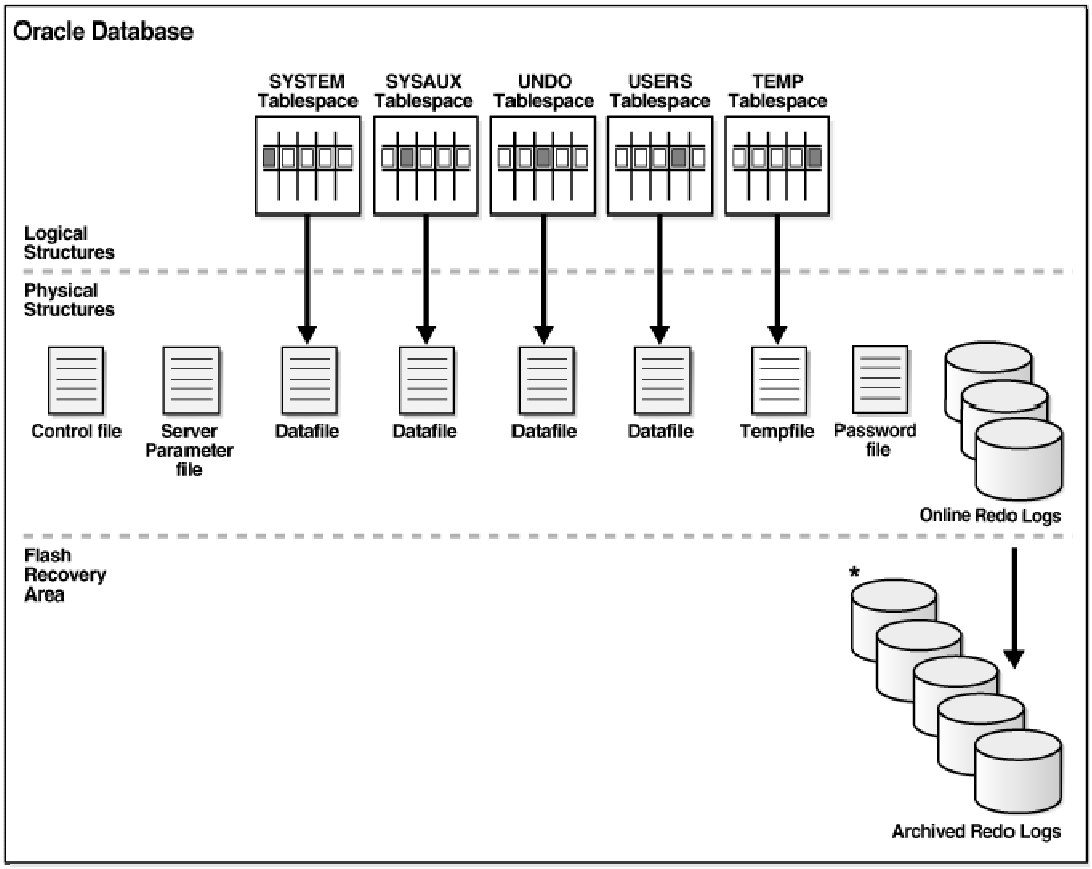
Rappel sur la structure physique d'une base Oracle

Les principaux composants physiques de la base (fichiers) sont résumés dans le schéma ci-après:

Il y a en gros au strict minimum :

- 1 control file
- des fichiers de données système (pour TBS system, sysaux, undo, temp,…)
- des fichiers de données utilisateur ou applicatifs (users, compta, RH, clients,…)
- 1 fichier de paramètres d'initialisation (spfile)
- 2 fichiers journaux (redolog)

Remarque : les fichiers archive log sont optionnels.



Infos utiles du dictionnaire sur la base et ses fichiers

v$database
description générale de la base
DBA_TABLESPACES
description des tablespaces et fichiers
DBA_DATA_FILES
description des fichiers de données
v$logfile
description des redologsz
v$log_history
info sur l'historique de tous les redos issus du control file
v$log
infos sur les groupes et les membres
v$parameter
TOUS les parametres d'init de l'instance, y compris CONTROL_FILES…
v$controlfile
nom des control files

Les cas de récupération automatique

Certains problèmes ne relèvent pas de la perte de données physiques, mais plutôt de la perte d'information en mémoire centrale : interruption d'une transaction par exemple. Ces problèmes sont automatiquement résolus par Oracle :

- échec d'un ordre SQL : provoque une erreur d'exécution documentée
- échec de process client (interruption du client, pb réseau, arrêt du shadow process) : provoque un Rollback de la transaction
- echec de l'instance (arrêt d'un process de fond du serveur, arrêt CPU serveur) : provoque un Rollback immédiat ou différé (au redémarrage de la base…)

Remarque : certains dispositifs matériels : disques miroir, CPUs redondants, machines à très haute disponibilité, etc. peuvent prendre en charge de manière automatique et transparente les pertes de données physiques, sans passer par un processus de sauvegarde / restauration.

Principes d'organisation et de répartition :

Multiplexages des fichiers de contrôle

Séparation des fichiers de données et des fichiers redologs sur des disques différents -

CREATE DATABASE test DATAFILE 'd:dataoratest_system' SIZE 10M
LOGFILE GROUP 1 ('e:/data/oratest_log1a') SIZE 500K,
GROUP 2 ('e:/data/oratest_log2a') SIZE 500K;

create tablespace TB1 datafile 'd:dataoradata1.dbf' size 100M';

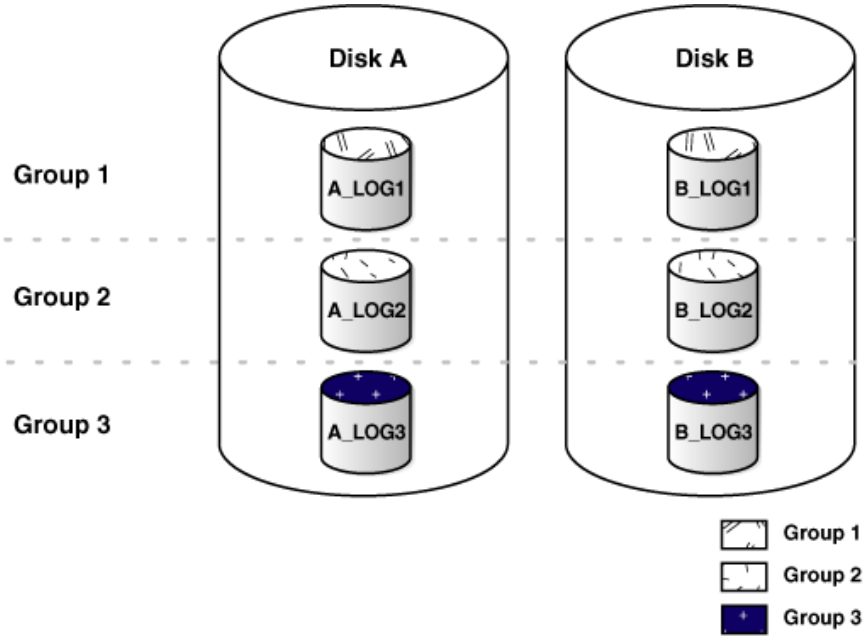
Ici on crée un nouveau Tablespace (et donc un fichier de données) sur le même disque que le fichier du tablespace 'SYSTEM' qui contient le dictionnaire. Ce n'est peut être pas une bonne idée en terme de performances…

Multiplexage des fichiers de contrôle

Pour ce faire il suffit de déclarer plusieurs fichiers de contrôle, sur des disques différents. Les mises à jour éventuelles par le noyau Oracle seront faites simultanément dans tous les fichiers. En cas de perte il suffira de recopier un des autres control file et de le renommer avec le nom du fichier perdu, puis de redémarrer la base.
la liste des fichiers de contrôle est spécifié dans le fichier de démarrage de la base : INIT.ORA
ex :
INIT_test.ora
DB_NAME = TEST
CONTROL_FILES = (/disk1/test_ctl1.ora, /disk2/test_ctl2.ora)

Multiplexage des journaux (redo log files)

On a vu que l'écriture dans les redo logs files est circulaire.
Au minimum une base doit contenir 2 groupes de un fichier.
Pour permettre l'écriture simultanée de plusieurs redo logs et ainsi les sécuriser, on crée 2 fichiers ou plus par groupe.



exemple :

le minimum, 2 groupes contenant chacun un fichier (pas de multiplexage donc…)
CREATE DATABASE test DATAFILE 'd:dataoratest_system' SIZE 10M
LOGFILE GROUP 1 ('d:dataoratest_log1a') SIZE 500K,
GROUP 2 ('d:dataoratest_log2a') SIZE 500K;

un peu mieux, LGWR va écrire simultanément dans les 2 fichiers du groupe courant
CREATE DATABASE test DATAFILE 'd:dataoratest_system' SIZE 10M
LOGFILE GROUP 1 ('d:dataoratest_log1a', 'e:dataoratest_log1b') SIZE 500K,
GROUP 2 ('d:dataoratest_log2a', 'e:dataoratest_log2b') SIZE 500K;

Protection max : ceinture ET bretelles (on écrit dans 3 fichiers en même temps)

CREATE DATABASE test DATAFILE 'd:dataoratest_system' SIZE 10M
LOGFILE GROUP 1 ('d:dataoratest_log1a', 'e:dataoratest_log1b' , 'f:dataoratest_log1b') SIZE 500K,
GROUP 2 ('d:dataoratest_log2a', 'e:dataoratest_log2b' , 'f:dataoratest_log1b') SIZE 500K;

si on veut augmenter le temps de rebouclage du cycle d'écriture, ou en d'autres termes augmenter le délai avant écrasement éventuel du contenu du 1er journal, on peut rajouter des groupes supplémentaires…

Multiplexage des logs ARCHIVES

on utilise le paramètre LOG_ARCHIVE_DEST_n (où n est un entier de 1 à 5)
LOG_ARCHIVE_DEST_1 = 'LOCATION=/disk1/arc/'
LOG_ARCHIVE_DEST_2 = 'LOCATION=/disk2/arc/'
LOG_ARCHIVE_DEST_3 = 'LOCATION=/disk3/arc/'
avec le format suivant
LOG_ARCHIVE_FORMAT = arch%t_%s.arc

on aura le résultat suivant (thread 1; log sequence de 100 à 102)

/disk1/arc/arch1_100.arc, /disk1/arc/arch1_101.arc, /disk1/arc/arch1_102.arc,
/disk2/arc/arch1_100.arc, /disk2/arc/arch1_101.arc, /disk2/arc/arch1_102.arc,
/disk3/arc/arch1_100.arc, /disk3/arc/arch1_101.arc, /disk3/arc/arch1_102.arc

l'archivage des redologs files

Ce processus (process ARCH ou "ARCHIVER") est optionnel et permet de faire des restaurations les plus à jour possible.
En l'absence d'archivage, on ne pourra récupérer les données que de la dernière sauvegarde. Avec l'archivage on récupérera ces mêmes données + les modifications qui ont été faites entre la sauvegarde et le crash. (seules les transactions en cours au moment du crash sont perdues).L'archivage permet de garder tout l'historique des fichiers redologs, qui sont recopier sur le répertoire d'archivage dès qu'ils sont pleins (Switch).
Si on n'archive pas les redologs sont écrasés cycliquement, puisque rappelons le, ils sont utilisés de manière séquentielle et circulaire !

* mise en place de l'archivage
autorisation de l'archivage automatique dans INIT.ORA (reg : ORA_TEST_PFILE)
log_archive_start = true
log_archive_dest_1 = "location=C:OracleoradataTESTarchive"
log_archive_format = %%ORACLE_SID%%T%TS%S.ARC
puis arrêt / redémarrage de la base.
Il est possible d'autoriser l'archivage automatique de manière dynamique (sans arrêter la base) :
SQL> ALTER SYSTEM ARCHIVE LOG START;

* déclenchement du mode archivage
SQL> CONNECT INTERNAL
SQL> SHUTDOWN IMMEDIATE
SQL> STARTUP MOUNT
SQL> ALTER DATABASE ARCHIVELOG
SQL> ALTER DATABASE OPEN

* vérification de l'archivage (remplissage ou forcage des logs switchs)
SQL> ALTER SYSTEM SWITCH LOGFILE
voir log_archive_dest_1 et log_archive_format pour voir où les fichiers d'archive créés sur le disque

on peut également vérifier le statut de l'archivage avec la commande :
SQL> ARCHIVELOG LIST

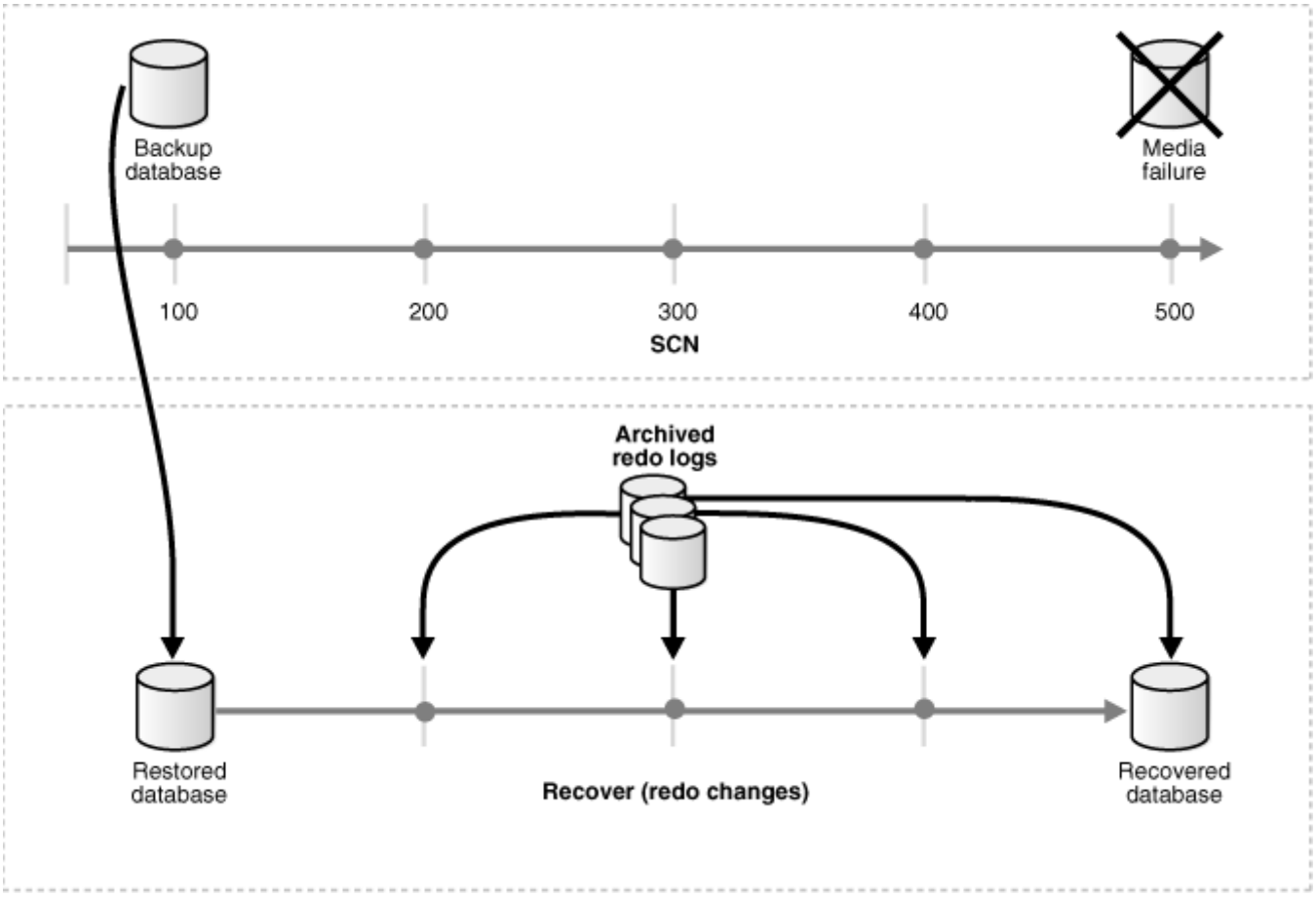
Attention à la prolifération des LOGS archivés, notamment en cas de grosse procédure batch de mise à jour, il est conseillé de désactiver l'archivage !
Il est conseillé également de purger les archives après chaque nouvelle sauvegarde réussie !

Il est possible comme on l'a vu plus haut de multiplexer les logs archivés pour (encore et toujours) plus de sécurité

Principe général de restauration physique

Le principe est d'avoir sous la main une sauvegarde plus ou moins ancienne et de disposer d'un historique de toutes les modifications effectuées depuis cette sauvegarde. Ceci grace aux redolog en ligne et si nécessaire aux redologs archivés.

Une fois que les fichiers nécessaires sont restaurés, le DBA doit initier une restauration du support (media recovery).
Pour que cette restauration assure des données cohérentes, Ceci implique notamment des mécanismes d'application d'image après et d'image avant (rollforward and rollback). Les modifications à appliquer sont choisies sélectivement à partir des redolog online et offline (archivés) et s'appuie sur les no SCN indiquant des points de cohérence dans le temps.



Restauration du control file perdu

en cas de perte d'un control file il suffira
d'arrêtera la base,
de vérifier les spécifications des fichiers de contrôle dans l' INIT.ORA,
de recopier le fichier manquant à partir d'un de ses jumeaux dans le répertoire de destination,
et de rouvrir la base

L'outil RMAN

L'outil RMAN (pour Recovery MANager) est l'outil standard d' Oracle, pour faire les sauvegardes / restaurations de manière efficace et documentée.

Il est disponible depuis la version 8 d'Oracle, et ne cesse d'évoluer au fur et à mesure des versions.

Il permet de :

- faire des sauvegardes physique ou logiques de la base (tablespaces, fichiers de données, de fichiers de contrôle et de fichiers d'archive (archivelog), totales ou partielles
- faire des sauvegardes incrémentielles (différentielles ou cumulatives) au niveau bloc
- documenter les sauvegardes dans un catalogue partagé, ou dans les fichiers de contrôle (dédié donc)
- gérer les périodes de rétention des sauvegardes
- de faire des duplications de base simplement

Types de sauvegarde

RMAN autorise des sauvegardes de type :

- COMPLET (ou FULL) : on sauvegarde tous les blocs
- Incrémentale

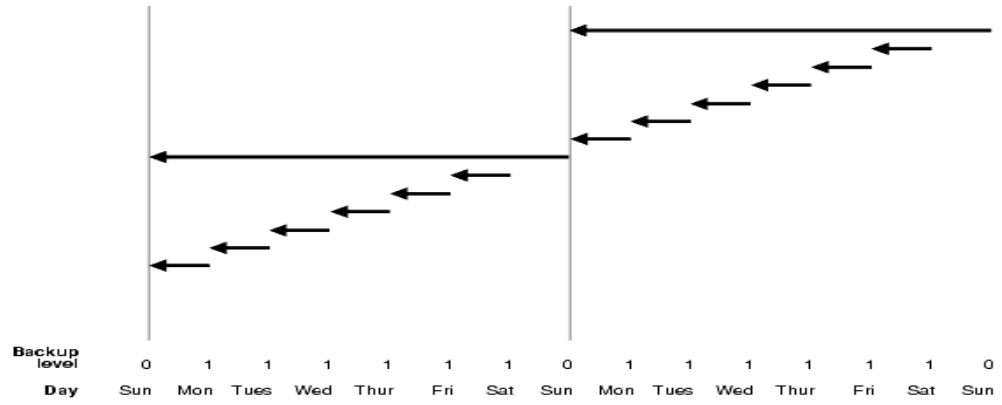
Les sauvegardes incrémentales

Une sauvegarde incrémentale permet comme son nom l'indique de ne pas sauvegarder la totalité des données tous les jours, mais simplement ce qui a été modifié depuis une certaine période.
Il en existe 2 types, qui se différencient par leur mode de résumé des opérations précédentes.

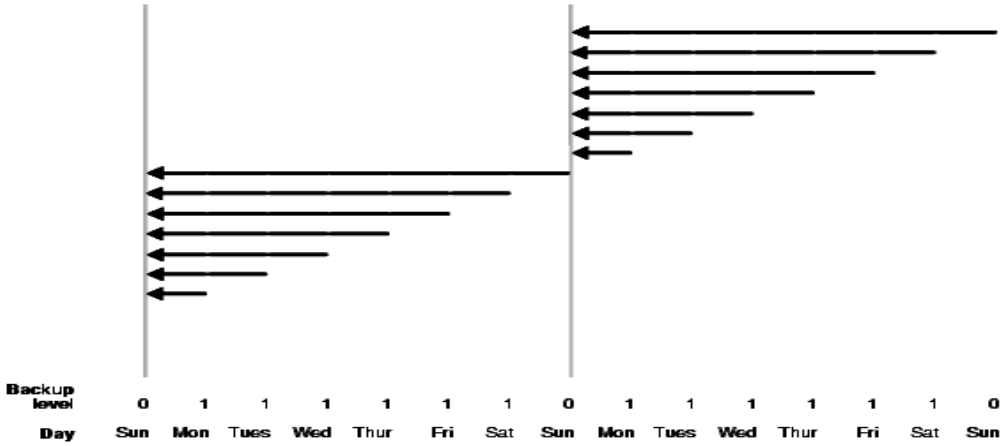
- DIFFERENTIEL : on ne sauvegarde que les blocs modifiés depuis la précédente sauvegarde de niveau n ou inférieur
- CUMULATIF : on ne sauvegarde que les blocs modifiés depuis la précédente sauvegarde de niveau n-1

Pour ce faire RMAN utilise des niveaux de sauvegarde (3) :
le Niveau 0 : Sauvegarde de référence : l'ensemble des blocs contenant des données
le Niveau 1 : Sauvegarde tous les blocs modifiés depuis la plus récente sauvegarde incrémentale de niveau 0
le Niveau 2 : sauvegarde tous les blocs modifiés depuis la plus récente sauvegarde incrémentale de niveau 0, 1 ou 2.

En différentiel ca fait ca :



et en cumulatif (qui permet de gagner du temps lors de la restaure…ca fait ca ;



Quelques exemples de scénarios de sauvegardes restauration avec RMAN

restauration complète

$rman catalog rman/rmanpwd@aliascatalog target / Recovery Manager: Release 9.2.0.1.0 - 64bit Production Copyright (c) 1995, 2002, Oracle Corporation. All rights reserved. connected to target database: DB1 (DBID=1966171371) connected to recovery catalog database RMAN>Run { allocate channel t1 type disk; restore database; recover database; sql "alter database open"; }

De la même façon que précédemment, on alloue un canal de type disk. On restaure la base de données. Etant donné qu'il n'y a aucun paramètre complémentaire, RMAN prend la dernière sauvegarde valide. Ensuite on applique un recover ayant pour effet de réappliquer l'ensemble des archives et des redo logs actifs. Et enfin on ouvre la base pour la rendre disponible à nouveau.

restauration de tablespace à chaud

$rman catalog rman/rmanpwd@aliascatalog target / Recovery Manager: Release 9.2.0.1.0 - 64bit Production Copyright (c) 1995, 2002, Oracle Corporation. All rights reserved. connected to target database: DB1 (DBID=1966171371) connected to recovery catalog database RMAN>Run { sql "alter tablespace TBS_DATA offline"; allocate channel t1 type disk; restore tablespace TBS_DATA; recover tablespace TBS_DATA; sql "alter tablespace TBS_DATA online;"; }

Il faut mettre le tablespace offline avant de pouvoir le restaurer. Ensuite on le restaure et on le remet online.
Les différents types de restauration sont très nombreux, on peut restaurer uniquement un fichier, ou des archives logs, etc…

Introduction au Flashback Recovery

Très succinctement introduit en 9i le concept de 'restauration Flash' (Flash Recovery) a été sensiblement étendue avec Oracle 10g.
Il permet de rapidement revenir en arrière dans le temps, sans nécessiter de sauvegarde préalable, comme dans une restauration classique jusqu'a une certaine date (point in time recovery).

Le 'Flashback recovery' peut suivant les cas être exécuté sans aucun pré requis ou nécessiter des proccess et de la place disque supplémentaire.
Il permet de récupéré la base suite à une erreur utilisateur ou un bug applicatif, qui aurait corrompu ou supprimé des données.

Il existe cinq niveaux de 'Flashback recovery' :

- Flashback drop : le plus simple, il permet de revenir en arrière et d'annuler la suppression d'une table
- Flashback database : permet de revenir en arrière jusqu'a une date/heure donnée
- Flashback table
- Flashback Version query
- Flashback Transaction Qeery

Récupération rapide d'une table supprimée (FLASHBACK DROP RECOVERY)

Les suppressions de table, ne sont plus désormais physiques et défintives, mais mettent simplement l'objet à la corbeille (RECYCLE BIN). Tant que cette dernière n'est pas vidée, il est possible de récupérer l'objet supprimé.

Des infos dans le dictionnaire…

Les informations concernant la corbeille peuvent être consultées dans le dictionnaire :

```
SQL> SHOW RECYCLEBIN
ORIGINAL NAME RECYCLEBIN NAME OBJECT TYPE DROP TIME
———— —————— ———— ———
T1 BIN$EkKm2r93vZfgQKjAhw1XeA==$0 TABLE 2006-04-25:15:51:52
```

ou via les tables USER_RECYCLEBIN et DBA_RECYCLEBIN.

Exemple :

```
SQL> select original_name nom, object_name "nom interne",
operation, createtime "date création ",
droptime "date suppression ", can_undrop "récupérable ? ", can_purge "purgeable ?"
FROM user_recyclebin
```

NOM NOM INTERNE OPERATION date création date suppression récupérable? purgeable?
— ———————— ——— ———————- ———————- ———— ———-
T1 BIN$Ejia7J3wZRXgQKjAhw0prg==$0 DROP 2006-04-25:15:10:17 2006-04-25:15:10:22 YES YES
T2 BIN$Ejia7J3vZRXgQKjAhw0prg==$0 DROP 2006-04-25:15:09:30 2006-04-25:15:09:35 YES YES

note : les colonnes CREATETIME et DROPTIME sont des…VARCHAR (?)

exemples de récupération

SQL> FLASHBACK TABLE T2 TO BEFORE DROP
SQL> FLASHBACK TABLE "BIN$Ejia7J3vZRXgQKjAhw0prg==$0" TO BEFORE DROP

note : on peut donc également utiliser le nom interne ET CE QUEL QUE SOIT LA COMMANDE SQL !
Il est conseillé de l'encadrer par des guillemets pour masquer les caractères spéciaux

Une table T1 peut avoir été supprimée plusieurs fois, et apparaître plusieurs fois dans la corbeille.
Pour éviter toute ambiguité, il faut vérifier le contenu de la poubelle avant récupération et utiliser le nom interne le cas échéant plutot que le nom logique.

exemple
SQL> create table t1 (n integer);
SQL> drop table t1;
SQL> create table t1(new_n integer);
SQL> drop table t1;
et dans la corbeille on a :
NOM nom interne OPERATION date création date suppression
T1 BIN$EkKm2r92vZfgQKjAhw1XeA==$0 DROP 2006-04-25:15:51:14 2006-04-25:15:51:28
T1 BIN$EkKm2r93vZfgQKjAhw1XeA==$0 DROP 2006-04-25:15:51:44 2006-04-25:15:51:52

SQL> desc "BIN$EkKm2r93vZfgQKjAhw1XeA==$0"
Name Type
NEW_N NUMBER(38)

SQL> –c'est bien celle que je veux recuperer…dont acte
SQL> FLASHBACK TABLE "BIN$EkKm2r93vZfgQKjAhw1XeA==$0" TO BEFORE DROP;

Suppression permanente

On peut supprimer une table de la corbeille avec la commande 'PURGE' :

SQL> PURGE TABLE T3

ou indirectement en utilisant des commandes affectant le tablespace utilisateur :

SQL> PURGE TABLESPACE users;
– purge tous les objets du tablespace
SQL> PURGE TABLESPACE users USER scott;
–purge tous les objets de SCOTT du tablespace USERS

ou vider en une fois la poubelle utilisateur ou la poubelle générale :

SQL> PURGE recyclebin;
SQL> PURGE dba_recyclebin;

Eviter la poubelle
On peut également supprimer physiquement et directement une table, en évitant de la conserver dans la corbeille donc, avec l'option 'PURGE' de la commande 'DROP TABLE'
SQL> DROP TABLE T5 PURGE

Ou invalider l'utilisation de la poubelle de manière transversale,
grace au paramètre d'initialisation 'RECYCLEBIN'

SQL> show parameter recyclebin
NAME TYPE VALUE
recyclebin string ON

SQL> ALTER SYSTEM SET recyclebin = OFF;
System Altered

SQL> show parameter recyclebin
NAME TYPE VALUE
recyclebin string OFF

Récupération rapide d'une base (Database FLASHBACK RECOVERY)

La récupération rapide d'une base, permet un retour arrière…grace aux images avant modification, à partir de l'état courant. Il ne nécessite pas comme la restauration jusqu'à une date donnée (POINT IN TIME RECOVERY), de sauvegarde des fichiers préalable. La durée de restauration est indépendante de la taille de la base, mais simplement fonction du nombre d'opérations à annuler.
A la différence du FLASHBACK DROP TABLE qui n'a besoin d'aucune ressource optionnelle, le FLASHBACK DATABASE s'appuie sur :
- une zone de récupération (RECOVERY AREA) qui contient des fichiers LOGs supplémentaires spéciaux : les Flashback Database LOGs
- un process de fond spécifique : RVWR (Recovery Writer) qui écrit dans ces LOGs.

Note : le mode FLASHBACK DATABASE utilise aussi les redologs file en ligne ou non, et nécessite donc d'être en mode ARCHIVELOG !

Passage en mode FLASHBACK DATABASE

On va préciser les spécifications de la zone de récupération (emplacement et taille) et passer la base en mode FLASHBACK. Pour cette opération la base doit être dans l'état montée non ouverte. Trois paramètres d'initialisation nous seront utiles :

DB_RECOVERY_FILE_DEST : répertoire contenant la zone de récupération (Flashback logs)
DB_RECOVERY_FILE_DEST_SIZE : taille en KO(K) ou MO(M) de cette zone
DB_FLASHBACK_RETENTION_TARGET : limite supérieure du temps de récupération en minutes (défaut 1440) Les paramètres de récupération rapide, peuvent être visualisés / spécifiés avec la console entreprise manager 10g, onglet 'maintenance' lien 'paramètres de récupération'

Le passage en mode récupération rapide se fait ensuite avec la commande 'ALTER DATABASE' :

SQL> alter database flashback on;

alter database flashback on*ERROR at line 1:ORA-38706: Cannot turn on FLASHBACK DATABASE logging.ORA-38707: Media recovery is not enabled.

exemple d'étapes manuelles :

SQL> –Passage en mode archivage si nécessaire…
SQL> select log_mode from v$database;
LOG_MODE
————
NOARCHIVELOG
SQL> shutdown immediate;
Database shhutdown
SQL> startup mount
ORACLE instance started.
Total System Global Area 499122176 bytes
Fixed Size 2021824 bytes
Variable Size 142607936 bytes
Database Buffers 348127232 bytes
Redo Buffers 6365184 bytes
Database mounted

SQL> alter system archive log start;
System altered.
SQL> alter database archivelog;
Database altered.

SQL> alter database open;
Database altered.
SQL> — passage en mode Flashback
SQL> alter database flashback ON;
Database altered.
SQL> –vérification pour les inquiets…
SQL> SELECT FLASHBACK_ON FROM v$database;
FLASHBACK_ON
————
NO

flashback coté systeme

SQL> show parameter recovery_file_dest
NAME TYPE VALUE
—- —- —-
db_recovery_file_dest string /oracle/10GDB/flash_recovery_a rea
db_recovery_file_dest_size big integer 2G

$ pwd
/oracle/10GDB/flash_recovery_area/PORT64/flashback

```
oracle@sli-portail-4 flashback]$ ls -l
total 242352
-rw-r—– 1 oracle dba 8200192 mai 3 17:23 o1_mf_25k5pzmz_.flb
-rw-r—– 1 oracle dba 8200192 mai 3 18:00 o1_mf_25klr420_.flb
-rw-r—– 1 oracle dba 4104192 mai 3 18:14 o1_mf_25knwwbz_.flb
-rw-r—– 1 oracle dba 3989504 mai 3 18:51 o1_mf_25koq0w9_.flb
-rw-r—– 1 oracle dba 3989504 mai 3 19:00 o1_mf_25kqw7bf_.flb
-rw-r—– 1 oracle dba 3989504 mai 3 19:15 o1_mf_25krdxhx_.flb
-rw-r—– 1 oracle dba 3989504 mai 3 19:44 o1_mf_25ks9462_.flb
-rw-r—– 1 oracle dba 3989504 mai 3 20:00 o1_mf_25kv0c3h_.flb
-rw-r—– 1 oracle dba 3989504 mai 3 20:21 o1_mf_25kvyobd_.flb
-rw-r—– 1 oracle dba 3989504 mai 3 20:59 o1_mf_25kx567w_.flb
```

ou l'on voit que les fichiers flashback ont l'extension '.flb' et sont créés de manière relativement régulière, quelle que soit l'activité de la base.

Note : la zone de restauration rapide (flashback area) sert à stocker les Flshback logs mais aussi tous les fichiers important de la BD, à savoir :
- datafiles,
- control files,
- redologs files curants,
- redolog files archivés.

Peut-on faire la récupération ?

En d'autres terme, est que les FLASHBACK LOGs permettent de 'remonter' suffisamment loin dans le temps ?
C'est le paramètre db_flashback_retention_target qui va le déterminer :

```
SQL> SELECT VALUE/60 "Durée en heures "
FROM v$PARAMETER
WHERE NAME = 'db_flashback_retention_target';

Durée en heures
———————
24
```

Quels flashback logs sont utiles ? (ceux depuis 9H30 du matin, pour etre sur, par exemple ici):

```
select NAME , LOG# , THREAD# ,SEQUENCE#,BYTES ,
TO_CHAR(FIRST_TIME,'DD/MM HH24:MI') "Modifié à "
from V$FLASHBACK_DATABASE_LOGFILE
WHERE TO_CHAR(FIRST_TIME,'DD/MM HH24:MI') >= '05/05 09:30'

NAME LOG# THREAD# SEQUENCE# BYTES Modifié à
…/flashback/o1_mf_25m79d8f_.flb 57 1 143 3981312 05/05 09:35
…/flashback/o1_mf_25m95vkq_.flb 58 1 144 3981312 05/05 10:00
…/flashback/o1_mf_25m9nqn0_.flb 59 1 145 3981312 05/05 10:00
```

Scénarios de backup et restore Oracle 11g

Vérification des hypothèses minimales de sécurité

Base en mode archivelog ? :

```
SQL>  SELECT log_mode FROM v$database;
LOG_MODE
--------
ARCHIVELOG

en cas de problème la passer en mode ARCHIVELOG :
SQL> SHUTDOWN IMMEDIATE ; (ou NORMAL mais pas ABORT !)
SQL> STARTUP MOUNT;
SQL> ALTER DATABASE ARCHIVELOG;
SQL> ALTER DATABASE OPEN;
```

Processus d'Archivage et espace disque OK ?

On peut tester que l'archivage fonctionne en forçant un switch de redolog et donc…une archive.

```
SQL> ARCHIVE LOG LIST;
...
Current Log sequence 22
...
SQL> ALTER SYSTEM SWITCH LOGFILE;
SQL> ARCHIVE LOG LIST;
...
Current Log sequence 23
...
```

Vérifier aussi qu'il y a de la place sur la destination d'archivage (parametre LOG_ARCHIVE_DEST) ou par défaut $ORACLE_HOME/dbs

Redolog files multiplexés ? :

Ici on veut plusieurs membres par groupes.

```
SQL> select group#, status, member
from v$logfile;
```

GROUP#	STATUS	MEMBER
1	ONLINE	/REDO/log_11.rdo
2	ONLINE	/REDO/log_21.rdo
3	ONLINE	/REDO/log_31.rdo

4	ONLINE	/REDO/log_41.rdo

et si ce n'est pas le cas (comme ici 4 GROUP avec chacun 1 fichier non multiplexé) on peut rajouter des membre SUR UN DISQUE DIFFERENT bien sur ! :

```
SQL>  ALTER DATABASE ADD LOGFILE MEMBER
'/DISK2/log_12.rdo' TO GROUP 1,
'/DISK2/log_22.rdo' TO GROUP 2,
'/DISK2/log_32.rdo' TO GROUP 3,
'/DISK2/log_42.rdo' TO GROUP 4;
```

Control File multiplexés :

```
SQL> select name
from v$controlfile;
```

NAME
/data/ctrl1PRDUN.ctl
/data/ctrl2PRDUN.ctl

et si ce n'est pas le cas (comme ici 2 fichier mais dans le même filesystem !!) :

```
SQL> SHUTDOWN IMMEDIATE;$> cp /data/ctrl1.ctl  /DISK2/ctrl3.ctl
SQL> CREATE PFILE FROM SPFILE;
$> modifier le paramètre CONTROL_FILE du nouveau fichier init.ora
pour y faire référence au nouveau control file(dans $ORACLE_HOME/dbs par défaut)
CONTROL_FILES=(....,nouveau_ctl.ctl)SQL> -- demarrer la base avec le pfile
SQL> STARTUP PFILE=$ORACLE_HOME/dbs/init<ORACLE_SID>.ora
SQL> recreer le spfile
SQL> CREATE SPFILE FROM PFILE;
SQL>-- redemarrer normalement...
SQL> SHUTDOWN IMMEDIATE; (ou abort...)
SQL> STARTUP
```

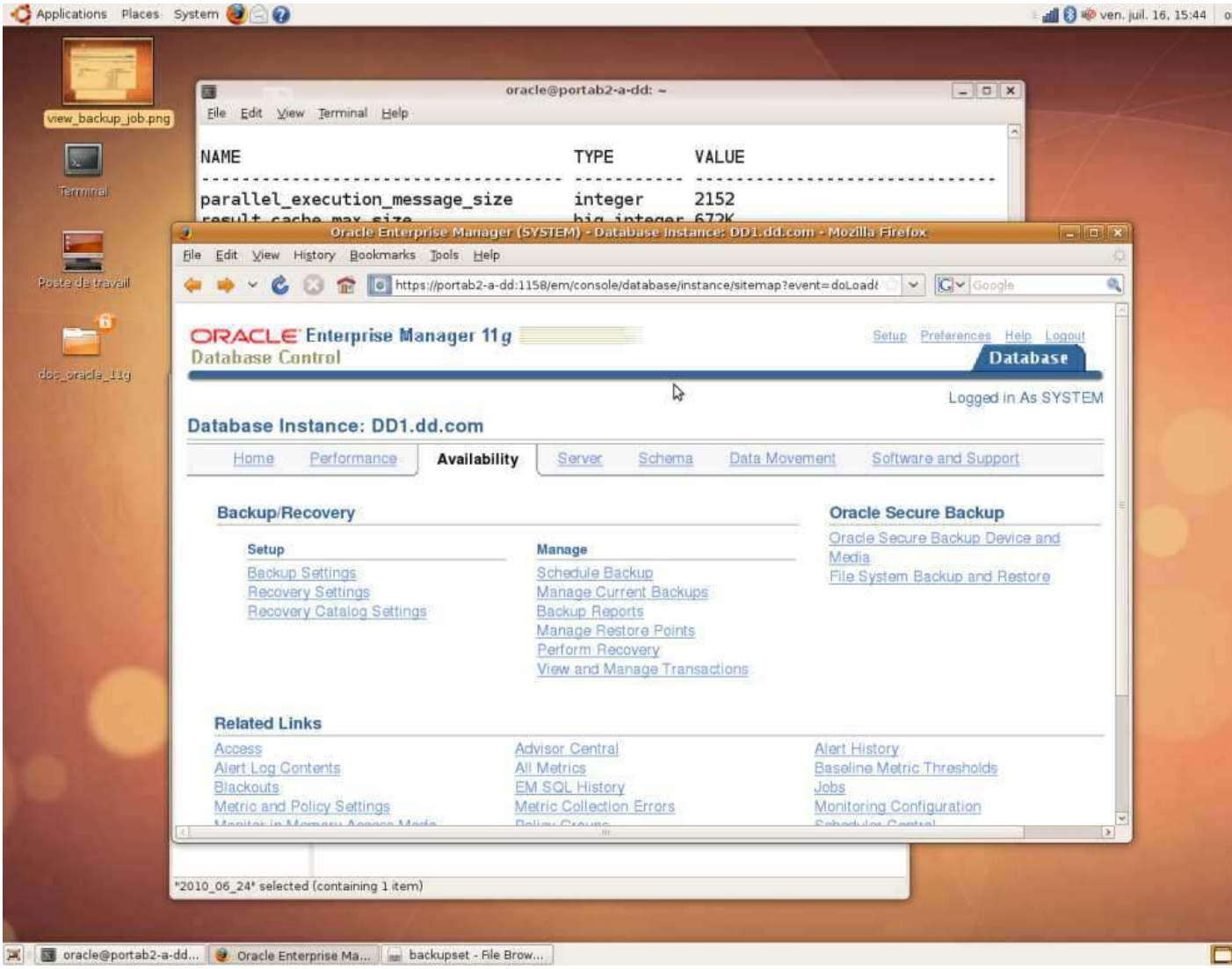
sauvegarder une base avec RMAN en 5 clics !

RMAN est réputé compliqué à utiliser car il faut connaitre son langage de SCRIPT.
Tant pis pour les geeks … on va mettre en place une sauvegarde quotidienne (automatique et incrémentale) de la base Oracle 11g, sans une seule ligne de commande avec Enterprise Manager…en 6 clic.

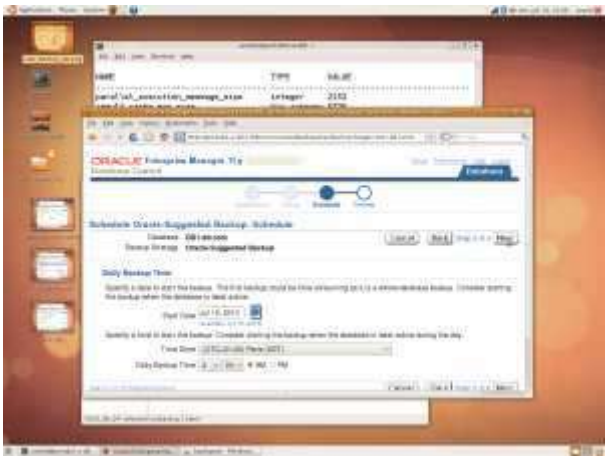
Prérequis avoir une Flashback Recovery Area définie et ed taille suffisante pour accueillir les backups…
Verifier avec :

SQL> SHOW PARAMETER DB_RECOVERY

1) dans la console EM aller sur l'onglet Availability / Disponibilité, puis cliquer sur le lien 'schedule backup'



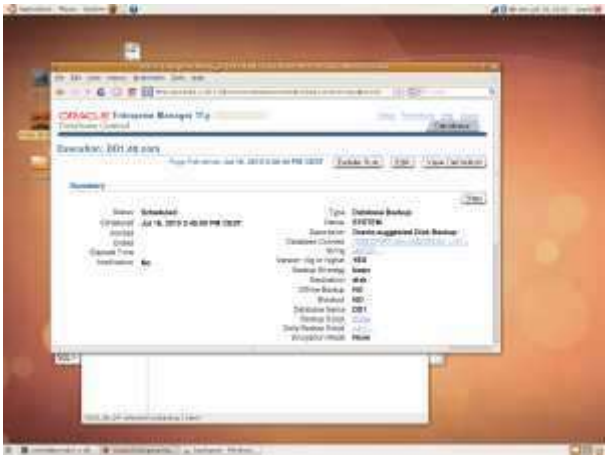
Laisser les choix des écrans suivants par défaut (backup sur disque dans la FLASH RECOVERY AREA, sauf celui de l'heure planifiée du backup …choisissez une heure :



Cliquer sur Next, puis soumettre le JOB pour automatiser les sauveagrdes quotidiennes.
Pour info le script RMAN généré automatiquement est le suivant :

Daily Script:
run {
allocate channel oem_disk_backup device type disk;
recover copy of database with tag 'ORA$OEM_LEVEL_0';
backup incremental level 1 cumulative copies=1 for recover of copy with tag
'ORA$OEM_LEVEL_0' database;
}

Si vous souhaitez vérifier votre JOB , vous pouvez clique sur le bouton 'View Job' :



Bon bah…c'est fini

CHAPITRE H - Optimisation

Optimiseur d'Oracle et statistiques

L'exécution d'un ordre SQL

Les principales étapes mises en jeu pour l'exécution d'un ordre SQL sont les suivantes :

1. Vérification syntaxique (présence et ordre des mots clés,…) réalisée par le 'parser'
2. Vérification sémantique (est ce les noms des objets sont corrects, les droits, etc)
3. Optimisation de la requête et recherche du meilleur plan d'exécution
4. Exécution effective de la requête et rapatriement des lignes du résultat

Nous allons nous focaliser ici sur l'étape d'optimisation.

L'optimiseur statistique

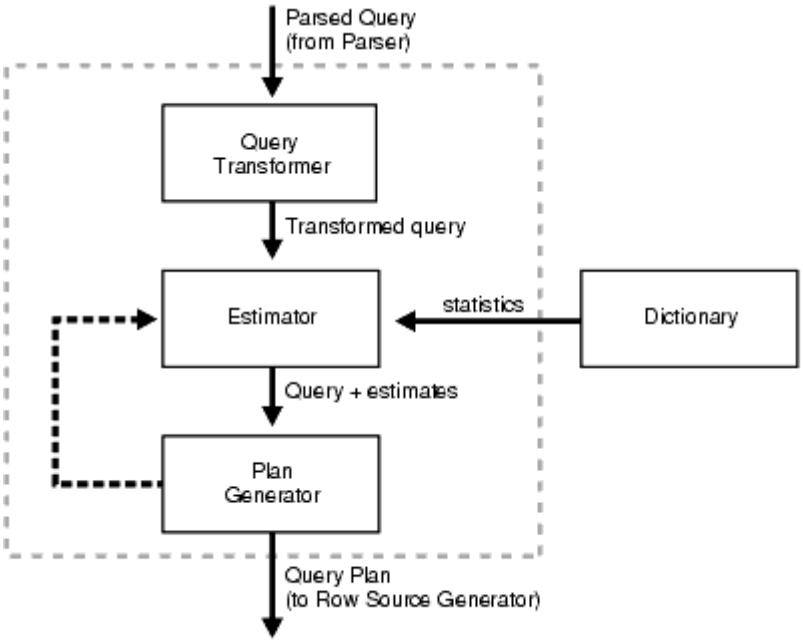
Pour exécuter une requête SQL de la manière la + efficace, le moteur d'Oracle utilise un optimiseur, qui détermine un plan / chemin d'exécution jugé optimal.
A partir de la version 9, cet optimiseur est généralement basé sur le coût de la requête (COST

BASED) et s'appuie pour ce faire sur des statistiques qui quantifient la qualité des données et des index.

Remarques :
- historiquement dans les versions précédentes, l'optimiseur s'appuyait sur des règles internes figées (Rule Based Optimizer ou RBO).
- l'optimiseur pouvant donnere des résultats très différents d'une version à une autre, il est possible de conserver un fonctionnement de l'optimiseur d'une version précédente grâce au paramètre OPTIMIZER_FEATURES_ENABLE

```
SQL> ALTER SYSTEM SET OPTIMIZER_FEATURES_ENABLE=10.0.0;
```

L'optimiseur effectue plusieurs taches :



Phase 1 : Réécriture de la requête (QUERY TRANFORMATION ou QUERY REWRITE)

Plusieurs techniques peuvent être utilisées pour transformer / simplifier la requête avant la recherche du plan d'exécution :

1. fusion des vues (view merging)
2. insertion de prédicate (predicate pushing)
3. extraction des sous-requêtes (subquery unnesting) : les sous-requêtes sont transformées en jointures…
4. utilisation de vues matérialisées (query rewrite with MVs) : si un SELECT est proche d'une vue matérialisée, c'est le résultat de la vue matérialisée (stocké / pré calculé) qui est utilisé

Les statistiques sur les objets

on y trouve par exemple des :

- statistiques sur les Tables : nb de lignes, nb de blocs, lg moyenne de lignes
- statistiques sur les Colonnes : (nb de valeurs distinctes, nb de valeurs NULL, histogramme de distribution des valeurs
- statistiques sur les Index : nb de feuilles, niveau (level) ou profondeur des index, facteur de grappe (clustering factor)
- statistiques sur le Système d'exploitation : performance et utilisation des E/S, performance et utilisation du CPU

Ces statistiques sont disponibles dans le dictionnaire notamment dans les tables suivantes :

- DBA_TABLES
- DBA_TAB_STATISTICS
- DBA_TAB_COL_STATISTICS
- DBA_TAB_HISTOGRAMS
- DBA_INDEXES
- DBA_IND_STATISTICS

On peut par exemple récupérer les statistiques des tables de SCOTT ainsi :

```
SQL> select table_name, num_rows, blocks, empty_blocks,
avg_row_len, sample_size, last_analyzed
from dba_tab_statistics
where owner='SCOTT';

TABLE_NAME NUM_ROWS BLOCKS EMPTY_BLOCKS     AVG_ROW_LEN SAMPLE_SIZE       LAST_ANALY
---------- -------- ------ ------------ ----------- ----------- ------------------
BONUS      0        0      0            0           0           07/11/2008 16:05:46
DEPT       4        5      0            20          4           07/11/2008 16:05:46
DUMMY      1        5      0            2           1           07/11/2008 16:05:46
EMP        14       5      0            37          14          07/11/2008 16:05:46
SALGRADE   5        5      0            10          5           07/11/2008 16:05:46
TESTECB    5376     80     0            50          5376   09/11/2008 04:06:04
```

On peut aussi vérifier la dernière mise à jour de ces infos dans les vues USER_TABLES ou DBA_TABLES :

```
SQL> select table_name,last_analyzed
from  user_tables;
TABLE_NAME      LAST_ANALYZED
------------    -------------------
DD              24/08/2009 22:00:14
DD_DBA_VIEWS    28/09/2008 12:20:31
```

Collecte automatique des statistiques

Les stats sont automatiquement collectés via le job GATHER_STATS_JOB. et ce pour tous les objets qui ont soit des stats manquantes soit des stats obsoletes.

Ce Job est créé (normalement!) automatiquement lors de la création de la base, et doit (normalement) s'exécuter pendant la fenêtre de maintenance de la base (par défaut chaque nuit de 10H à 6H du mat).

On peut vérifier que ce job est présent et s'exécute correctement avec la console em ou grid, dans l'onglet ' Administration ' rubrique ' gestion des statistiques ' ou en SQL :

```
SQL> select owner,job_name,job_creator
from dba_scheduler_jobs;
OWNER   JOB_NAME            JOB_CREATOR
-----   ---------           -----------
SYS     GATHER_STATS_JOB      SYS
SYS     AUTO_SPACE_ADVISOR_JOB SYS
```

et si on veut des infos plus précises sur les récentes exécutions du JOB on regarde la vue des LOG de JOBs : dba_scheduler_job_log. Ca nous donne

```
SQL> select LOG_ID, LOG_DATE, JOB_NAME, STATUS
from dba_scheduler_job_log
where job_name like '%STAT%';
LOG_ID  LOG_DATE                          JOB_NAME         STATUS
------  --------------------------------- ---------------- ---------
10198     27/04/2010 22:27:57.812343 +02:00 GATHER_STATS_JOB SUCCEEDED
10479     17/05/2010 22:11:36.792538 +02:00 GATHER_STATS_JOB SUCCEEDED
10465     13/05/2010 22:02:28.469371 +02:00 GATHER_STATS_JOB SUCCEEDED
10165     23/04/2010 22:26:20.788045 +02:00 GATHER_STATS_JOB SUCCEEDED
10314     04/05/2010 22:11:01.983675 +02:00 GATHER_STATS_JOB SUCCEEDED
```

Remarque : il se peut que toutes les LAST_ANALYZED ne soient pas à la même date, apparemment Oracle ne fait un COMPUTE STATISTIC que sur les tables qui le nécessitent…

Les différents stratégies d'optimisation d' Oracle (OPTIMIZER GOAL)

L'optimiseur a 2 stratégies au choix :

- minimiser les ressources pour obtenir TOUTES les leslignes (best throughput)
- minimiser le temps de réponse pour obtenir LA PREMIER ligne de résultat (best response time)

Le choix de la stratégie peut être configuré à 3 niveaux :

- au niveau base (ALTER SYSTEM SET OPTIMIZER_MODE = …)
- au niveau session (ALTER SESSION SET OPTIMIZER_MODE= …)
- au niveau de l'ordre SQL (HINT)

Les différentes valeurs de OPTIMIZER_MODE sont les suivantes :

FIRST_ROWS (déconseillé par Oracle)

FIRST_ROWS_n (avec n= 1, 10, 100, ou 1000) -> temps réponse minimal

ALL_ROWS -> ressources minimales (c'est de défaut !)

Voici quelques exemples :

```
SQL> — changement au niveau base DYNAMIQUEMENT (sans restart instance)
SQL> alter system set optimizer_mode =first_rows_10 scope=memory
SQL> — changement au niveau session
SQL> alter session set optimizer_mode =all_rows
SQL> — et un HINT qui force l'optiseur au sein même d'un ordre SQL
SQL> select /*+ ALL_ROWS */ * from scott.emp
```

rem : ATTENTION !! Les erreurs de syntaxe dans les HINTs sont ignorées

```
SQL> select /*+ TURLUTUTU */ * from scott.emp;
SQL> …
SQL> 14 lignes ramenées
```

CHAPITRE I -Exercices

Serveurs utilisateur dédiés et partagés

Au niveau architecture système, il existe essentiellement 2 architectures :

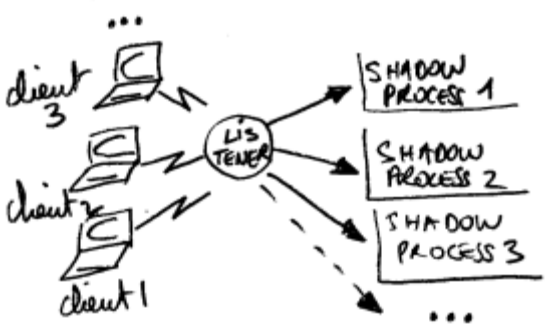
- l'une dédiée (dedicated)
- l'autre partagée (shared ou multithreaded)

Par défaut Oracle fonctionne dans le premier mode.

Le choix du mode de fonctionnement impactera fortement la charge du système en terme de nombre de processus.

Serveurs dédiés (Oracle dedicated server)

En architecture standard Oracle, un process serveur (shadow process) est dédié à chaque session cliente.
En clair il ya autant de process sur le serveur que de clients connectés.



On peut lister ces process sur le serveur en sachant qu'ils ont un attribut 'LOCAL=NO'

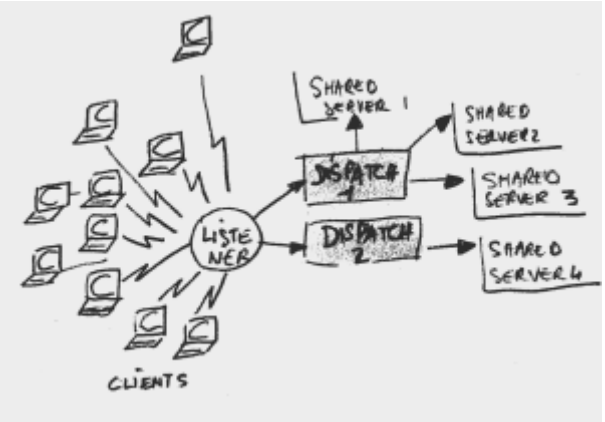
```
$> ps -ef|grep LOCAL
oracle 270380 1 0 Mar 03 - 10:05 oracleTEST (LOCAL=NO)
oracle 995516 4636838 0 13:42:07 pts/6 0:00 grep LOCAL
oracle 1024208 1 0 10:17:01 - 0:08 oracleTEST (LOCAL=NO)
oracle 4419740 1 0 11:38:21 - 0:00 oracleTEST (LOCAL=NO)
oracle 4468764 1 0 11:24:12 - 1:13 oracleTEST (LOCAL=NO)
oracle 4481238 1 0 10:15:28 - 0:07 oracleTEST (LOCAL=NO)
```

C'est donc le mode par défaut. Il est satisfaisant dans la plupart des cas, hormis lorsque le nombre d'utilisateurs connectés simultanément 'sessions) est très important. Dans ce cas il sera peut-être intéressant de passer en architecture partagée pour diminuer le nombre de processus sur le serveur…

Serveurs partagés (Shared Server)

L'architecture clients / serveurs partagés (shared servers), baptisée 'MultiThread' dans les précédentes versions d'Oracle permet de ne pas dédier un process serveur à chaque demande d'ouverture de session d'un client.
Un ou plusieurs process partagés pourront ainsi supporter un grand nombre de connexions simultanées, sans augmenter linéairement le nombre de process serveurs.

architecture client-serveur partagé



un "listener" SQL*Net est obligatoire, même si le client est situé sur la même machine que le serveur. On sera alors en pseudo accès distant, en "Loopback"

Les process en architecture serveurs partagés

Il faut au minimum 3 process pour que ça marche :

- un listener , (LISTENER) process réseau qui attend les demandes de connexion utilisateur, et connecte le process utilisateur à un serveur partagé (ou à un serveur dédié en cas de demande expresse)
- un dispatcher (ora_D00n) , qui place la requête dans une file d'attente (request queue) où le premier serveur partagé disponible se servira
  le nombre initial de dispatchers est donné par le paramètre DISPATCHERS de l'init.ora

- un serveur partagé (ora_S00n) , qui traite effectivement la requête, accède au SGBD et met le résultat dans la file d'attente de réponse (response queue) du dispatcher demandeur.
  Il y a entre SHARED_SERVERS and MAX_SHARED_SERVERS (paramètres de l'ini.ora) qui peuvent être créés.

On peut voir sur le listing suivant, outre les process de fond du serveur, les process dispatcher et shared

```
$ > ps -ef|grep PPRUN
oracle 291042 1 49 Mar 03 - 53:21 ora_d003_PPRUN
oracle 315414 1 0 Mar 03 - 71:35 ora_d002_PPRUN
oracle 327688 1 0 Mar 03 - 112:29 ora_d001_PPRUN
oracle 368882 1 0 Mar 03 - 0:05 ora_qmnc_PPRUN
oracle 381180 1 2 Mar 03 - 51:42 ora_d000_PPRUN
oracle 385276 1 0 Mar 03 - 15:00 ora_mmnl_PPRUN
oracle 389214 1 76 13:41:14 - 0:22 ora_s001_PPRUN
oracle 663766 1 0 Mar 03 - 2:04 ora_pmon_PPRUN
oracle 819304 1 0 Mar 03 - 0:54 ora_mmon_PPRUN
oracle 843914 1 0 Mar 03 - 2:25 ora_smon_PPRUN
oracle 848122 1 0 Mar 03 - 6:08 ora_ckpt_PPRUN
oracle 925788 1 0 Mar 03 - 1:10 ora_psp0_PPRUN
oracle 930042 1 0 Mar 03 - 9:19 ora_lgwr_PPRUN
oracle 934012 1 2 Mar 03 - 5:55 ora_dbw0_PPRUN
oracle 938124 1 0 Mar 03 - 0:16 ora_mman_PPRUN
oracle 942294 1 0 Mar 03 - 0:00 ora_reco_PPRUN
oracle 954520 4636838 0 13:43:57 pts/6 0:00 grep PPRUN
oracle 962780 1 0 Mar 03 - 0:02 ora_q000_PPRUN
oracle 970974 1 0 Mar 03 - 0:04 ora_q001_PPRUN
oracle 4501582 1 0 13:38:20 - 0:49 ora_s000_PPRUN
```

rem : la zone mémoire utilisateur standard (PGA), dans ce type d'architecture, ne contient qu'une pile et les informations vraiment spécifiques du process user. Les autres informations normalement contenues en PGA sont déportées en SGA.

rem : en architecture serveur partagée, la taille de la SGA doit être revue sensiblement à la hausse du fait de l'incorporation des infos des PGAs utilisateur.

Le paramétrage

SQL> show parameter shared_server

NAME TYPE VALUE
------------------ ------ -----
max_shared_servers integer 40
shared_server_sessions integer
shared_servers integer 10

SHARED_SERVERS
————————-

SHARED_SERVERS spécifie le nb de process lancés systématiquement au démarrage de l'instance. En cas de montée en charge, ce nombre pourra augmenter, mais ne descendra jamais en dessous de cette limite dans le cas inverse.

MAX_SHARED_SERVERS
———————-
MAX_SHARED_SERVERS spécifie le nb maximum de process qui peuvent tourner simultanément. Il est compris entre SHARED_SERVER et PROCESSES.
Imposer cette limite permet de laisser des 'slots' libres pour d'autre process comme les proces dédiés par exemple.
si MAX_SHARED_SERVERS n'est pas spécifié, un SHARED SERVER sera déclenché tant que le nb de 'slots' de libre est > a 1/8 max(nb max process)
ou 2 si PROCESSES < 24.

Configuration d'Oracle Net Service

Le but de cet exercice est de configurer l'accès à une base distante à partir de notre serveur de données.
On utilisera pour ce faire Enterprise manager et la résolution locale de nom.
En résumé notre seveur deviendra le 'client' d'une nouvelle base de données, et l'on constituera ainsi une architecture minimale à 3 poles (3-tiers).
Chaque stagiaire devra créer un nouveau nom de service réseau TESTn, qui pointe sur la base ORCL du serveur de cours.

Etapes :
- lancer la console Enterprise Manager (http://serveur_de_cours:no_port/em)
- vérifier que le LISTENER est actif
- cliquer sur le lien 'Administration Service réseau' en bas à gauche
- choisissez 'Résolution locale des noms' dans la liste déroulante 'Administrer'
- créer un nouveau nom de service
- puis une nouvelle adresse (avec les bons paramètres TCP/IP) sur laquelle va pointer ce nom de service
- tester une connexion scott/tiger sur TESTn
- quels fichiers ont été modifiés sur le serveur ?

Gestion des Users et des Roles - 1

A l'aide la console EM
- creer un utilisateur USERn
avec les caractéristiques suivantes :
Espace disque logique par defaut TBSn, TBS temporaire TEMP, quota sur TBSn 50K
et les privilèges minimum pour pouvoir créer des tables
- vérifier la possibilité de création d'une table et les caractéristiques son implantation
en créant une table sous i*SQLPLUS
: CREATE TABLE nom_table (nom_col type , nom_col type, …)

Pour la suite des TPs donner le role DBA a USERn

Tablespaces et stockage

Evolution des TBS
——————————
Visualisez les infos sur le TBS 'USERS'
Quel taille fait-il? Quel est son pourcentage de remplissage actuel ?
Quelles sont ses règles d'extension ? (méthode auto ou non, taille de l'extension, limite)

Création (pseudo) manuelle de TBS
————————————————
En vous aidant de la console Enterprise Manager pour afficher l'ordre SQL qui va bien.
Connectez-vous DB1/DBA1.
Préparez la création d'un espace disque logique 'TBSn' permanent,
géré localement, contenant un fichier 'tbsn_1.dbf'
de 50K, auto extensible jusqu'a 150K par paquet de 50K.

Prise en main de la console enterprise

Prise en main de la base : configuration et architecture générale
Utilisation basique d'Enterprise Manager Console (EM)
(http://serveur_de_cours:no_port/em)

Ressources EM utilisées : Page d'accueil et
onglet 'Administration', rubriques 'stockage', 'configuration', 'objets de la base' et 'utilisateurs'

Connectez-vous dba1/dba1 et répondez aux sections suivantes :

1) quelle version d'Oracle est exactement utilisée ?
quelle est la configuration du serveur hôte (RAM, disque, CPU, @IP)
quelle est la taille mémoire globale (SGA) utilisée par Oracle,
et quelle est la taille réservée au cache de tampon des données ?
2) quelle taille fait en gros la base physique et ou sont implantés ses fichiers ?
3) les fichiers de contrôle sont il multiplexés ? Si oui qu'en pensez-vous ?
4) les fichiers journaux sont ils multiplexés ?
5) combien y a t il de tablespaces purement 'utilisateur' ?
6) en recherchant les paramètres d'initialisation 'NLS' (National Language Support) de la base, précisez dans quelle langue est configuré le serveur ?

Gestion des users et des roles 2

Création de rôle de type Infocentre et développeur :
- créer un user USERn_INT (pous faire des interrogations uniquement)
- créer un rôle CONSn qui ait le droit de consulter la table EMP et DEPT de SCOTT.
- affecter ce role à USERn_INT et tester
- créer un rôle de type developpeur DEVn
- doit-on lui donner le role 'CONNECT' ? 'DBA' ?
- quels sont les privilèges système adéquats pour qu'il puisse travailler…les affecter au role
- affecter le rôle à USERn et tester la création d'une procédure stockée vide par exemple

Vérification des droits d'accès du rôle SELECT_CATALOG_ROLE
(accès complet en lecture aux tables du dictionnaire appartenant a SYS) :

- donner 'SELECT_CATALOG_ROLE' à USERn:
- vérifier qu'il existe une table 'LINK$' dans le schéma 'SYS'
qui documente les accès réseau a une autre base
- vérifier les droits d'accès aux objets du rôle 'SELECT_CATALOG_ROLE'
- pouvez vous accéder au dictionnaire à partir de ce USER ? à la table 'LINK$' ?
- à votre avis pourquoi ?

exercice - flashback table

Récupération de table grâce à la recovery Area

Vérifier que la poubelle est active au niveau instance
et l'activer si nécessaire
Créer une table T1 avec une colonne CHAR, insérer une ou 2 lignes
Créer deux tables T2 et T3 avec une colonne INTEGER
Supprimer la table T1 et T2 logiquement
Supprimer la table T3 physiquement
Vérifier le contenu de la poubelle
Récupérer la table T1
Purger si nécessaire la corbeille

CHAPITRE J -Corrigés

Serveurs utilisateur dédiés et partagés

Au niveau architecture système, il existe essentiellement 2 architectures :

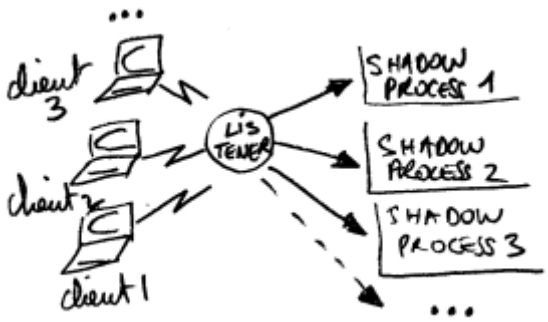
- l'une dédiée (dedicated)
- l'autre partagée (shared ou multithreaded)

Par défaut Oracle fonctionne dans le premier mode.

Le choix du mode de fonctionnement impactera fortement la charge du système en termes de nombre de processus.

Serveurs dédiés (Oracle dedicated server)

En architecture standard Oracle, un process serveur (shadow process) est dédié à chaque session cliente.
En clair il ya autant de process sur le serveur que de client connectés.



On peut lister ces process sur le serveur en sachant qu'ils ont un attribut 'LOCAL=NO'

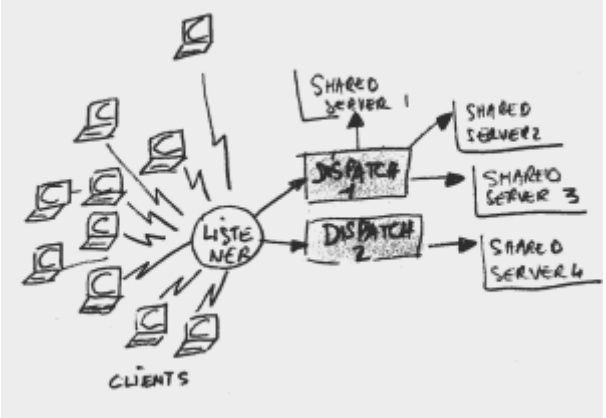
```
$> ps -ef|grep LOCAL
oracle 270380 1 0 Mar 03 - 10:05 oracleTEST (LOCAL=NO)
oracle 995516 4636838 0 13:42:07 pts/6 0:00 grep LOCAL
oracle 1024208 1 0 10:17:01 - 0:08 oracleTEST (LOCAL=NO)
oracle 4419740 1 0 11:38:21 - 0:00 oracleTEST (LOCAL=NO)
oracle 4468764 1 0 11:24:12 - 1:13 oracleTEST (LOCAL=NO)
oracle 4481238 1 0 10:15:28 - 0:07 oracleTEST (LOCAL=NO)
```

C'est donc le mode par défaut. Il est satisfaisant dans la plupart des cas, hormis lorsque le nombre d'utilisateurs connectés simultanément 'sessions) est très important. Dans ce cas il sera peut-être intéressant de passer en architecture partagée pour diminuer le nombre de processus sur le serveur…

Serveurs partagés (Shared Server)

L'architecture clients / serveurs partagés (shared servers), baptisée 'MultiThread' dans les précédentes versions d'Oracle permet de ne pas dédier un process serveur à chaque demande d'ouverture de session d'un client.
Un ou plusieurs process partagés pourront ainsi supporter un grand nombre de connexions simultanées, sans augmenter linéairement le nombre de process serveurs.

architecture client-serveur partagé



un "listener" SQL*Net est obligatoire, même si le client est situé sur la même machine que le serveur. On sera alors en pseudo accès distant, en "Loopback"

Les process en architecture serveurs partagés

Il faut au minimum 3 process pour que ça marche :

- un listener , (LISTENER) process réseau qui attend les demandes de connexion utilisateur, et connecte le process utilisateur à un serveur partagé (ou à un serveur dédié en cas de demande expresse)
- un dispatcher (ora_D00n) , qui place la requête dans une file d'attente (request queue) où le premier serveur partagé disponible se servira
  le nombre initial de dispatchers est donné par le paramètre DISPATCHERS de l'init.ora
- un serveur partagé (ora_S00n) , qui traite effectivement la requête, accède au SGBD et met le résultat dans la file d'attente de réponse (response queue) du dispatcher demandeur.
  Il y a entre SHARED_SERVERS and MAX_SHARED_SERVERS (paramètres de l'ini.ora) qui peuvent être créés.

On peut voir sur le listing suivant, outre les process de fond du serveur, les process dispatcher et shared

```
$ > ps -ef|grep PPRUN
oracle 291042 1 49 Mar 03 - 53:21 ora_d003_PPRUN
oracle 315414 1 0 Mar 03 - 71:35 ora_d002_PPRUN
oracle 327688 1 0 Mar 03 - 112:29 ora_d001_PPRUN
oracle 368882 1 0 Mar 03 - 0:05 ora_qmnc_PPRUN
oracle 381180 1 2 Mar 03 - 51:42 ora_d000_PPRUN
oracle 385276 1 0 Mar 03 - 15:00 ora_mmnl_PPRUN
oracle 389214 1 76 13:41:14 - 0:22 ora_s001_PPRUN
oracle 663766 1 0 Mar 03 - 2:04 ora_pmon_PPRUN
oracle 819304 1 0 Mar 03 - 0:54 ora_mmon_PPRUN
oracle 843914 1 0 Mar 03 - 2:25 ora_smon_PPRUN
oracle 848122 1 0 Mar 03 - 6:08 ora_ckpt_PPRUN
oracle 925788 1 0 Mar 03 - 1:10 ora_psp0_PPRUN
oracle 930042 1 0 Mar 03 - 9:19 ora_lgwr_PPRUN
oracle 934012 1 2 Mar 03 - 5:55 ora_dbw0_PPRUN
oracle 938124 1 0 Mar 03 - 0:16 ora_mman_PPRUN
oracle 942294 1 0 Mar 03 - 0:00 ora_reco_PPRUN
oracle 954520 4636838 0 13:43:57 pts/6 0:00 grep PPRUN
oracle 962780 1 0 Mar 03 - 0:02 ora_q000_PPRUN
oracle 970974 1 0 Mar 03 - 0:04 ora_q001_PPRUN
oracle 4501582 1 0 13:38:20 - 0:49 ora_s000_PPRUN
```

Remarque: la zone mémoire utilisateur standard (PGA), dans ce type d'architecture, ne contient qu'une pile et les informations vraiment spécifiques du process user. Les autres informations normalement contenues en PGA sont déportées en SGA.

Remarque : en architecture serveur partagée, la taille de la SGA doit être revue sensiblement à la hausse du fait de l'incorporation des infos des PGAs utilisateur.

Le paramétrage

SQL> show parameter shared_server

NAME TYPE VALUE
---------- ------ -----
max_shared_servers integer 40
shared_server_sessions integer
shared_servers integer 10

SHARED_SERVERS
———————-
SHARED_SERVERS spécifie le nb de process lancés systématiquement au démarrage de l'instance. En cas de montée en charge, ce nombre pourra augmenter, mais ne descendra jamais en dessous de cette limite dans le cas inverse.

MAX_SHARED_SERVERS
———————-
MAX_SHARED_SERVERS spécifie le nb maximum de process qui peuvent tourner simultanément.
Il est compris entre SHARED_SERVER et PROCESSES.
Imposer cette limite permet de laisser des 'slots' libres pour d'autre process comme les proces dédiés par exemple.
si MAX_SHARED_SERVERS n'est pas spécifié, un SHARED SERVER sera déclenché tant que le nb de 'slots' de libre est > a 1/8 max(nb max process)
ou 2 si PROCESSES < 24.

Gestion des Users et des Roles - 1

Gestion des Users et des rôles 1

A l'aide la console EM
- créer un utilisateur USERn
avec les caractéristiques suivantes :
Espace disque logique par défaut TBSn, TBS temporaire TEMP, quota sur TBSn 50K
et les privilèges minimum pour pouvoir créer des tables
- vérifier la possibilité de création d'une table et les caractéristiques son implantation
en créant une table sous i*SQLPLUS
: CREATE TABLE nom_table (nom_col type , nom_col type, …)

Pour la suite des TPs donner le rôle DBA a USERn

Tablespace et stockage

Evolution des TBS
———————
Visualisez les infos sur le TBS 'USERS'
Quel taille fait-il? Quel est son pourcentage de remplissage actuel ?
Quelles sont ses règles d'extension ? (méthode auto ou non, taille de l'extension, limite…)

onglet 'Administration', rubrique 'Stockage', lien 'espace disque logique',
choisissez 'USERS' puis 'Visualiser'.
Cliquez ensuite sur le lien 'users01.dbf' (le 1er et unique fichier qui comose ce TBS)
Vous obtenez des informations de ce style :

Nom /oracle/DBADE/users01.dbf
Espace disque logique USERS
Statut En ligne
Taille de fichier (Ko) 5120
AutoExtend Oui
Incrément 1280Ko
Taille de fichier maximale 32767Mo

Stockage des segments
————————
- comment sont stockés physiquement les objets du tablespace USERS,
combien d'extents et de quelle taille ?

Même chemin que précédemment.
Quand vous êtes sur la page 'Visualiser Espace disque logique : USERS'
Exécutez 'Afficher le contenu des espaces disque logiques'
Vous obtenez des informations de ce style :

Nom du segment Type Taille Ensembles de blocs contigus (les extents!)
————— —- —— ——————————
SCOTT.PK_DEPT INDEX 64 1
SCOTT.DEPT TABLE 64 1
SCOTT.EMP TABLE 64 1
SCOTT.PK_EMP INDEX 64 1
SCOTT.BONUS TABLE 64 1
SCOTT.SALGRADE TABLE 64 1

Création (pseudo) manuelle de TBS
————————————

SOUS i*SPLPLUS !!!! :
créez un TBS 'TBSn' permanent, géré localement,
de 50K, auto extensible jusqu'a 150K par paquet de 50K.
Vous pouvez le cas échéant vous aider de la console pour afficher l'ordre SQL qui va bien…

Sur la console;
… Espace disque logique, 'créez', entrez le nom, ajoutez un fichier de données
puis 'Afficher le SQL'

CREATE SMALLFILE TABLESPACE "TBS1"
DATAFILE '/oracle/DBADE/tbs11.dbf'
SIZE 50K AUTOEXTEND ON NEXT 50K MAXSIZE 150K LOGGING EXTENT MANAGEMENT
LOCAL
SEGMENT SPACE MANAGEMENT AUTO

Corrigé - flashback table

Récupération de table grâce à la recovery Area

————————————————————
Vérifier que la poubelle est active au niveau instance
et l'activer si nécessaire
SQL> SHOW PARAMETER recycle
SQL> ALTER SYSTEM SET recyclebin= ON;
————————————————————————
Créer une table T1 avec une colonne CHAR, insérer une ou 2 lignes
Créer deux tables T2 et T3 avec une colonne INTEGER
Supprimer la table T1 et T2 logiquement
Supprimer la table T3 physiquement

```
create table t1 (c char(10));
create table t2 (n integer);
create table t3 (n integer);
drop table t1;
drop table t2;
drop table t3 purge;
```

————————————-
Vérifier le contenu de la poubelle
SQL> SHOW recyclebin

————————
Récuperer la table T1
SQL> FLASHBACK TABLE T2 TO BEFORE DROP;

————————————-
Purger si nécessaire la corbeille
SQL> PURGE recyclebin;

CHAPITRE K -liens_utiles

Sites Oracle intéressants

Les tutoriels sur Oracle et le web de Didier Deleglise
http://didier.deleglise.free.fr/

Les forums de Didier Deleglise
http://didier.deleglise.free.fr/w-agora/index.php?site=oracle

Le blog technique Oracle de Didier Deleglise
http://didier.deleglise.free.fr/w-agora/index.php?site=oracle

Doc sur les packages PLSQL 10g
http://www.psoug.org//reference/builtin_packages.html

Doc sur les fonctions SQL 10g
http://www.psoug.org//reference/builtin_functions.html

Le site de Gilles briard
http://gbriard.club.fr/index.php

LES FAQs non officiels (beaucoup d'infos)
http://orafaq.com/

infos techniques diverses
http://www.dbapool.com/

20 tutoriels 10g intéressants : The Top 20 Features for DBAs
http://www.oracle.com/technology/pub/articles/10gdba/index.html

Les tutoriels DBA 10g gratuits de Exforsys
http://www.exforsys.com/content/category/17/261/343/

La sécurite Oracle par Pete Finnigan
http://www.petefinnigan.com/orasec.htm

Le labo Oracle de Supînfo
http://www.labo-oracle.com/

Oracle SQL et standard SQL par Stanford EDU
http://infolab.stanford.edu/~ullman/fcdb/oracle/or-nonstandard.html

SQL Oracle et SQL des autres SGBDs
http://sqlzoo.net/

SQL standard et validateur SQL, par MIMER
http://developer.mimer.com/validator/index.htm

et plus officiel :

Oracle Corp
http://www.oracle.com/index.html

Metalink : le support et les patchs…
https://metalink.oracle.com/

OTN : le reseau technique Oracle
http://www.oracle.com/technology/index.html

un lien direct sur la doc
http://tahiti.oracle.com/