

Administration Oracle

Guide d'utilisation

Michaël Genin

Sommaire

Introduction à l'architecture Oracle.....	5
Structure Logique	6
Espace disque logique (tablespace)	6
Data Blocks	7
Extents	7
Segments	7
Schema	7
Structure Physique	8
Datafiles (.dbf files):	8
Redo Log Files (.rdo & .arc):	8
Control Files (.ctl):	9
Administration basique de base de données	9
Session administrateur et remarques préliminaires	9
Définition du rôle d'administrateur d'un serveur Oracle.....	9
Privilèges de l'administrateur.....	10
Les USERNAMES de l'administrateur.	10
Le rôle DBA	10
Les utilitaires Administrateur	11
Soumettre des commandes et des commandes SQL à la base	11
A propos de SQL*Plus.....	12
Création et gestion d'une base de données.....	12
A propos de la création d'une base de données	12
Remarques préliminaires avant l'installation d'une base de données	12
Pré-requis avant la création de la base de données	14
Création d'une base de données via l'assistant de création de base de données DBCA.....	14
Création d'une base de données sans le DBCA, avec la commande CREATE DATABASE	27

Lancement et arrêt d'une base de données	33
Démarrage d'une instance et d'une base de données.....	34
Changer la disponibilité d'une base de données	36
Arrêter une instance et base de données	37
A propos de l'environnement Réseau	38
Instance Oracle.....	40
Présentation générale	40
System Global Area (SGA)	41
Shared Pool.....	41
Database Buffer Cache	42
Redo Log Buffer	42
Program Global Area (PGA)	43
Processus d'arrière plan	43
DBWR (Database Writer).....	43
LGWR (Log Writer).....	44
PMON (Process Monitor)	44
SMON (System Monitor)	44
Création d'une instance Oracle	44
Gestion des structures de stockage de la base de données	45
Fichiers de contrôle	45
Gestion des fichiers de contrôles	46
Online Redo Log Files	56
Archived Redo Log Files.....	57
Gestion des Redo Log	58
Rollback Segments.....	67
Fichiers de données.....	67
Espace disques logiques (Tablespace).....	68

Gestion des espaces disque logique.....	70
Données UNDO.....	74
Gestion des données Undo	75
GESTION DES UTILISATEURS.....	78
Informations sur les utilisateurs.....	78
Création d'utilisateur : CREATE USER.....	79
Modification d'un utilisateur : ALTER USER	82
Suppression d'un utilisateur : commande DROP USER.....	83
GESTION DES RÔLES ET PRIVILEGES	85
Assignation de privilèges système.....	85
Retrait de privilèges système	86
Assignation de privilèges objets	87
Retrait de privilèges objet	88
Création de rôles	89
Informations relatives aux rôles.....	91
Modification de rôle	92
Suppression d'un rôle.....	94
Liste des privilèges système	95
Liste des privilèges objet	101
Sauvegarde et restauration de base de données.....	103
LES UTILITAIRES EXPORT ET IMPORT	103
Remarque préliminaire.....	103
Les différents modes d'export et d'import.....	103
Les principaux paramètres de contrôle.....	105
Méthode pour exporter et importer	108
SAUVEGARDE A FROID	109
Introduction.....	109

Sauvegarde	109
SAUVEGARDE A CHAUD.....	110
Introduction.....	110
Sauvegarde	111
RESTAURATION DES DONNEES	113
Restauration des fichiers de données	113
Restauration totale de la base de données.....	113
Restauration complète à partir de la sauvegarde à froid.....	113

Introduction à l'architecture Oracle

Cette partie va s'attacher à décrire les différentes entités qui composent Oracle. Cela va nous permettre d'acquérir des notions introductives afin de mieux appréhender les parties suivantes. Par ailleurs, une description plus détaillée des différentes entités est disponible dans ce cours.

Oracle Database est composé des éléments suivants :

- Le logiciel Oracle qui est installé sur l'ordinateur ;
- La base de données qui est une collection de données physiques sur un ou plusieurs disques durs. Une base de données est composée de données utilisateurs, de métadonnées ainsi que des structures de contrôle. Les métadonnées, ou encore données à propos des données, sont des ensembles d'informations qui permettent au logiciel Oracle de gérer les données des utilisateurs. Un exemple de métadonnée est le dictionnaire de données. Les structures de contrôles, comme les fichiers de contrôle ou

encore les fichiers Redo Log) assure l'intégrité, la disponibilité et capacité de restauration des données des utilisateurs.

- L'instance Oracle qui est composée des éléments suivants :
 - o Les processus d'arrière plan qui sont les processus ou threads du système d'exploitation qui réalisent les travaux d'accès, de stockage, de suivi et restauration des données utilisateurs, métadonnées et les fichiers de contrôle associés à la base de données.
 - o Les espaces de la mémoire partagée (shared memory areas) qui sont utilisés par les processus d'arrière plan.
- Les processus serveur qui réalisent les actions sur la base au nom des utilisateurs connectés et des applications. Par exemple, les processus serveur analysent et réalisent les requêtes SQL, les récupèrent et les retournent à l'utilisateur ou à l'application.
- Oracle Net qui est une couche logicielle qui permet aux applications clientes de communiquer avec une base de données Oracle au travers d'un réseau. L'Oracle Net listener est processus qui « écoute » les requêtes de connexion provenant du réseau.

L'architecture Oracle peut être décrite en termes de structures logique et physique. L'avantage de séparer ces deux types de structures réside dans le fait que la structure physique de stockage peut être modifiée sans pour autant affecter la structure logique.

Structure Logique

La structure logique consiste en deux éléments :

- Espace disque logique (tablespace)
- Schéma

Espace disque logique (tablespace)

Une base de données Oracle est constituée d'une ou plusieurs portions logiques appelées Espace disque logique. C'est un groupement logique de données liées ensemble.

Un administrateur de base de données peut utiliser les espaces disque logique pour effectuer les actions suivantes :

- Contrôler l'allocation d'espace disque pour les données d'une base ;
- Assigner des quotas d'espace disque pour les bases de données des utilisateurs ;
- Réaliser des sauvegardes (backup) partielles de base de données ou des opérations de restauration (recovery).

Chaque base de données a au minimum un espace disque logique nommé SYSTEM. Pendant la création d'une base de données, Oracle crée automatiquement ce tablespace. Cependant, il est important de préciser que bien qu'une base de données de petite taille peut être contenue dans le tablespace SYSTEM, il est recommandé de créer un espace disque logique séparé pour les données des utilisateurs.

Oracle utilise le tablespace SYSTEM pour stocker des informations comme le dictionnaire de données. Celui-ci contient les métadonnées (ou encore les données à propos des données). Celles-ci contiennent des informations telles que les permissions d'accès aux tables, des informations concernant les clés, etc...

Les données d'une base de données sont stockées dans un type de fichiers appelé Datafiles. Chaque espace disque logique est une collection d'une ou plusieurs Datafiles. Chaque Datafile peut être décomposé en DataBlocks, Extents et Segments.

Data Blocks

Au meilleur niveau de détails, les données d'une base de données Oracle sont stockées dans des datablocks (aussi appelés blocs logiques, Oracle blocs ou pages). Une base de données Oracle utilise et alloue de l'espace libre au sein des datablocks.

Extents

Le niveau suivant d'espace logique d'une base de données est appelé Extent. Ce dernier est un nombre spécifique de datablocks contigus qui sont alloués pour le stockage d'un type spécifique d'information.

Segments

Le niveau logique supérieur est un segment. Un segment est un ensemble d'extents qui ont été alloués pour un type spécifique de structure de données et tous stockés dans la même Espace disque logique (tablespace). Par exemple, chaque table est stockée dans son propre segment de table tandis que chaque index est stocké dans son propre segment d'index. Oracle alloue de l'espace mémoire pour les extents dans un segment. Par ailleurs, lorsque les extents d'un segment sont pleins, Oracle alloue un autre extent pour ce segment.

Une base de données Oracle peut utiliser quatre types de segment :

- Data segment : Stocke les données des utilisateurs dans la base
- Index segment : Stocke les indexes.
- Rollback segment : Stocke les informations concernant les rollback.
- Temporary segment : Créé lorsqu'une requête SQL a besoin d'un espace temporaire de travail. Ce type de segment est détruit lorsque la requête SQL est terminée. Par exemple, ce type de segment est utilisé pour des opérations de tri.

Schema

Un schéma est un groupement logique d'objets tels que :

- Tables

- Clusters
- Indexes
- Vues
- Procédures stockées
- Triggers
- Séquences

Un schéma définit aussi le niveau d'accès des utilisateurs.

Structure Physique

La couche physique de la base de données est constituée de trois types de fichiers :

1. Une ou plusieurs Datafiles
2. Deux ou plusieurs redo log files
3. Une ou plusieurs control files

Datafiles (.dbf files):

Les datafiles stockent les informations continues dans la base de données. Par exemple, une base de données peut avoir très peu de datafiles et d'autre en avoir des centaines. Les informations relatives à une table peuvent utiliser plusieurs datafile ou au contraire plusieurs tables peuvent partager un set de datafiles. Le fait de disséminer les tablespaces sur plusieurs datafiles peut s'avérer efficace en termes de performance. Le nombre de datafiles qui sont configurable est limité par le paramètre Oracle MAXDATAFILES.

Redo Log Files (.rdo & .arc):

Oracle enregistre des logs de toutes les transactions réalisées au sein de la base de données. Ces transactions sont enregistrées dans des fichiers appelés Online Redo Log Files (Redo Logs). L'objectif principal de ces fichiers est de contenir des informations de restaurations dans le cas d'une panne du système. Un Redo log stocke un journal de tous les changements effectués au sein de la base de données. Les redo log files doivent fonctionner correctement et être protégées contre les pannes hardware. En effet, si une seule information relative au fichier redo log, ou même le fichier en lui-même est perdu, le système ne peut être restauré.

Lorsqu'une transaction est effectuée dans la base de données, celle-ci est entrée dans les buffers du redo log tandis que les datablocks affectés par cette transaction ne sont pas directement modifiés sur le disque. Dans une base de données Oracle, il y a au moins trois Red log Files.

Oracle écrit dans les Redo Log dans un ordre cyclique. En effet, après que le premier fichier Red Log ait été rempli, le système écrit dans le second et ainsi de suite jusqu'au moment où tous les fichiers aient été remplis. Après cette étape, Oracle commence à écrire à nouveau des informations

d'une nouvelle transaction sur le premier fichier Red Log, écrasant les informations qu'il contenait. A noter que la base de données est exécutée en mode ARCHIVELOG, cette dernière effectue des copies des fichiers redo log avant la réécriture.

Control Files (.ctl):

Les fichiers de contrôle enregistrent les informations de contrôle de tous les fichiers au sein de la base. Ces fichiers maintiennent la consistance interne et guident l'opération de restauration. Ils contiennent des informations utilisées pour le démarrage d'une instance comme l'emplacement des datafiles et des fichiers redo log.

Il est important de préciser que les fichiers de contrôles doivent être protégés. A cet effet, Oracle propose un mécanisme de sauvegardes multiples des fichiers de contrôle. Ces multiples sauvegardes sont stockées sur différents disques afin de minimiser les dommages potentiels dus à une panne de disque. Les noms des fichiers de contrôles d'une base de données sont spécifiés via le paramètre d'initialisation CONTROL_FILES.

Administration basique de base de données

Session administrateur et remarques préliminaires

Définition du rôle d'administrateur d'un serveur Oracle

De par le fait que Système de Gestion de Base de Données Relationnel (SGBDR) Oracle peut être relativement important et comporter un nombre important d'utilisateur, quelqu'un ou un groupe de personnes se doivent de gérer le système. L'administrateur de base de données est cette personne (DataBase Administrator (DBA) en anglais).

Ainsi, en tant qu'administrateur Oracle, les responsabilités peuvent être les suivantes :

- Installation du serveur Oracle et des outils d'administration
- Création de la structure de stockage de la base de données (tablespace)
- Création d'objets primaires (tables, index...)
- Modification de la structure de la base de données, si nécessaire, à partir des informations données par les développeurs
- Gestion des utilisateurs et maintenance de la sécurité du système
- Gestion des privilèges
- Visualisation et optimisation des performances du système
- Gestion et planification des sauvegardes (back up) et restauration (recovery) des informations de la base de données

Privilèges de l'administrateur

Afin de réaliser les tâches administratives dans Oracle, l'administrateur doit disposer de privilèges à la fois au sein de la base de données et dans le système d'exploitation du serveur sur lequel la base de données est installée.

En effet, afin de réaliser de nombreuses tâches administratives pour une base de données, l'administrateur doit pouvoir exécuter des commandes système. Selon le système d'exploitation sur lequel Oracle est installé, le DBA a besoin d'un compte ou d'un identifiant afin d'obtenir un accès au système d'exploitation. Dans ce cas, le compte du système d'exploitation requiert plus de privilèges systèmes ou droit d'accès qu'un utilisateur standard de la base de données (par exemple, pour installer Oracle).

Les USERNAMES de l'administrateur.

Deux comptes administrateurs sont automatiquement créés avec la base de données :

- **SYS** (mot de passe initial : `CHANGE_ON_INSTALL`)
Toutes les tables et vues pour le dictionnaire de données de la base sont stockées dans le schéma SYS. Ces tables et vues sont essentielles pour les opérations d'Oracle. Aussi, dans le but de maintenir l'intégrité des données, elles ne doivent en aucun cas être modifiées par un utilisateur ou même un administrateur. Par ailleurs, il est important de préciser que personne ne devrait créer de tables dans le schéma SYS.
- **SYSTEM** (mot de passe initial : `orcl`)
Ce compte est sensiblement identique au précédent à l'exception de tables additionnelles relatives à des informations administratives ou encore de tables et vues internes correspondantes aux outils d'administration.

Note : la plupart des utilisateurs ne doivent pas être capables de se connecter en utilisant les comptes SYS ou SYSTEM. Par ailleurs, afin de prévenir des accès non désirés à ces deux comptes, il est important de changer les mots de passe de ces deux comptes.

Le rôle DBA

Un rôle prédéfini nommé « DBA » est automatiquement créé avec chaque base de données Oracle. Ce rôle contient l'ensemble des privilèges système de la base. Aussi, ce dernier est puissant et se doit d'être attribué uniquement aux administrateurs de la base.

Les utilitaires Administrateur

- **Oracle Universal Manager (OUI)**

C'est un utilitaire qui permet d'installer le logiciel Oracle. Par ailleurs, il peut automatiquement lancer l'assistant de configuration de base de données (DBCA) afin d'installer une base.

- **Oracle Database Configuration Assistant (DBCA)**

L'assistant de configuration de base de données est un utilitaire qui permet de créer, configurer ou encore modifier une base de données à partir des modèles disponibles dans Oracle. Par ailleurs, il est possible de créer son propre modèle.

- **Database Upgrade Assistant**

Cet outil permet de guider l'administrateur au travers de l'extension d'une base de données existante vers un nouvel environnement Oracle (par exemple une nouvelle version d'Oracle).

- **Net Configuration Assistant**

L'assistant de configuration réseau (NETCA) est un utilitaire qui permet de configurer les listeners et les naming methods, qui sont les composants essentiels du réseau d'Oracle Database.

- **Oracle Enterprise Manager Database Control**

Cette interface web constitue le principal outil de gestion d'une base de données. Après avoir installé le logiciel Oracle, créé une base de données et configuré le réseau, il est possible d'utiliser Database Control pour gérer la base de données. Cet outil propose également une interface permettant suivre les performances de la base de données mais également des utilitaires tels que SQL*Loader (récupération de données pour une base à partir de fichiers texte ou fichier C) ou encore le Recovery Manager (RMAN).

Soumettre des commandes et des commandes SQL à la base

Le premier moyen de communication avec une base de données Oracle est la soumission de requêtes SQL. Une base de données Oracle supporte aussi une « super collection » SQL, qui incluse les commandes pour le démarrage et l'arrêt d'une base, la modification de la configuration de la base etc...

Il y a deux moyens de soumettre ces requêtes SQL et commandes à une base de données Oracle :

- En utilisant le client SQL*Plus ;

- En utilisant l'interface utilisateur d'Oracle Enterprise Manager. Ce dernier propose une interface intuitive afin d'administrer la base de données, et il soumet les commandes et requêtes SQL en arrière plan.

A propos de SQL*Plus

Sql*Plus est l'interface primaire en lignes de commandes pour une base de données Oracle. Vous pouvez l'utiliser afin de démarrer ou arrêter une base, définir les paramètres d'initialisation, créer et gérer des utilisateurs, créer et modifier des objets de la base de données (comme les tables ou encore les indexes)

Création et gestion d'une base de données

A propos de la création d'une base de données

Sous Oracle, il est possible de créer une base de données par le biais d'un outil graphique ou par commandes SQL. Dans tous les cas, vous créez une base de données pendant l'installation d'Oracle. Cependant, vous pouvez aussi en créer une après installation. Les raisons de cette création peuvent être les suivantes :

- Vous avez utilisé *Oracle Universal Installer* (OUI) uniquement pour l'installation et vous n'avez donc pas créé de base de données ;
- Vous voulez créer une autre base de données (et donc une autre instance de base) sur le même ordinateur.
- Vous voulez faire une copie d'une base de données existante (clone).

Les méthodes de création d'une base de données sont les suivantes :

- Avec l'interface graphique : Assistant de configuration de base de données (*Database Configuration Assistant* : DBCA)
- Avec la commande SQL CREATE DATABASE ;

Remarques préliminaires avant l'installation d'une base de données

Avant de passer à l'étape de création à proprement dite, il est important de réfléchir à de multiples critères relatifs à la création. En effet, vous devez réfléchir par exemple au nombre de fichiers de données (datafiles) ou encore le nombre d'instances qui peuvent accéder à celle-ci.

Note : Vous pouvez également créer une base de données dans le but d'effacer les informations d'une base de données existante par le biais de la création d'une base ayant le même nom et structure physique.

Aussi certaines réflexions préalables sont recommandées avant la création d'une base de données (les notions citées infra seront explicitées dans ce guide) :

- Prévoir les tables et les indexes de la base de données et ainsi estimer la quantité d'espace mémoire suffisante pour accueillir ces objets ;
- Prévoir l'architecture des différents systèmes d'exploitation que la base de données va comprendre. Une distribution appropriée des fichiers peut améliorer grandement les performances d'une base de données en divisant les entrées/sorties pendant l'accès aux données. Vous pouvez répartir les entrées/sorties de différentes façons quand vous installez l'application Oracle ou pendant la création d'une base de données. Par exemple, vous pouvez placer les fichiers Redo Log sur des disques différents (pour minimiser les problèmes liés au crash d'une panne d'un disque) ou encore contrôler la densité de données (nombre de lignes d'un bloc de données (datablock)).
- Définir le nom global de la base de données qui est le nom et la localisation de la base dans le réseau informatique. Vous pouvez créer le nom global en définissant les paramètres d'initialisation DB_NAME et DB_DOMAIN.
- Sélectionner le Character Set de la base de données. Toutes les données de nature alphanumériques, incluant les données dans le dictionnaire de données, sont stockées dans le Character Set de la base de données. Ce dernier doit être spécifié lors de la création d'une base de données.

Si les clients de la base utilisent des character sets différents, il est important de choisir un superset qui englobe l'ensemble des sets des clients. Sinon, les conversions de données peuvent induire d'éventuelles pertes de données mais aussi peuvent avoir des effets néfastes sur la performance de la base.

Par exemple, le character set WE8ISO8850P1 est correspond à l'ensemble des pays de l'Europe de l'ouest. Par conséquent le client de la base peut avoir un character set correspondant à langue française et un autre client avec un character set correspondant à la langue anglaise, il n'y aura pas de problème de conversion. En effet, le character set de la base est un superset qui englobe ceux des clients.

- Définir la taille standard de la taille d'un bloc de données de la base. Celle-ci est spécifiée à la création de la base de données par le paramètre d'initialisation DB_BLOCK_SIZE et ne

peut être changée après la création. Par exemple, le tablespace SYSTEM et la plupart des autres espaces disque logique utilisent la taille de bloc de données standard.

- Prévoir un tablespace par défaut pour les utilisateurs de la base de données afin d'éviter que ces derniers ne fassent pas, par exemple, de création d'objets dans le tablespace SYSTEM.
- Prévoir d'utiliser un tablespace UNDO afin de gérer les données UNDO.
- Elaborer une stratégie de sauvegarde (backup) et de restauration (recovery) afin de protéger la base de données d'éventuelles pannes systèmes.

Pré-requis avant la création de la base de données

Avant la création d'une base de données, certaines conditions sont pré requises :

- L'environnement Oracle doit être installé. Cela inclus le fait que les différentes variables d'environnement aient été définies ainsi que la création d'une arborescence de répertoires en vue d'accueillir les données des applications et de la base de données.
- Une mémoire vive assez importante afin de lancer une instance de base de données.
- Un espace mémoire sur le disque suffisant pour accueillir la base de données que nous voulons créer.

Création d'une base de données via l'assistant de création de base de données DBCA.

Cette partie va s'attacher à décrire les différentes étapes de création d'une base de données via le DBCA et en parallèle nous allons créer une base de données simple comme exemple.

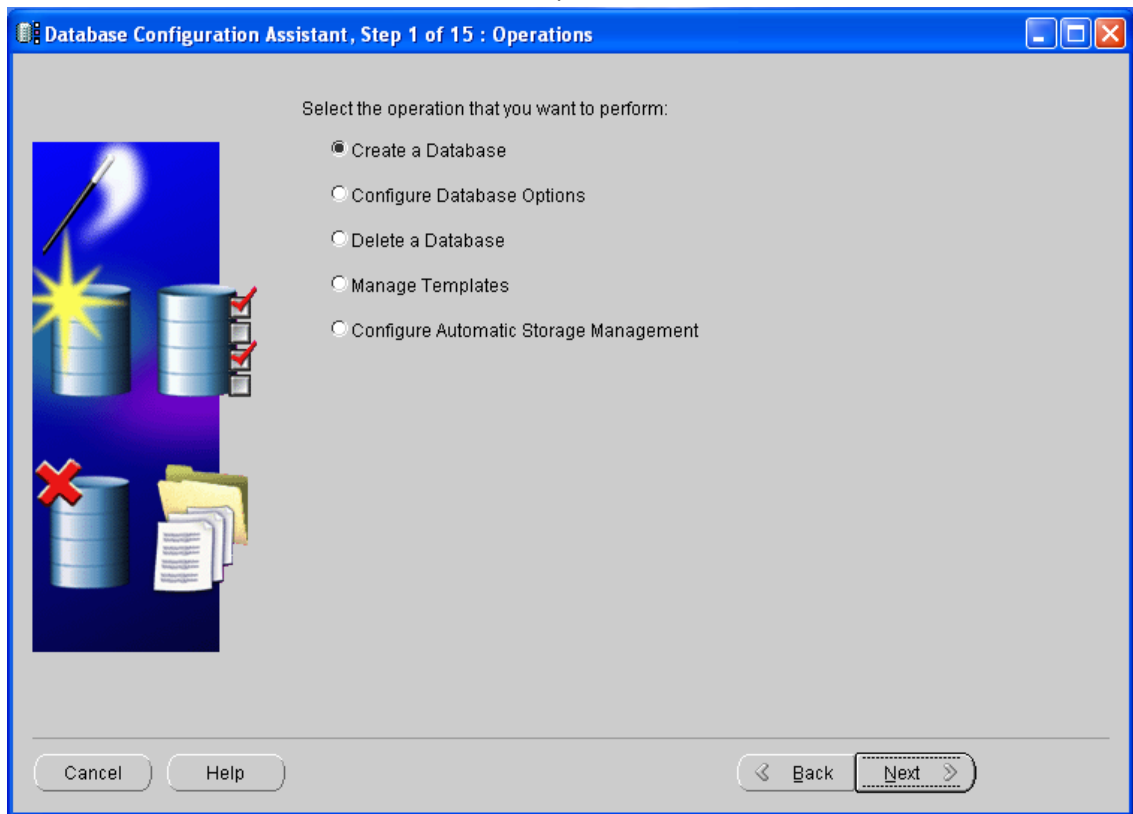
Démarrage du DBCA :

- Vérifiez que vous êtes connectés sur l'ordinateur en tant qu'administrateur, afin d'avoir les mêmes droits que lors de l'installation d'Oracle.
- Dans une console (Terminal sous Linux ou console MS-Dos sous Windows) tapez la commande dbca.
- Cliquez sur Next afin de passer la fenêtre d'accueil.

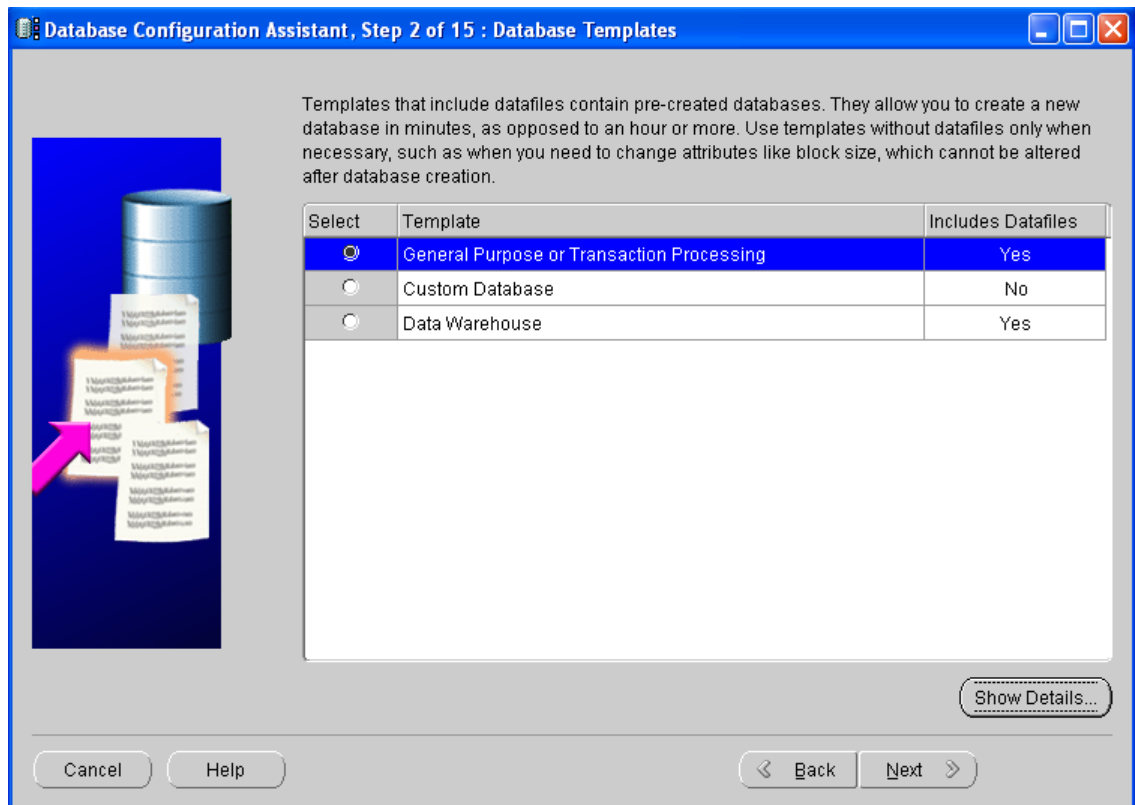
Création de la base de données

▪ Etape 1

Sélectionnez CREATE DATABASE et cliquez sur Next.



▪ Etape 2 : Modèle de base de données



Cette fenêtre vous permet de sélectionner le type de base de données que vous voulez créer. Vous pouvez créer une base de données à partir d'un modèle proposé par Oracle ou à partir de votre propre modèle. Les modèles contiennent les configurations optimisées pour un type particulier de charge de travail. Oracle propose des modèles pour deux types de charge de travail : des bases de données courantes et des entrepôts de données.

Vous pouvez cliquer sur Shows Details afin de voir la configuration pour chaque type de base de données.

Si vous n'êtes pas sûr du type que vous devez choisir, sélectionnez General Purpose or Transaction Processing.

Dans notre exemple, nous allons choisir ce type.

Cliquez sur Next.

Note : Pour des environnements plus complexes, vous choisir l'option Custom DataBase. Cette option n'utilise pas de modèles, ce qui implique que la création de votre base de données va sûrement prendre beaucoup plus de temps.

▪ Etape 3 : Identification de la base de données

Dans le champ **Global Database Name**, entrez le nom de la base de données. Dans le champ **SID**, entrez l'identifiant de l'instance de la base de données. Par défaut, le SID est le nom de la base de données et identifie uniquement l'instance qui lance la base de données.

Database Configuration Assistant, Step 3 of 15 : Database Identification

An Oracle database is uniquely identified by a Global Database Name, typically of the form "name.domain".

Global Database Name:

A database is referenced by at least one Oracle instance which is uniquely identified from any other instance on this computer by an Oracle System Identifier (SID).

SID:

Cancel Help < Back Next >

▪ **Etape 4 : options de gestion**

Database Configuration Assistant, Step 4 of 15 : Management Options

☒ **Configure Enterprise Manager**

☐ Register with Grid Control for centralized management

Management Service:

☒ **Configure Database Control for local management**

☐ **Enable Alert Notifications**

Outgoing Mail (SMTP) Server:

Recipient Email Address:

☐ **Enable Daily Disk Backup to Recovery Area**

Backup Start Time: AM ☐ PM

OS Username:

OS Password:

Cancel Help < Back Next >

Cette étape permet de configurer la base de données afin qu'elle puisse être gérée avec l'outil Oracle Enterprise Manager précité.

Aussi, sélectionnez Configure Enterprise Manager afin de l'utiliser. Ensuite vous pouvez sélectionner les options suivantes :

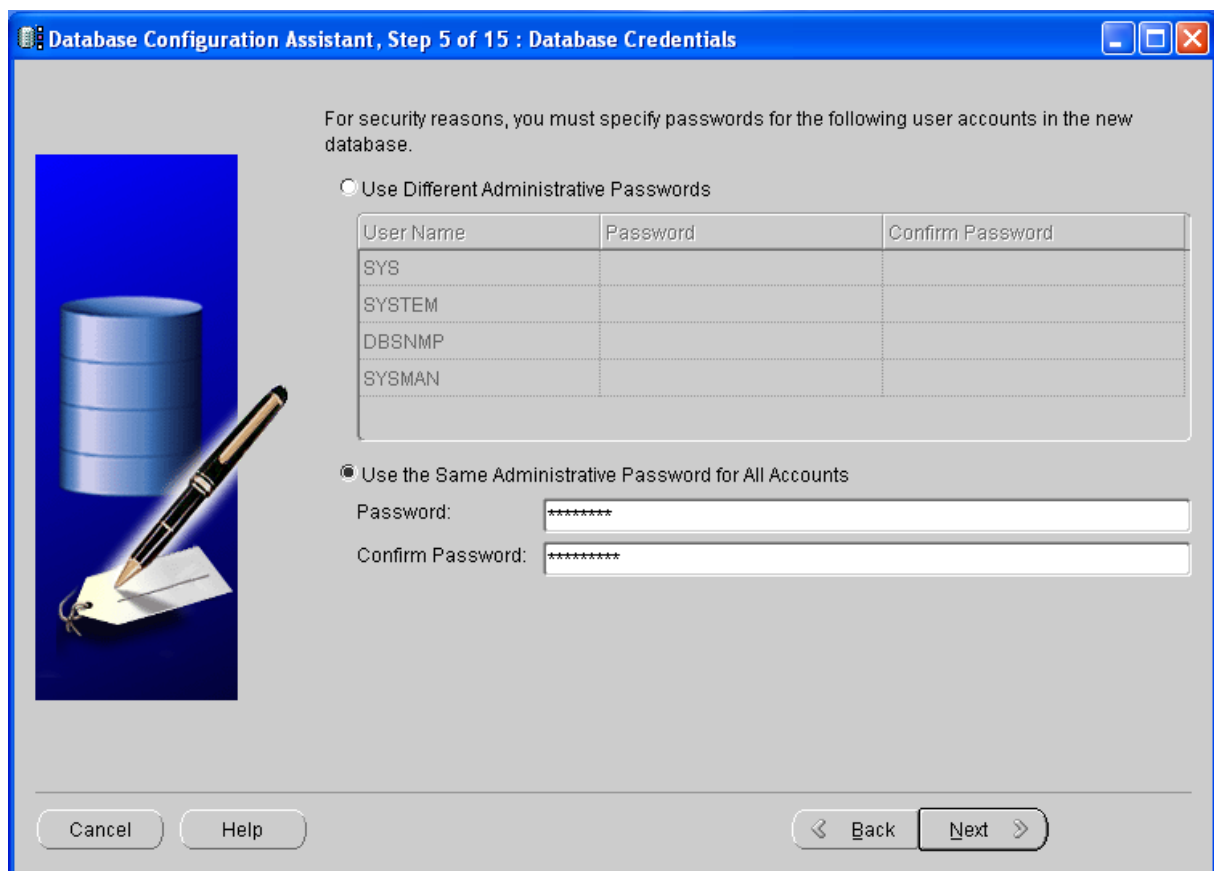
- Si l'agent de gestion Oracle (Oracle Management Agent) est installé sur votre machine, vous pouvez choisir une gestion centralisée en sélectionnant l'option *Register with Grid Control for centralized management* et ensuite sélectionner le service de gestion.
- Dans le but de gérer la base de données de façon locale (ce qui nous intéresse ici), sélectionnez l'option : *Configure Database Control for local management*. De plus, des options vous permettent de configurer des notifications par mail relatives alertes concernant la base de données mais aussi une sauvegarde journalière de la base.

Dans le cadre de l'exemple, nous avons choisir d'utiliser Enterprise Manager et de gérer la base localement (c.f. image *supra*).

▪ Etape 5 : Certificats de la base de données

Dans cette fenêtre, spécifiez les mots de passe relatifs aux comptes administrateurs tels que SYS ou SYSTEM.

Pour les besoin de notre exemple, choisissez un mot de passe commun à l'ensemble des compte administrateurs.



Database Configuration Assistant, Step 5 of 15 : Database Credentials

For security reasons, you must specify passwords for the following user accounts in the new database.

☐ Use Different Administrative Passwords

User Name	Password	Confirm Password
SYS		
SYSTEM		
DBSNMP		
SYSMAN		

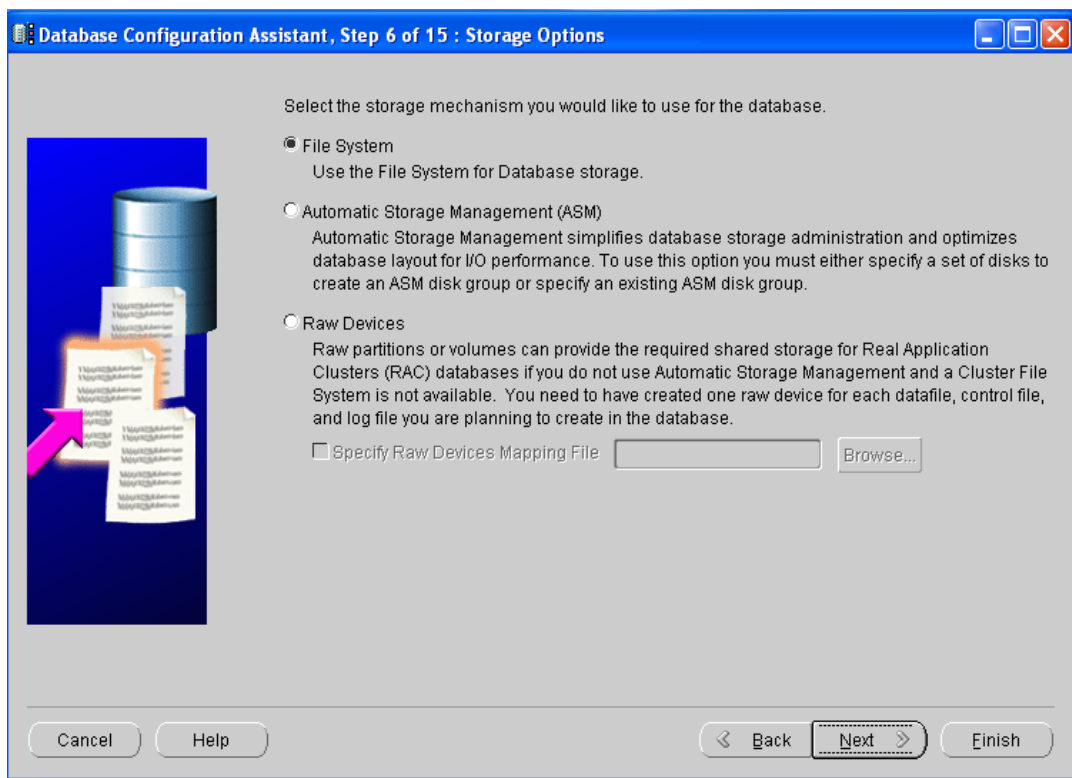
☒ Use the Same Administrative Password for All Accounts

Password:

Confirm Password:

Cancel Help < Back Next >

▪ Etape 6 : Options de stockage de la base de données



Une base de données inclut plusieurs fichiers qui stockent les données des utilisateurs, les métadonnées de la base de données et les informations requises pour les restaurations en cas de panne du système. En tant qu'administrateur, vous devez choisir quel type de sous-système de stockage à utiliser pour ces fichiers. Vous sélectionnez les options suivantes :

- **File System** : cette option par défaut crée des fichiers d'une base de données qui sont gérés par le système de fichier du système d'exploitation sur lequel l'environnement Oracle est installé. Vous pouvez également spécifier le chemin du répertoire dans lequel les fichiers de données de la base doivent être stockés.

Note : si vous n'êtes pas certain de l'option que vous devez choisir, sélectionnez cette option (par défaut).

- **Automatic Storage Management** : cette option permet à l'administrateur de définir le nombre de stockage, appelés groupes de disques dans lesquels l'environnement Oracle gère automatiquement les fichiers de base de données ainsi que leurs noms. Pour les environnements avec un nombre importants de disques durs, cette option simplifie l'administration de la base et maximise les performances.
- **Raw Devices** : cette option permet de gérer des systèmes de stockage qui se trouvent pas sur le système d'exploitation sur lequel est installé l'environnement Oracle en fournissant de l'espace disque appelé Raw devices. (à voir)

Pour les besoins de notre exemple, nous allons choisir le système de stockage par défaut.

▪ Etape 7 : Emplacement des fichiers de la base de données

Cette fenêtre permet de spécifier le répertoire racine du logiciel Oracle ainsi que chemin du répertoire dans lequel nous souhaitons créer les fichiers de la base de données. Vous avez le choix entre différentes options :

- **Use Database File from Template** : cette option permet au DBCA d'utiliser les informations concernant le répertoire racine qui sont spécifiées dans le modèle. Vous pouvez ultérieurement réaliser des modifications sur les noms de fichiers de données de la base ainsi que les emplacements.
- **Use Common Location for All Database Files** : cette option demande que vous spécifiez un nouveau répertoire pour Oracle home. Ainsi, tous les fichiers de la base de données seront créés dans ce répertoire. Vous pouvez également modifier ces informations ultérieurement.
- **Use Oracle Managed Files** : cette option demande à Oracle Database de gérer l'ensemble des fichiers relatifs à une base de données Oracle. Il est demandé de spécifier un emplacement par défaut, appelé un espace de base de données (database area), pour l'ensemble des fichiers. Par la suite, Oracle crée et efface automatiquement les fichiers à cet emplacement, si nécessaire. Vous pouvez également choisir de créer de multiples copies des fichiers Redo Log et Control Files en sélectionnant Multiplex Redo Logs and Control Files.

En sélectionnant cette option, vous déléguez la gestion complète des fichiers de la base de données à celle-ci. Vous n'avez plus besoin de spécifier les noms de fichiers, leurs emplacements ou encore leur taille.

Pour les besoins de notre exemple, nous allons choisir d'utiliser la première option et utiliser ainsi le modèle.

▪ Etape 8 : Configuration de la restauration de la base de données

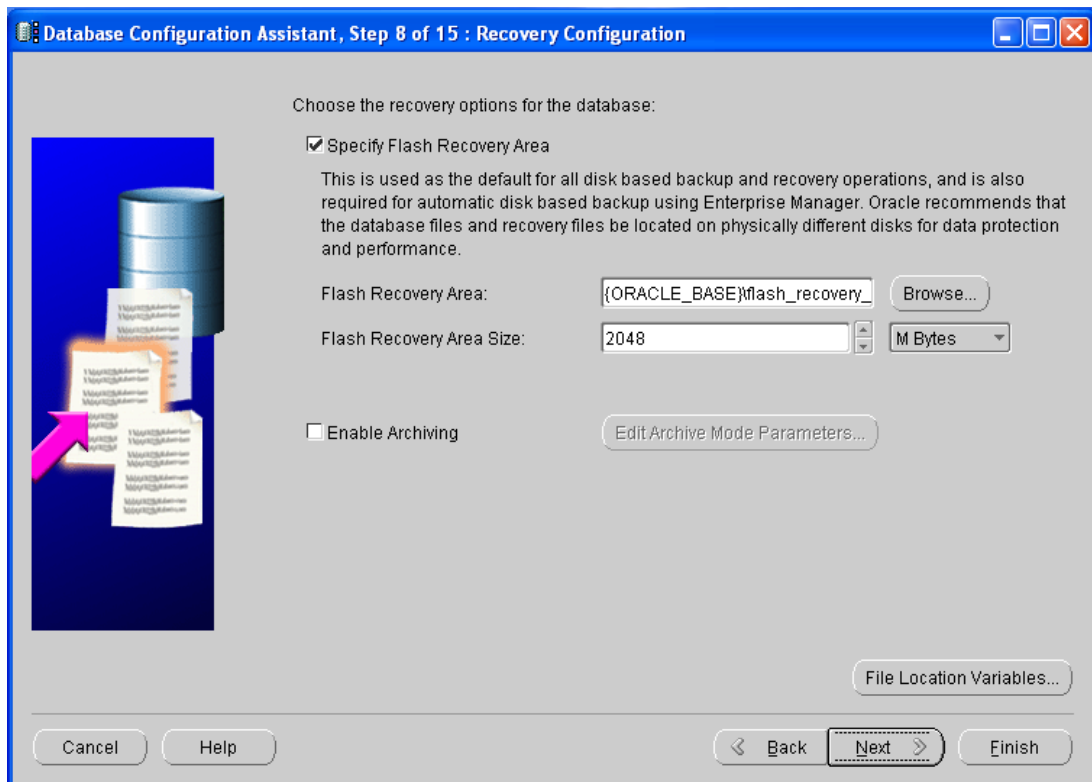
Lors de la création d'une base de données, il est important de la configurer dans l'optique de pouvoir restaurer ses données dans l'éventualité d'une panne du système. A cet effet, les fichiers Redo Log contiennent des enregistrements qui ont été réalisés sur les fichiers de données. Les fichiers Redo Logs sont stockés dans des groupes Redo Log et vous devez avoir au moins deux groupes Redo Log pour votre base de données. Après que tous les fichiers Redo Log d'un groupe aient été remplis, le processus d'écriture des logs (LGWR) change de groupe de Redo Log. Oracle peut sauvegarder automatiquement les groupes inactifs vers une ou plusieurs destinations, appelées communément Archive Redo Log (Archive Log). Plus concrètement, Oracle peut archiver les fichiers Redo Log.

Pour plus d'informations, veuillez vous reporter à la section Gestion des Redo Log de ce cours.

Lorsque vous créez votre base de données, vous pouvez choisir les options suivantes :

- **Specify Flash Recovery Area** : cette option permet de spécifier un emplacement pour de sauvegarde et restauration pour les archives des fichiers Redo Log. Vous devez spécifier un répertoire ainsi que la taille de celui-ci.
- **Enable Archiving** : cette option permet d'enclencher le dispositif d'archivage des fichiers Redo Logs de la base de données. La sélection de cette option est l'équivalent d'enclenchement de l'archivage dans l'Oracle Enterprise Manager Database Control ou encore le fait de lancer la base dans le mode ARCHIVELOG.
Oracle recommande de sélectionner cette option car elle permet une meilleure protection de la base en cas de panne software ou hardware. Cependant, si vous ne sélectionnez pas cette option, vous pouvez tout de même lancer la base en mode ARCHIVELOG ultérieurement.

Dans notre exemple de création de base de données, nous allons uniquement choisir l'option Flash Memory Area, dont l'emplacement est défini par défaut par Oracle à l'adresse suivante : {ORACLE_BASE}\flash_memory_area, et une taille par défaut de 2048 Mb.



▪ Etape 9 : Contenu de la base de données

Lors de la création de la base de données, vous pouvez la remplir de données via deux méthodes :

- **Sample schemas** : cette option permet de charger dans la base des schémas exemples dans la base de données (Espace disque logique (tablespace) EXAMPLE). Cette option peut s'avérer intéressante en termes de compréhension.

- **Custom Scripts** : dans cet onglet, vous pouvez spécifier un ou plusieurs scripts SQL qui seront exécutés après la création de la base de données. Ce genre de scripts est utile afin de réaliser des tâches « postinstallation » telles que le chargement de schémas personnalisés. Il est important de préciser que ces scripts doivent contenir une chaîne de caractères spécifique à la connexion, qui identifie la base de données. Plus de précisions sont disponibles dans l'aide.

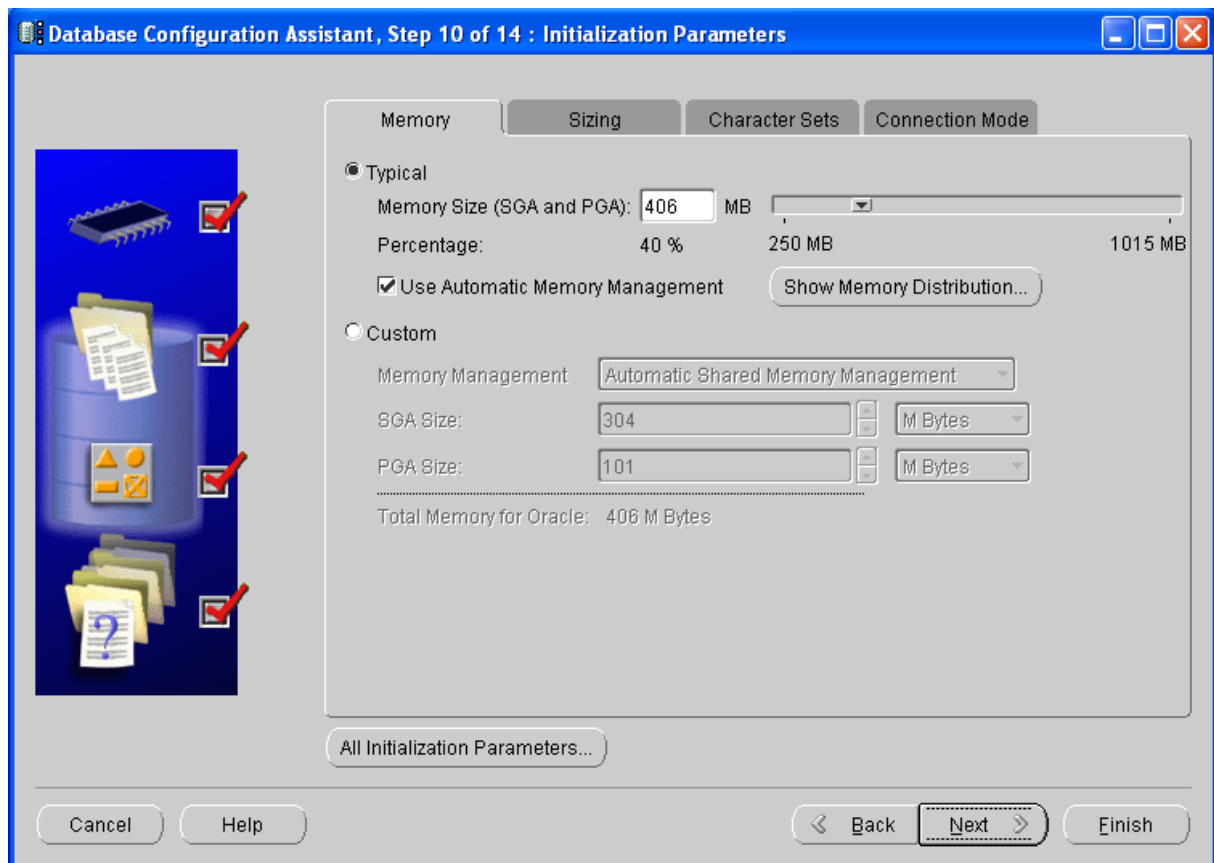
Pour notre exemple, choisissez l'option Sample schemas.

▪ Etape 10 : Initialisation des paramètres

Les différents onglets dans cette fenêtre permettent de changer réglages par défaut des paramètres d'initialisation. Ces paramètres peuvent être sous-divisés en quatre catégories :

- Memory
- Sizing
- Character Sets
- Connection Mode

Par ailleurs, vous pouvez également cliquer sur le bouton **All Initialization Parameters** afin d'afficher la liste de tous les paramètres d'initialisation de la base ainsi que leur réglage actuel.



Onglet Memory

Cet onglet permet de configurer l'ensemble des paramètres qui contrôlent la manière selon laquelle la base de données gère sa mémoire. Plusieurs méthodes de gestion de mémoire sont disponibles :

- **Typical** : cette méthode ne requiert pas une importante configuration système et alloue de la mémoire en termes de pourcentage de la mémoire physique totale du système. Aussi, sélectionnez Typical et entrez un pourcentage.
- **Custom** : cette méthode requiert plus de configuration que la précédente mais permet un meilleur contrôle de l'utilisation de la mémoire du système par la base de données. Vous avez le choix entre deux sous-options :
 - *Automatic Shared Memory Management* permet de spécifier la quantité de mémoire utilisée par le SGA (System Global Area) et le PGA (Program Global Area). Cette option permet de partager automatiquement la mémoire entre ces deux entités selon les besoins.
 - *Manual Shared Memory Management* : cette option vous permet de spécifier la quantité de mémoire pour chaque entité précitée. Cependant, le partage de la mémoire n'est pas réalisée par le système de manière automatique.

Onglet Sizing

Cet onglet permet de spécifier la taille du plus petit bloc de données et le nombre maximum de processus utilisateurs qui peuvent se connecter simultanément à la base de données.

- **Taille des blocs de données (Block Size)**
Cette option permet de définir la taille des blocs de données de la base. Par défaut, la taille d'un bloc de données est de 8Ko. Il est important de préciser que le choix d'une autre taille de bloc requiert des connaissances avancées et ne doit être réalisé uniquement si ce la est vraiment nécessaire.
- **Processus (processes)**
Dans ce champ, vous pouvez spécifier le nombre maximum de processus qui peuvent se connecter à la base de données en même temps. A noter que le nombre par défaut (150 processus) est approprié pour de nombreux environnements.
Ces processus se déclinent en processus d'arrière plans, processus utilisateurs et processus d'exécution.

Onglet Character Set

Cet onglet permet de définir les caractères utilisés par la base de données. Le caractère set détermine quels langages peuvent être représentés dans la base. La plupart du temps, il est conseillé de choisir les paramètres par défaut proposés. Pour plus de précision, référez vous à la documentation Oracle.

Onglet Connection Mode

Cet onglet permet de choisir un mode de connexion pour la base de données. Vous pouvez lancer la base selon deux modes différents :

- **Dedicated Server Mode** : ce mode met en place un processus serveur dédié à chaque processus utilisateur. Vous pouvez sélectionner cette option si le nombre de client attendu est faible, par exemple 50 ou moins. C'est le mode par défaut.
- **Shared Server Mode** : ce mode permet de partager les ressources de la base de données à plusieurs connexions clientes. Il est utilisé dans le cas où un grand nombre de connexions cliente est prévu et pourrait impliquer une utilisation trop intensive de la mémoire et des autres ressources systèmes. Dans le cas du choix de cette option, vous devez spécifier le nombre de processus serveur que vous voulez créer lors du démarrage d'une instance Oracle. Pour plus d'informations, cliquer sur Help.

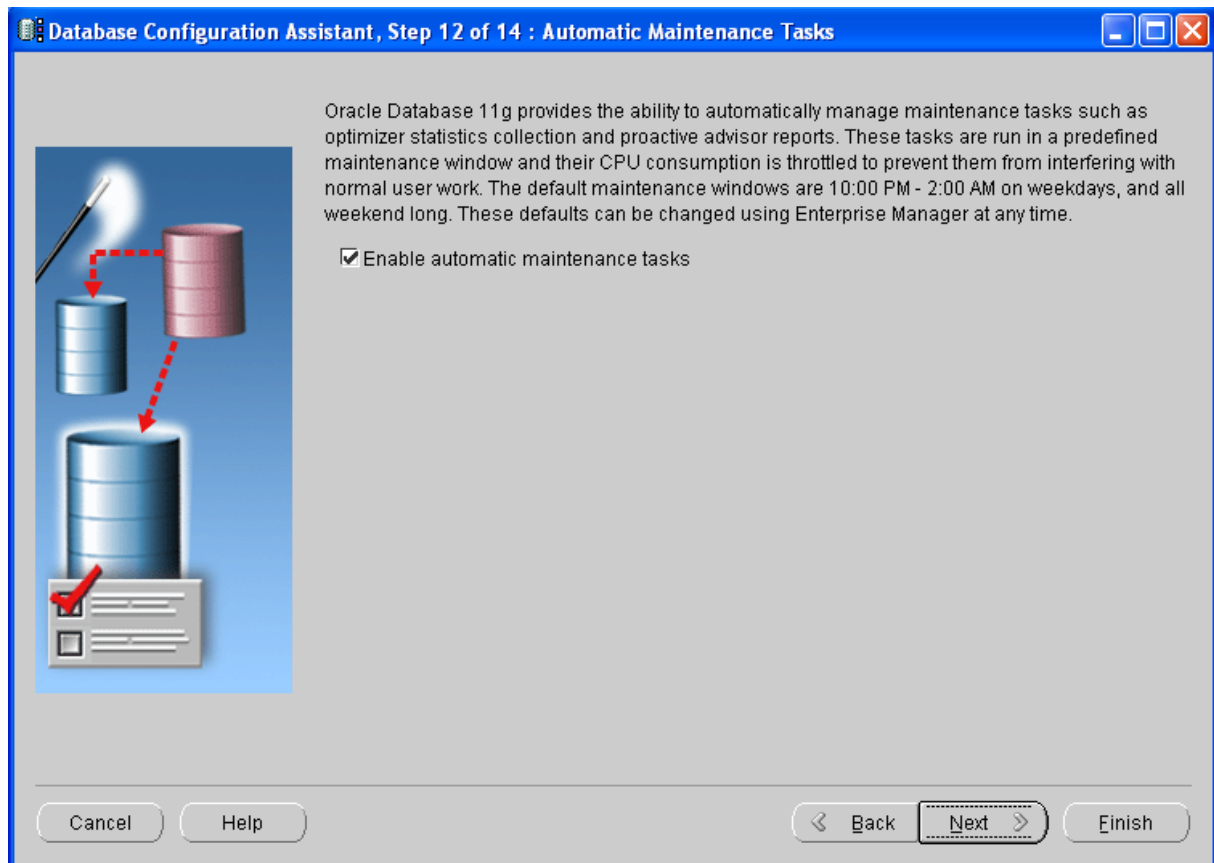
Pour la création de notre base de données exemple, nous allons sélectionner l'ensemble des paramètres par défaut.

– **Etape 11 : Réglages de la sécurité**

Dans cette fenêtre, vous pouvez choisir d'utiliser soit des réglages de sécurité améliorés soit utiliser ceux d'une version précédente d'Oracle. Par exemple, les premiers réglages contiennent des mots de passe sensible à la casse.

– **Etape 12 : Tâche de maintenance automatique**

Cette page permet de choisir d'activer (ou non) des tâches de maintenance automatique, propres à Oracle 11g. Celles-ci sont lancées à intervalles réguliers afin de réaliser des opérations de maintenance de la base de données.

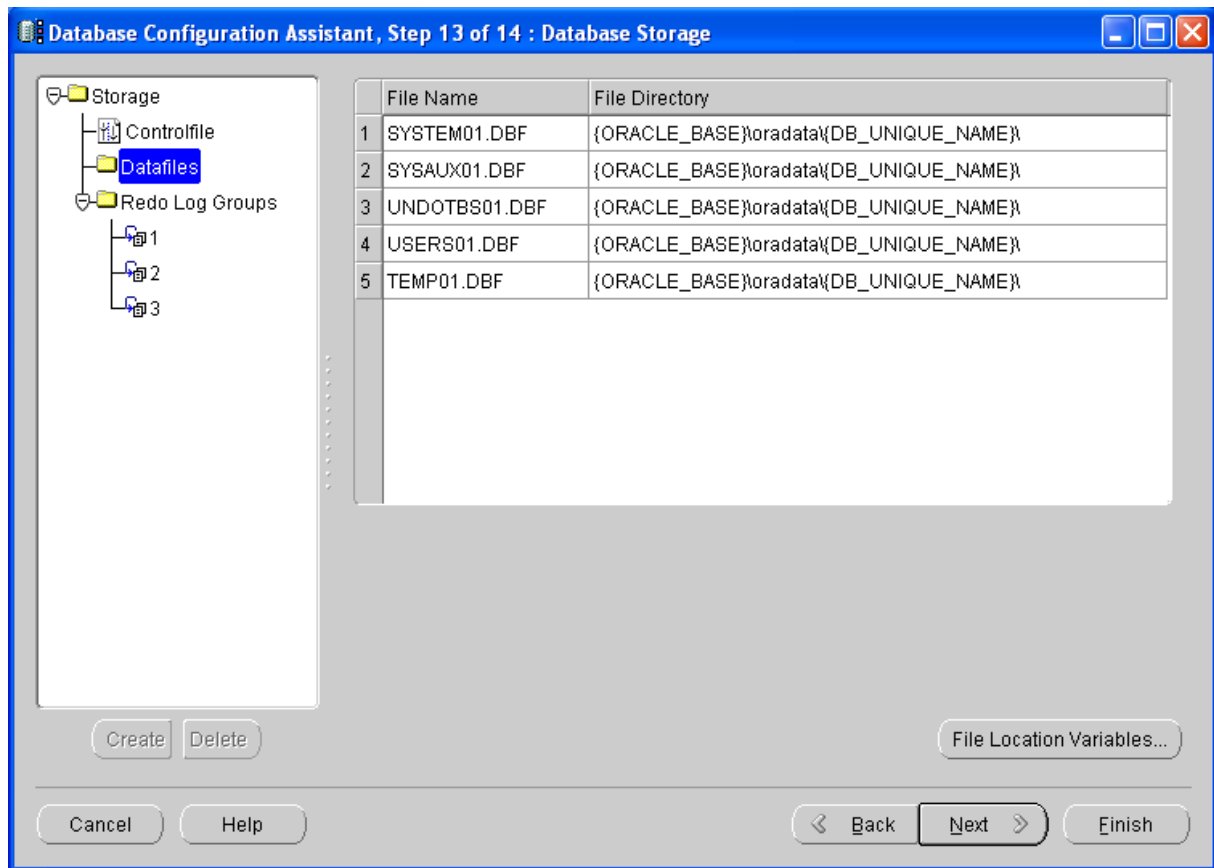


- **Automatic Optimizer Statistics Collection** : elle permet de réaliser des statistiques sur l'ensemble des schémas de la base de données.
- **Automatic Segment Advisor** : elle identifie les segments qui ont de l'espace mémoire libre dédié à la restauration et donne des recommandations sur la façon de les défragmenter.
- **Automatic SQL Tuning Advisor** : elle examine les performances de requêtes SQL nécessitant beaucoup de ressources et donne des conseils pour les optimiser.

Il est intéressant d'activer ces tâches de maintenance dans une optique d'amélioration et d'optimisation de la base de données.

– Etape 13 : Stockage de la base de données

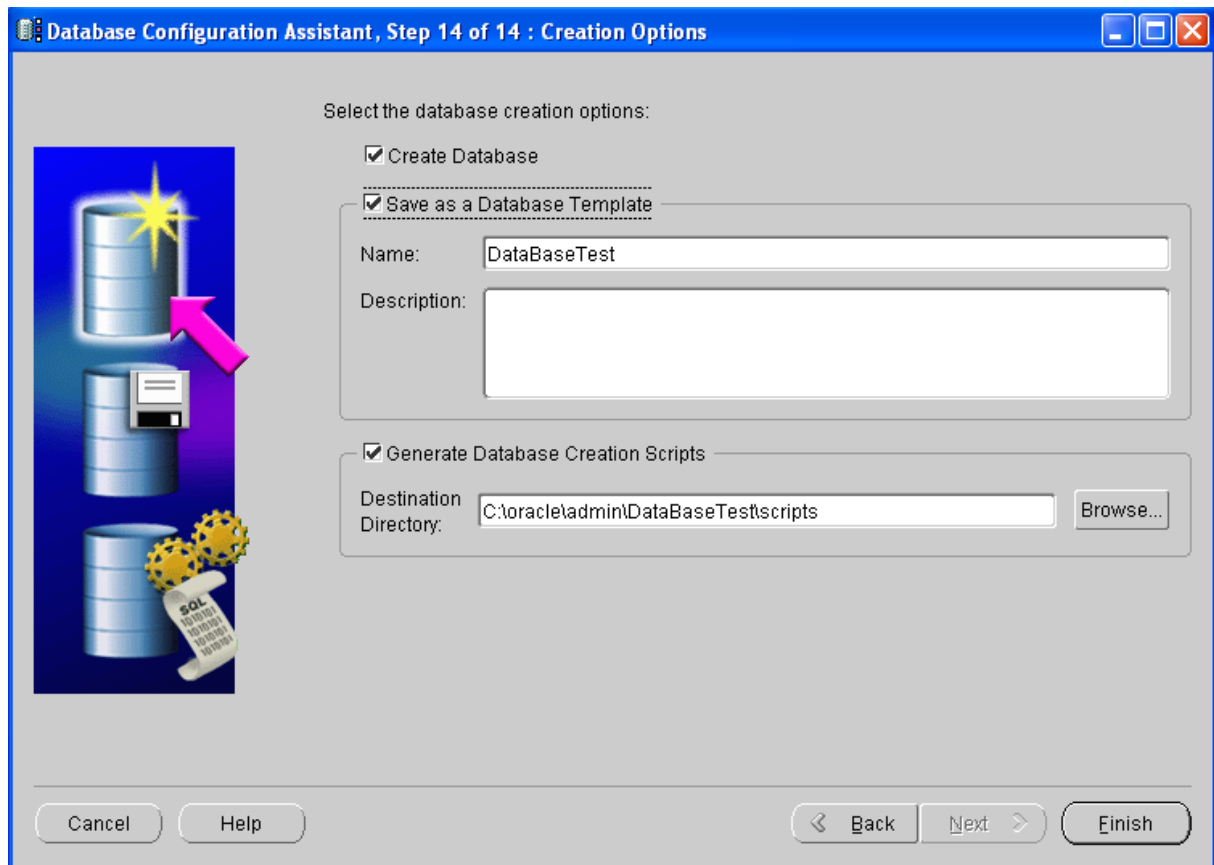
Dans cette fenêtre, une arborescence permet de visualiser la structure de stockage de votre base de données (control files, datafiles, redo log groups...). Il est également possible de modifier cette arborescence dans le cas où vous avez choisi un modèle de base de données.



– Etape 14 : options de création de la base

Dans cette page, vous pouvez choisir différentes options pour la création de la base de données.

- **Create Database** : sélectionnez cette option pour créer la base de données.
- **Save as a Database Template** : sélectionnez cette option si vous désirez sauvegarder la définition de votre base de données (l'ensemble de sa configuration) comme un modèle afin d'éventuellement le réutiliser.
- **Generate Database Creation Scripts** : cette option permet de générer un script de création de la base de données.



Dans notre exemple, sélectionnez les trois options.

Cliquez sur finish. Votre base de données a été créée.

Remarques à propos du DBCA

L'assistant de configuration de base de données Oracle peut être également utilisé pour changer la configuration d'une base de données existante (comme les réglages de sécurité, le mode de connexion (dedicated server ou shared server) , gérer des modèles de base de données ou encore supprimer une base.

Création d'une base de données sans le DBCA, avec la commande CREATE DATABASE

L'utilisation de la commande CREATE DATABASE est une approche plus manuelle afin de créer une base de données. Il est important de préciser que si vous voulez utiliser cette commande, vous devez réaliser des actions supplémentaires avant d'avoir une base de données opérationnelle. En effet, il est nécessaire de créer des vues sur les tables du dictionnaire de données et d'installer des packages standards PL/SQL. Ces actions seront réalisées au travers de l'exécution de scripts disponible dans l'environnement Oracle.

- **Etape 1 : spécification d'un identifiant d'instance (SID)**

A cette étape vous devez choisir un identifiant pour votre base de données. Pour ce faire, il vous suffit de modifier la variable d'environnement ORACLE_SID.

Comment modifier une variable d'environnement ?

Sous Linux (dans un shell) :

```
ORACLE_SID=DataBaseTest  
  
Export $ORACLE_SID
```

Sous windows (console MsDos) :

```
Set ORACLE_SID= DataBaseTest
```

- **Etape 2 : s'assurer que toutes les variables d'environnement requises sont configurées**

Mise à part la variable d'environnement ORACLE_SID, il faut également vérifier que ORACLE_HOME soit bien configurée.

Sous Windows, OUI assigne automatiquement des valeurs à ORACLE_HOME et ORACLE_SID. Cependant, si vous n'avez pas créé de base de données à l'installation, vous devez spécifier un SID dans cette variable avant la création d'une base de données.

Sous linux, les variables d'environnement précitées doivent être configurées. De plus, il est intéressant de modifier la variable PATH afin d'inclure le répertoire ORACLE_HOME/bin.

- **Etape 3 : Vérification des privilèges**

Vous devez vérifier que vous disposez des privilèges systèmes nécessaires afin de pouvoir créer une base de données. En d'autres termes vous devez être connecté en administrateur sur le système. Sous Linux, vous devez en général appartenir au groupe d'utilisateurs dba.

Sous Windows, l'utilisateur qui a installé le logiciel Oracle est directement placé dans le groupe d'utilisateurs ayant les privilèges nécessaires à la création d'une base de données.

- **Etape 4 : Création d'un fichier de paramètre d'initialisation**

Quand une instance Oracle est lancée, cette dernière utilise un fichier de paramètres d'initialisation. Ce fichier peut être un fichier texte qui peut être créé ou modifié par un éditeur de texte ou un fichier binaire qui est créé et automatiquement modifié par la base de données (server parameter file).

A cette étape nous allons créer un fichier texte et par la suite nous allons créer un server parameter file grâce à ce dernier.

Cet exemple de fichier texte de paramètre d'initialisation est nommé init.ora et il se situe, sur la plupart des systèmes d'exploitation, à ces emplacements :

Pour Windows:

%ORACLE_HOME%\database

Pour Linux

ORACLE_HOME/dbs.

Ce fichier est celui qui a été généré lors la précédente création de base de données via le DBCA.

Dans ce fichier, vous devez remplacer '<ORACLE_BASE>' par l'emplacement par défaut qui a été défini à l'installation.

```
#####
# Copyright (c) 1991, 2001, 2002 by Oracle Corporation
#####

#####
# Cache and I/O
#####
db_block_size=8192
# Taille des blocs de données : 8kb

#####
# Cursors and Library Cache
#####
open_cursors=300

#####
# Database Identification
#####
db_domain=""
db_name=DataBase
#nom de la base de données

#####
# File Configuration - Emplacement des fichiers de contrôle et des espaces de recovery
#####
control_files=("E:\app\DTR\oradata\DataBaseTest\control01.ctl",
"E:\app\DTR\oradata\DataBaseTest\control02.ctl",
"E:\app\DTR\oradata\DataBaseTest\control03.ctl")
db_recovery_file_dest=E:\app\DTR\flash_recovery_area
db_recovery_file_dest_size=2147483648

#####
# Miscellaneous
#####
compatible=11.1.0.0.0
db_unique_name=DataBaseTest
diagnostic_dest=E:\app\DTR
memory_target=425721856
#memoire allouée pour la base de données

#####
# Processes and Sessions
#####
processes=150
#nombre de processus qui peuvent être connectés à la base en même temps

#####
# Security and Auditing
#####
audit_file_dest=E:\app\DTR\admin\DataBaseTest\adump
audit_trail=db
```

```
remote_login_passwordfile=EXCLUSIVE

#####
# Shared Server
#####
dispatchers="(PROTOCOL=TCP) (SERVICE=DataBaseTestXDB)"

#####
# System Managed Undo and Rollback Segments
#####
undo_Espace disque logique (tablespace)=UNDOTBS1
# Espace disque logique (tablespace) UNDO
```

– Etape 5 : (Pour Windows uniquement) Création d'une instance

Sur une plateforme Windows, avant de pouvoir se connecter à une instance, vous devez la créer manuellement si elle n'existe pas encore. La commande ORADIM crée une instance Oracle par la création d'un nouveau service Windows.

Pour créer une instance Oracle, entrez cette commande dans une console MS-DOS :

```
oradim -NEW -SID sid -STARTMODE MANUAL -PFILE pfile
```

Où *sid* est le nom du SID choisi précédemment (DataBaseTest par exemple) et *pfile* le chemin complet du fichier de paramètres d'initialisation que nous avons vu précédemment.

Note : cette commande crée une instance Oracle mais ne la démarre pas.

– Etape 6 : Se connecter à l'instance

Lancez SQL*Plus et connectez vous à votre instance Oracle avec le système de privilèges SYSDBA.

Pour ce faire :

```
$ sqlplus /nolog
SQL> CONNECT / AS SYSDBA
```

Sql*Plus affiche le message suivant : Connected to an idle instance.

– Etape 7 : création d'un Server Parameter File

Le Server Parameter File (SPFILE) permet de changer les paramètres d'initialisation en utilisant la commande ALTER SYSTEM et fait persister les changements après arrêt et redémarrage de la base de données. La création de ce fichier se réalise grâce au fichier texte d'initialisation que nous avons réalisé précédemment.

La commande SQL*Plus suivante crée un SPFILE à l'emplacement par défaut avec un nom par défaut :

```
CREATE SPFILE FROM PFILE;
```

Vous pouvez également spécifier les noms ainsi que les chemins complets d'accès pour les deux fichiers si vous n'utilisez pas les noms et emplacements par défaut.

Note : la base de données doit être redémarrée afin que le SPFILE puisse prendre effet.

– Etape 8 : Démarrage de l'instance

Nous allons démarrer l'instance sans monter la base de données. D'ordinaire, ce type d'action est réalisé lors de la création ou la maintenance d'une base de données.

Dans cet exemple, vous devez utiliser la commande STARTUP avec la clause NOMOUNT. Les PFILE et SPFILE étant situés aux emplacements par défaut, vous n'avez pas besoin de spécifier les chemins pour y accéder dans la commande précédente.

A cette étape, l'espace mémoire relatif à l'instance est alloué et ses processus sont lancés. Cependant, la base de données en elle-même n'existe pas encore.

– Etape 9 : création de la base de données

Le code suivant crée la base de données DataBaseTest. Ce nom de base doit être en accord avec le paramètre DB_NAME du fichier de paramètres d'initialisation. Par ailleurs, cet exemple considère les éléments suivants :

- Le fichier de paramètres d'initialisation spécifie le nombre et l'emplacement des fichiers de contrôle au travers du paramètre CONTROL_FILES.
- Le répertoire E:\app\DTR\oradata\DataBaseTest existe. (celui-ci est bien sûr cité à titre d'exemple, il dépend de la variable d'environnement ORACLE_BASE)

```
CREATE DATABASE DataBase
USER SYS IDENTIFIED BY sys_password
USER SYSTEM IDENTIFIED BY system_password
LOGFILE
GROUP 1 ('E:\app\DTR\oradata\DataBaseTest\redo01.log') SIZE 100M,
GROUP 2 (' E:\app\DTR\oradata\DataBaseTest\redo02.log') SIZE 100M,
GROUP 3 (' E:\app\DTR\oradata\DataBaseTest\redo03.log') SIZE 100M
MAXLOGFILES 5
MAXLOGMEMBERS 5
MAXLOGHISTORY 1
MAXDATAFILES 100
CHARACTER SET US7ASCII
NATIONAL CHARACTER SET AL16UTF16
EXTENT MANAGEMENT LOCAL
DATAFILE ' E:\app\DTR\oradata\DataBaseTest\system01.dbf' SIZE 325M REUSE
SYSAUX DATAFILE ' E:\app\DTR\oradata\DataBaseTest\sysaux01.dbf' SIZE 325M REUSE
DEFAULT TABLESPACE users
DATAFILE ' E:\app\DTR\oradata\DataBaseTest\users01.dbf'
SIZE 500M REUSE AUTOEXTEND ON MAXSIZE UNLIMITED
DEFAULT TEMPORARY TABLESPACE tempts1
TEMPFILE ' E:\app\DTR\oradata\DataBaseTest\temp01.dbf'
SIZE 20M REUSE
UNDO TABLESPACE undotbs
DATAFILE ' E:\app\DTR\oradata\DataBaseTest\undotbs01.dbf'
SIZE 200M REUSE AUTOEXTEND ON MAXSIZE UNLIMITED;
```

Une base de données est créée avec les caractéristiques suivantes :

- La base de données est nommée DataBase.
- Trois fichiers de contrôle ont été créés comme il était spécifié par le paramètre d'initialisation CONTROL_FILES.
- Les mots de passe pour les comptes SYS et SYSTEM sont définis aux valeurs qui ont été spécifiées au préalable
- La nouvelle base de données contient trois fichiers Redo Log comme il était spécifié dans la clause LOGFILE
- MAXLOGFILES, MAXLOGMEMBERS et MAXLOGHISTORY définissent les limites pour les Redo Logs.
- MAXDATAFILES spécifie le nombre maximum de fichiers de données (datafiles) qui peuvent être ouverts dans la base de données. Ce nombre affecte la taille initiale des fichiers de contrôle.
- Le character set US7ASCII est utilisé pour stocker les données dans la base.
- Le character set AL16UTF16 est spécifié comme le NATIONAL CHARACTER SET, utilisé pour stocker les données dans des colonnes spécifiquement définies comme NCHAR, NCLOB ou encore NVARCHAR2
- Le tablespace SYSTEM a été créée, qui est reliée au système de fichiers : 'E:\app\DTR\oradata\DataBaseTest\sysaux01.dbf', comme le stipule la clause SYSAUX DATAFILE.
- La clause DEFAULT TABLESPACE créé et nomme un tablespace permanent par défaut pour cette base de données.
- La clause DEFAULT TEMPORARY TABLESPACE créé et nomme un tablespace temporaire par défaut pour cette base de données.
- La clause UNDO TABLESPACE créé et nomme un tablespace UNDO qui est utilisée pour stocker les données Undo pour cette base de données.
- Les fichiers Redo Log ne seront pas archivés car la clause ARCHIVELOG n'est pas spécifiée dans la commande CREATE DATABASE. Cependant, vous pouvez, a posteriori, utiliser la commande ALTER DATABASE afin de passer en mode ARCHIVELOG.

– Etape 10 : Création de tablespace additionnels

Afin de rendre la base de données fonctionnelle, vous devez créer des tablespace supplémentaires pour les données d'application. Le script exemple suivant vous permet d'en créer deux en plus :

Création d'un tablespace pour les données d'application


```
CREATE TABLESPACE apps_tbs LOGGING
DATAFILE ':\app\DTR\oradata\DataBaseTest\apps01.dbf'
SIZE 500M REUSE AUTOEXTEND ON NEXT 1280K MAXSIZE UNLIMITED
EXTENT MANAGEMENT LOCAL;
```

Création d'un tablespace pour les indexes, séparé du tablespace users (optionnel)

```
CREATE TABLESPACE indx_tbs LOGGING
DATAFILE 'e:\app\DTR\oradata\DataBaseTest\indx01.dbf'
SIZE 100M REUSE AUTOEXTEND ON NEXT 1280K MAXSIZE UNLIMITED
EXTENT MANAGEMENT LOCAL;
```

– Etape 11 : Exécution de scripts pour la création des vues du dictionnaire de données

A cette étape vous devez exécuter des scripts afin de construire les données des vues du dictionnaire de données, les synonymes mais aussi installer les packages PL/SQL nécessaires au bon fonctionnement de SQL*Plus.

Dans la documentation Oracle, on trouve un script déjà réalisé qui permet de réaliser ces actions.

```
@?/rdbms/admin/catalog.sql
@?/rdbms/admin/catproc.sql
@?/sqlplus/admin/pupbld.sql
EXIT
```

Remarque :

Le signe @ permet à SQL*Plus d'exécuter un script (fichier texte). Le point d'interrogation est une variable SQL*Plus qui indique le répertoire racine d'Oracle.

Les scripts qui sont lancés sont les suivants :

- **CATALOG.SQL** : ce script crée les vues des tables du dictionnaire de données, les vues dynamiques et les synonymes publics pour la plupart des vues. Egalement, ce script donne un accès PUBLIC aux synonymes.
- **CATPROC.SQL** : ce script exécute l'ensemble des scripts requis pour ou utilisés avec le langage PL/SQL.
- **PUBBLD.SQL** : ce script est requis par SQL*Plus. Il permet à SQL*Plus de permettre à l'utilisateur de désactiver des commandes.

Lancement et arrêt d'une base de données

Procédures qui sont impliquées dans le démarrage et l'arrêt d'une base de données Oracle.

Démarrage d'une instance et d'une base de données

Lors du démarrage d'une base de données, une instance de cette base est créée et l'état (la disponibilité) de cette dernière doit être déterminé. D'ordinaire, le démarrage d'une instance précède le montage et l'ouverture d'une base de données. Ces actions rendent la base disponible pour chaque utilisateur pour qu'il puisse se connecter et réaliser des opérations standard d'accès aux informations.

Il est possible de démarrer une instance de base de données via SQL*Plus, Recovery Manager ou encore Enterprise Manager.

Pour les besoins de ce cours, nous utiliserons uniquement SQL*Plus. Pour plus de précisions quant aux deux autres, référez vous à la documentation Oracle.

Sous SQL*Plus, la commande permettant de démarrer une instance base de données est la suivante :

```
STARTUP
```

Remarques sur les paramètres d'initialisation et la commande STARTUP

Afin de démarrer une instance, la base de données doit lire les paramètres de configuration de la première à partir de soit un server parameter file (SPFILE) soit un fichier texte de paramètre d'initialisation (PFILE).

Lorsque vous lancez la commande SQL*Plus STARTUP, la base de données tente de lire les paramètres d'initialisation à partir d'un SPFILE dans le répertoire par défaut. Si ce dernier fichier n'est pas trouvé, la base de données cherche alors le PFILE.

Le répertoire par défaut est :

Pour Windows:

```
%ORACLE_HOME%\database
```

Pour Linux

```
ORACLE_HOME/dbs.
```

Dans le répertoire par défaut, Oracle localise le fichier de paramètres d'initialisation en examinant des noms de fichiers dans l'ordre suivant :

1. spfile\$ORACLE_SID.ora
2. spfile.ora
3. init\$ORACLE_SID.ora

Conditions nécessaires au démarrage d'une instance

Comme il a été expliqué dans la partie traitant de la création d'une base de données avec la commande CREATE DATABASE, il est important de vérifier que les variables d'environnement sont correctement définies.

Par la suite, lancez SQL*Plus sans vous connecter à une base par la commande suivante :

```
sqlplus /nolog
```

Connectez vous ensuite à Oracle en tant que SYSDBA via la commande suivante :

```
Connect username as SYSDBA ;
```

Ainsi vous êtes connecté à la base de données et prêt à démarrer l'instance de votre base de données.

Démarrage de l'instance

C'est ici que nous allons utiliser la commande STARTUP. Cette dernière peut être utilisée de différentes façons :

Démarrage de l'instance sans monter la base de données.

Ceci ne permet pas d'avoir accès à la base de données et est d'habitude utilisé pour la création ou re-création d'une base de données. La commande associée est la suivante :

```
STARTUP NOMOUNT
```

Démarrage de l'instance et montage de la base de données, cette dernière restant fermée.

Cette méthode permet certain activités administrateur mais ne permet un accès général à la base. La commande associée est la suivante :

```
STARTUP MOUNT
```

Démarrage de l'instance, montage de la base de données et ouverture de celle-ci.

Cette action peut être réalisée soit en mode restreint (permettant un accès uniquement aux administrateurs) soit en mode non-restreint (pour les tous les utilisateurs). Le mode restreint peut être intéressant pour les administrateurs lorsqu'ils doivent réaliser des tâches telles que l'exportation ou l'importation de données, un chargement de données (SQL*Loader) ou encore d'autres opérations.

En mode non-restreint : STARTUP

En mode restreint : STARTUP RESTRICT

Note : si vous avez choisi ce dernier mode, vous pouvez ultérieurement utiliser la commande ALTER SYSTEM afin de désactiver ce mode :

```
ALTER SYSTEM DISABLE RESTRICTED SESSION;
```

Forcer le démarrage d'une instance après un problème de démarrage ou d'arrêt.

La commande est la suivante :

```
STARTUP FORCE
```

Changer la disponibilité d'une base de données

Le changement de la disponibilité d'une base de données peut être réalisé dans le but de restreindre son accès pour des raisons de maintenance ou encore afin de mettre la base en mode lecture seule.

Monter une base de données

Lorsque vous avez besoin de réaliser des tâches administratives, la base de données doit être démarrée et montée sur l'instance, mais rester fermée (cf partie précédente). Cependant, si vous êtes dans le cas où l'instance a été démarrée mais que la base n'a pas été montée, vous pouvez utiliser la commande suivante afin de monter la base de données :

```
ALTER DATABASE MOUNT
```

Ouvrir une base de données fermée.

Vous pouvez être dans le cas où la base de données est montée mais n'est pas ouverte. Par cette commande, vous ouvrez la base :

```
ALTER DATABASE OPEN
```

Ouvrir une base de données en mode lecture-seule

Ouvrir une base dans ce mode permet d'empêcher tout changement éventuel sur les données. La commande suivante permet cette action :

```
ALTER DATABASE READ ONLY ;
```

Par la suite, si vous souhaitez désactiver ce mode, il vous suffit de taper la commande suivante :

```
ALTER DATABASE OPEN READ WRITE ;
```

Base de données ouverte en accès restreint

Comme il a été explicité précédemment, ce mode restreint l'accès à la base de données aux seuls utilisateurs disposant de privilèges administrateurs. Pour placer la base dans ce mode, utilisez la commande :

```
ALTER DATABASE ENABLE RESTRICTED SESSION
```

Par la suite, pour désactiver ce mode, tapez la commande suivante :

```
ALTER DATABASE DISABLE RESTRICTED SESSION
```

Arrêter une instance et base de données

La commande SQL*Plus qui permet d'arrêter une base de données est la suivante : SHUTDOWN. Elle peut être accompagnée de différentes clauses, qui permettent d'opérer des arrêts différents.

SHUTDOWN NORMAL : elle correspond aux situations standards et elle constitue la clause par défaut ; elle est donc optionnelle. Avant sa fermeture, la base attend que l'ensemble des utilisateurs se déconnecte.

SHUTDOWN IMMEDIATE : cette clause est utile lorsque qu'une sauvegarde de la base de données va être opérée, qu'une coupure de courant va être bientôt réalisée ou encore lorsque que la base de données ou une des ces applications fonctionne de manière anormale implique et l'administrateur ne peut pas contacter les utilisateurs afin qu'ils puissent se déconnecter ou encore que ces derniers ne puissent effectuer cette action.

Les transactions en cours sont prises en compte, ce qui peut parfois prendre du temps.

SHUTDOWN TRANSACTIONAL : cette clause permet de fermer la base de données mais uniquement après prise en compte de l'ensemble des transactions en cours sur la base. Cette clause permet aux utilisateurs ne pas perdre les actions qu'ils étaient en train de réaliser sur la base. Ces derniers sont par ailleurs tous déconnectés.

SHUTDOWN ABORT : cette clause est à utiliser qu'en dernier recours. En effet, si l'ensemble des autres clauses ne fonctionne pas, cette clause permet de fermer la base de données en déconnectant l'ensemble des utilisateurs, en abandonnant les transactions qui étaient en cours sur la base.

A propos de l'environnement Réseau

A la suite de l'installation d'Oracle, vous avez une base de données fonctionnelle avec un réseau client/serveur qui est configuré de façon minimale. Cette partie va s'attacher à décrire plus précisément ce réseau.

Un client est une application qui se connecte à Oracle afin d'envoyer ou de récupérer des données. Par ailleurs, Oracle Net est une couche logicielle qui est présente à la fois sur l'ordinateur client et sur l'ordinateur sur lequel est installé Oracle. Il permet d'établir et de maintenir la connexion entre le client et la base de données au travers du réseau.

Afin qu'un client ainsi qu'une base de données puissent communiquer, l'application cliente doit être capable d'identifier la base de données à laquelle il souhaite se connecter, et la base de données doit donner une identification ou une adresse.

Il est possible d'utiliser un nom de service (service name) afin de se connecter à une base de données. Un nom de service est une représentation logique de la base, et c'est de cette façon que cette dernière est présentée aux clients.

L'utilisation de nom de services permet par exemple le fait que les applications clientes n'ont pas besoin de connaître l'emplacement du serveur. Lorsqu'une base de données est déplacée à un autre emplacement, il suffit uniquement de reconfigurer Oracle Net et les modifications sur les applications clientes sont inutiles.

Listener Oracle Net

Sur le serveur de base de données, le listener Oracle Net est un processus qui écoute les requêtes de connexion des clients. Il reçoit ces dernières et gère leur transport vers le serveur.

Le fichier de configuration par défaut du listener est nommé listener.ora et se trouve dans le répertoire ORACLE_HOME/network/admin..

Ce fichier contient une adresse de protocole qui identifie la base de données. Cette adresse définit le protocole sur lequel le listener écoute et d'autres informations spécifiques. Par exemple, le listener peut être configuré pour écouter à cette adresse de protocole :

```
(DESCRIPTION =ADDRESS=(PROTOCOL=tcp)(HOST=MonServeur)(PORT=1521))
```

Cet exemple, montre une adresse TCP/IP qui spécifie le serveur de base de données sur lequel le listener opère ainsi qu'un numéro de port.

Le fichier listener.ora est automatiquement configuré pendant l'installation. Par ailleurs, de par le fait que tous les paramètres de configuration ont des valeurs par défaut, il est possible de lancer et d'utiliser le listener sans l'avoir au préalable configuré. Le listener par défaut est nommé LISTENER et ne supporte aucun nom de service au démarrage et écoute à cette adresse TCP/IP :

```
(ADDRESS=(PROTOCOL=tcp)(HOST=host_name)(PORT=1521))
```

Les noms de services que le listener peut connaître ,et ainsi rediriger les requêtes clientes, peuvent être configurés dans le fichier listener.ora mais ces informations peuvent être également

dynamiquement enregistrées auprès du listener. Cette façon de faire est appelée l'enregistrement de services. Elle est réalisée par le processus PMON (processus d'arrière plan d'une instance) de chaque instance de base de données qui a une configuration nécessaire de son fichier de paramètre d'initialisation. Aussi, cette méthode ne requiert aucune modification du fichier listener.ora.

Descripteur de connexion

Le client utilise un descripteur de connexion afin de spécifier la base de données à laquelle il désire se connecter. Ce descripteur contient un protocole et nom de service d'une base. De par le fait qu'une base de données peut éventuellement avoir de multiples noms de service, il faut donc donner un nom de service spécifique au descripteur.

Dans une base de données préconfigurée, il n'y a qu'un seul nom de service qui est par défaut le nom global de base de données

Prenons l'exemple d'un descripteur de connexion qui permet à un client de se connecter à une base de données dont le nom global est DataBaseTest.

```
(DESCRIPTION =ADDRESS=(PROTOCOL=tcp)(HOST=MonServeur)(PORT=1521))  
CONNECT DATA = SERVICE_NAME=DataBaseTest))
```

Requêtes de connexion

Un client lance une requête de connexion en fournissant une chaîne de caractères de connexion. Cette chaîne contient un nom d'utilisateur, un mot de passe et un identifiant de connexion. Ce dernier peut être le descripteur de connexion lui-même ou un nom qui correspond au descripteur, qui est stocké dans un fichier de configuration du serveur (explicité *infra*) en utilisant une méthode de nommage (également explicitée *infra*). Ce nom est appelé nom de service Net (net service name).

Méthodes de nommage

Une méthode de nommage est une méthode de résolution utilisée par un client afin de faire correspondre un identifiant de connexion à un descripteur de connexion lors d'une tentative de connexion à une base de données.

Nous pouvons distinguer trois types de méthodes de nommage :

Easy connect naming

Cette méthode de nommage permet aux clients de se connecter à une base de données en utilisant uniquement une chaîne de connexion TCP/IP qui contient le nom du serveur et le nom du service. Cette méthode ne requiert aucune configuration.

Local Naming

Cette méthode de nommage stocke les descripteurs de connexion, identifiés par leur nom de service net (net name service) dans un fichier de configuration, nommé tnsnames.ora, sur l'ordinateur client. Ce fichier se trouve à l'emplacement ORACLE_HOME/network/admin. Lorsqu'une base de données est créée via le DBCA, local naming est automatiquement configurée. Ensuite, vous devez utiliser l'assistant de configuration Net (NETCA) afin de créer les descripteurs de connexion ainsi que leur nom de service net associés.

Directory Naming

Cette méthode associe un service de base de données, un nom de service net ou encore un alias de service net à un descripteur de connexion stocké dans un serveur LDAP.

Assistant de configuration Net (Net Configuration Assistant)

Lors d'une installation classique, l'assistant de configuration Net (NETCA) configure automatiquement un listener par défaut appelé LISTENER que nous avons explicité *supra*. Cependant, pour une installation personnalisée, NETCA vous demande de configurer vous-même un nom de listener ainsi qu'une adresse de protocole de votre choix.

Instance Oracle

Cette partie va s'attacher à décrire une instance Oracle ainsi que les différentes parties qui la composent.

Présentation générale

Une instance Oracle est composée de deux parties différentes :

- **System Global Area (SGA)** : cette zone mémoire a pour rôle de stocker les données relatives aux fichiers de données dans le but de les partager entre différents processus.
- **Program Global Area (PGA)** : cette zone mémoire permet le bon fonctionnement des processus en stockant l'ensemble des données qui n'ont pas besoin d'être partagées.

- **Processus d'arrière plan** : ils ont pour rôle de s'occuper de la gestion des transferts de données entre la mémoire et le disque. De plus, ils réalisent des tâches concernant le fonctionnement de la base de données.

Une instance est nécessaire afin d'assurer le bon fonctionnement d'une base de données Oracle. En effet, sans celle-ci, la connexion à la base est impossible.

System Global Area (SGA)

La base de données utilise la SGA afin de distribuer les données relatives aux fichiers de données aux différents processus Oracle.

Cette zone mémoire est subdivisée en différentes zones mémoire :

- Shared Pool
- Database Buffer Cache
- Redo Log buffer

Shared Pool

Cette zone mémoire est utilisée afin de partager les renseignements concernant l'ensemble des objets de la base. Par ailleurs, elle contient également les informations concernant les privilèges assignés aux utilisateurs.

Le paramètre d'initialisation `SHARED_POOL_SIZE` permet de définir la taille de cette zone mémoire.

Elle peut être subdivisée en deux zones mémoire :

Library Cache

Cette zone mémoire a pour objectif de stocker des informations relatives aux commandes SQL qui ont été exécutées de façon récente. Ces informations sont le corps de la commande, sa version compilée ainsi que son plan d'exécution.

Oracle utilise cette zone mémoire dans le cas où une commande SQL est exécutée plusieurs fois. En effet, étant donné que la zone mémoire dispose déjà de la version compilée de la commande ainsi

que son plan d'exécution, Oracle peut directement utiliser ces informations et n'est donc pas obligé de créer à nouveau ces informations.

.

Dictionary Cache

Cette zone mémoire a pour rôle de stocker les structures, composants et privilèges liés aux objets de la base qui ont été utilisés récemment.

L'intérêt de cette zone mémoire réside dans le fait qu'Oracle n'est pas obligé de récupérer ces informations sur le disque lors d'exécution de commandes SQL concernant ces objets.

Database Buffer Cache

Cette zone mémoire a pour rôle de stocker les blocs de données qui ont été utilisés de manière récente. Plus précisément, lors de la première exécution d'une requête SQL, Oracle va rechercher sur le disque dur les données. Cependant, si cette requête est à nouveau exécutée, les blocs de données sont récupérés dans le buffer de la base de données et non à partir du disque dur. Cette zone mémoire permet donc des gains de temps conséquents.

La zone mémoire DataBase Buffer Cache est définie par deux paramètres d'initialisation :

DB_BLOCK_SIZE : détermine la taille d'un bloc de données Oracle.

DB_BLOCK_BUFFER : détermine le nombre de blocs de données que peut contenir cette zone mémoire.

Par ailleurs, il est intéressant de préciser qu'il est possible de visualiser la taille de cette zone mémoire en multipliant la valeur de DB_BLOCK_SIZE par la valeur du paramètre DB_BLOCK_BUFFER. Le résultat de cette opération est en Ko.

Redo Log Buffer

Cette zone mémoire permet d'enregistrer l'ensemble des informations relatives aux transactions qui ont été réalisées sur la base.

Cette zone mémoire est de type circulaire et séquentiel. Aussi, l'enregistrement de ces informations est réalisé de façon simultanée. Par ailleurs, comme la mémoire est de type circulaire, les données de celle-ci ne peuvent être écrasées par Oracle tant qu'elles n'ont pas été enregistrées dans les fichiers Redo Log.

Program Global Area (PGA)

Cette zone mémoire est utilisée uniquement par les processus d'arrière plan ou encore les processus serveurs. Elle n'est pas partagée et est allouée lorsque l'instance est démarrée.

La PGA contient les ensembles suivants :

- **Sort Area** : cette zone de tri réalise, comme son nom l'indique, les tris relatifs aux requêtes SQL exécutées par l'utilisateur. Par ailleurs, le paramètre d'initialisation `SORT_AREA_SIZE` permet de définir la taille de la zone mémoire `SORT_AREA`.
- **Les renseignements relatifs aux sessions** : cette zone contient les renseignements relatifs aux sessions ainsi que des informations concernant les privilèges de l'utilisateur.
- Une zone qui permet de renseigner l'état des curseurs de l'utilisateur.
- **Stack Space** : cette zone mémoire contient l'ensemble des variables de sessions et d'environnement relatives à l'utilisateur.

Processus d'arrière plan

Le fonctionnement interne de la base de données est assuré par un ensemble de processus appelés processus d'arrière plan (background process).

Nous pouvons distinguer quatre processus principaux que sont DBWR, LGWR, PMON, SMON.

DBWR (Database Writer)

Ce processus a pour rôle d'écrire dans les fichiers de données de la base les bloc de données qui ont été modifiés et qui se trouvent dans la SGA. Aussi, lorsque qu'une requête SQL a pour conséquence une modification des données de la base (un INSERT par exemple) le processus DBWR modifie en conséquence les fichiers de données correspondants.

LGWR (Log Writer)

Ce processus a pour objectif d'écrire les informations contenues dans la zone mémoire Redo Log Cache dans les fichiers Redo Log. Pour ce faire, ce processus attend la réception d'un ordre COMMIT. Ensuite, il réalise l'écriture dans les fichiers Redo Log.

PMON (Process Monitor)

Ce processus est consacré aux processus utilisateurs. Son rôle est d'annuler les transactions relatives à une session utilisateur lorsque, par exemple, cette dernière est interrompue de manière anormale. Ce processus permet aussi de libérer l'ensemble des verrous inhérents à la session ainsi que l'ensemble des ressources utilisées par cette dernière.

SMON (System Monitor)

Le rôle de ce processus est la vérification de la cohérence (synchronisation des données) de la base de données et son rétablissement en cas d'incident. En effet, si nous sommes dans le cas où l'arrêt de la base ne s'est pas déroulé de manière correct, ce processus analyse les informations contenues dans les fichiers Undo et procède à l'annulation des transactions qui n'ont pas fait l'objet d'un COMMIT.

Ce processus a donc pour rôle de libérer les ressources qui sont utilisées de manière superflue par le système.

Création d'une instance Oracle

Les méthodes de création d'une instance Oracle diffèrent selon le système d'exploitation sur lequel est installé Oracle. Nous allons voir les méthodes concernant Windows et Linux.

Sous Windows

Sous ce système d'exploitation, la création d'un processus d'instance doit se faire sous la forme d'un service. L'utilitaire ORADIM permet de créer, modifier et supprimer un service Windows. Aussi, nous allons l'utiliser pour générer notre instance.

Prenons l'exemple suivant :

```
c:\> oradim -new -sid MONSID -startmode auto -pfile  
E:\app\DTR\admin\MikaTestGood\pfile\init.ora
```

Cette commande permet de créer un nouveau service (-new) pour l'instance MONSID (-sid). Ce service démarrera automatiquement au démarrage de Windows (-startmode auto). Nous faisons aussi référence au PFILE en donnant son emplacement (-pfile).

Sous Linux

Il suffit de définir la variable d'environnement ORACLE_SID :

```
ORACLE_SID = MONSID
```

```
EXPORT $ORACLE_SID
```

Puis de créer un PFILE (init.ora). Pour cela vous pouvez vous baser sur l'exemple contenu dans la partie création de base de données de ce cours.

Enfin, il vous suffit de démarrer l'instance sous SQL*Plus.

Gestion des structures de stockage de la base de données

Comme il a été expliqué dans la partie introduction à l'architecture, Oracle est composé de structures physiques et de structures logiques. Les structures physiques peuvent être vues et utilisées au travers du système d'exploitation comme les fichiers physiques qui sont stockés en mémoire.

Les structures logiques sont créées et reconnues par Oracle et ne sont pas connus par le système d'exploitation. La principale structure logique d'une base de données est un espace disque logique (tablespace) et contient des fichiers physiques.

Le DBA se doit de comprendre les relations entre les structures physiques et logiques d'une base de données.

Oracle peut réaliser une grande partie de la gestion de sa structure. En effet, Oracle Enterprise Manager (DataBase Control) est une interface web permettant une gestion simple et intuitive ainsi qu'un suivi de la base de données.

Fichiers de contrôle

Un fichier de contrôle suit les composants physiques de la base de données. C'est le fichier de base qu'utilise la base afin de trouver l'ensemble des autres fichiers utilisés par la base de

données. Il est créé à la création de la base de données, et il est fréquemment modifié par le serveur Oracle. Par ailleurs, il est indispensable pour la restauration de la base.

Il comprend, entre autre, les informations suivantes :

- Informations sur la base (nom, id, date et heure de création)
- Informations sur les fichiers Redo log online
- Historique des Redo Log archivés
- Le mode d'archivage actuel
- Informations sur les espaces disque logique et les fichiers de données (nom de fichiers, statut read/write, online ou non)

De par leur importance, Oracle recommande que les fichiers de contrôle soient multiplexés. En d'autres termes, ces derniers doivent avoir plusieurs copies identiques. En ce qui concerne les bases de données créées avec le DBCA, trois copies des fichiers de contrôle sont automatiquement créées et sont synchronisées les unes avec les autres.

Si n'importe quel fichier de contrôle est défectueux, la base de données devient alors indisponible. Aussi, tant que vous avez une copie d'un fichier de contrôle défectueux, vous pouvez tout de même arrêter votre base de données et la créer à nouveau en utilisant la copie non défectueuse. Une autre option consiste à supprimer le fichier de contrôle défectueux du paramètre d'initialisation `CONTROL_FILES` et redémarrer votre base de données en utilisant les fichiers de contrôle restants.

La taille d'un fichier contrôle dépend des paramètres `MAXDATAFILES`, `MAXLOGFILES`, `MAXLOGMEMBERS`, `MAXLOGHISTORY`, et `MAXINSTANCES` définis lors de la création de la base.

Gestion des fichiers de contrôles

Informations sur les fichiers de contrôle

Il existe des vues du dictionnaire de données qui permettent d'obtenir des informations relatives à un fichier de contrôle :

- `V$DATABASE`
- `V$CONTROLFILE`
- `V$CONTROLFILE_RECORD_SECTION`
- `V$PARAMETER`

La vue `V$CONTROLFILE`

Cette vue a pour objectif d’afficher le nom ainsi que l’emplacement, et leur statut (qui peut NULL ou INVALID).

```
select value from v$parameter where name='control_files';
```

```
VALUE
-----
E:\APP\DTR\ORADATA\MIKATESTGOOD\CONTROL01.CTL,
E:\APP\DTR\ORADATA\MIKATESTGOOD\CONTROL02.CTL,
E:\APP\DTR\ORADATA\MIKATESTGOOD\CONTROL03.CTL
```

Une autre commande est également disponible afin d’afficher les informations précédentes.

```
show parameter control_files;
```

NAME	TYPE	VALUE
control_files	string	E:\APP\DTR\ORADATA\MIKATESTGOOD\CONTROL01.CTL, E:\APP\DTR\ORADATA\MIKATESTGOOD\CONTROL02.CTL, E:\APP\DTR\ORADATA\MIKATESTGOOD\CONTROL03.CTL

La vue V\$CONTROLFILE_RECORD_SECTION

Cette vue permet d’afficher des informations plus précises quant à un fichier de contrôle.

```
select * from v$controlfile_record_section;
```

TYPE	RECORD_SIZE	RECORDS_TOTAL	RECORDS_USED	FIRST_INDEX	LAST_INDEX	LAST_RECID
DATABASE	316	1	1	0	0	0
CKPT PROGRESS	8180	11	0	0	0	0
REDO THREAD	256	8	1	0	0	0
REDO LOG	72	16	3	0	0	3
DATAFILE	520	100	7	0	0	139
FILENAME	524	2298	12	0	0	0
TABLESPACE	68	100	7	0	0	5
TEMPORARY FILENAME	56	100	1	0	0	1
RMAN CONFIGURATION	1108	50	0	0	0	0
LOG HISTORY	56	292	52	1	52	52
OFFLINE RANGE	200	163	0	0	0	0

TYPE	RECORD_SIZE	RECORDS_TOTAL	RECORDS_USED	FIRST_INDEX	LAST_INDEX	LAST_RECID
ARCHIVED LOG	584	28	0	0	0	0
BACKUP SET	40	409	0	0	0	0
BACKUP PIECE	736	200	0	0	0	0
BACKUP DATAFILE	200	245	0	0	0	0
BACKUP REDOLOG	76	215	0	0	0	0
DATAFILE COPY	736	200	1	1	1	1
BACKUP CORRUPTION	44	371	0	0	0	0
COPY CORRUPTION	40	409	0	0	0	0
DELETED OBJECT	20	818	1	1	1	1
PROXY COPY	928	246	0	0	0	0
BACKUP SPFILE	124	131	0	0	0	0

TYPE	RECORD_SIZE	RECORDS_TOTAL	RECORDS_USED	FIRST_INDEX	LAST_INDEX	LAST_RECID
DATABASE INCARNATION	56	292	2	1	2	2

FLASHBACK LOG	84	2048	0	0	0	0
RECOVERY DESTINATION	180	1	1	0	0	0
INSTANCE SPACE RESERVATION	28	1055	1	0	0	0
REMOVABLE RECOVERY FILES	32	1000	0	0	0	0
RMAN STATUS	116	141	0	0	0	0
THREAD INSTANCE NAME MAPPING	80	8	8	0	0	0
MTTR	100	8	1	0	0	0
DATAFILE HISTORY	568	57	0	0	0	0
STANDBY DATABASE MATRIX	400	10	10	0	0	0
GUARANTEED RESTORE POINT	212	2048	0	0	0	0

TYPE	RECORD_SIZE	RECORDS_TOTAL	RECORDS_USED	FIRST_INDEX	LAST_INDEX	LAST_RECID

RESTORE POINT	212	2083	0	0	0	0
DATABASE BLOCK CORRUPTION	80	8384	0	0	0	0
ACM OPERATION	104	64	4	0	0	0
FOREIGN ARCHIVED LOG	604	1002	0	0	0	0

Voyons à quoi correspondent les colonnes de ce tableau :

- TYPE : Type de la section
- RECORD_SIZE : Taille d'une entrée en bits
- RECORDS_TOTAL : Nombre d'entrées allouées
- RECORDS_USED : Nombres d'entrées utilisées
- FIRST_INDEX : Index de la première entrée
- LAST_INDEX : Index de la dernière entrée
- LAST_RECID : Id de la dernière entrée

Par ailleurs, cette vue nous permet de visualiser les valeurs des paramètres MAXLOGFILES, MAXINSTANCES, MAXDATAFILES et MAXLOGHISTORY.

TYPE	RECORDS_TOTAL	
REDO LOG	16	<- MAXLOGFILES (nombre maximum de groupes Redo Log)
REDO THREAD	8	<- MAXINSTANCES
LOG HISTORY	292	<- MAXLOGHISTORY
DATAFILE	100	<- MAXDATAFILES

Afin d'afficher la valeur du paramètre MAXLOGMEMBERS (nombre maximum de fichiers Redo Log dans un groupe Redo Log) il est possible d'utiliser la vue suivante : X\$KCCDI.

```
select dimlm from x$kccdi;
```

DIMLM

Aussi, dans notre base de données, nous avons au maximum, trois fichiers Redo Log par groupe.

Multiplexage des fichiers de contrôles

Multiplexage en utilisant le PFILE

Dans le but le multiplexer des fichiers de contrôle, nous devons uniquement copier le fichier qui nous intéresse et le copier à un emplacement différent et ensuite l'indiquer dans le paramètre d'initialisation CONTROL_FILES du PFILE.

Dans notre base exemple, nous disposons de trois fichiers de contrôle, que nous avons lister précédemment :

```
VALUE
-----
E:\APP\DTR\ORADATA\MIKATESTGOOD\CONTROL01.CTL,
E:\APP\DTR\ORADATA\MIKATESTGOOD\CONTROL02.CTL,
E:\APP\DTR\ORADATA\MIKATESTGOOD\CONTROL03.CTL
```

Partons du principe que nous voulons en créer un quatrième nommé CONTROL04.CTL, situé à un emplacement différent : C:\MESFICHIERSDECONTROLES\

Etape 1 : arrêt de la base

```
Shutdown immediate ;
```

Etape 2 : modification du paramètre CONTROL_FILES du PFILE :

Nous avons à l'heure actuelle les informations suivantes :

```
control_files=('E:\app\DTR\oradata\MikaTestGood\control01.ctl',
'E:\app\DTR\oradata\MikaTestGood\control02.ctl',
'E:\app\DTR\oradata\MikaTestGood\control03.ctl')
```

Il nous faut rajouter le nom et l'emplacement du nouveau fichier de contrôle :

```
control_files=('E:\app\DTR\oradata\MikaTestGood\control01.ctl',
'E:\app\DTR\oradata\MikaTestGood\control02.ctl',
'E:\app\DTR\oradata\MikaTestGood\control03.ctl', 'C:\MESFICHIERSDECONTROLES\CONTROL04.CTL')
```

Etape 3 : copie du fichier de contrôle

Il vous suffit de réaliser une copie d'un des fichiers de contrôle qui se trouvent à l'emplacement :

```
E:\app\DTR\oradata\MikaTestGood\
```

Vers :

C:\MESFICHIERSDECONTROLES\

En renommant le fichier par le nom souhaité: CONTROL04.CTL

Etape 4 : démarrage de la base de données

startup

Vous venez de multiplexer un fichier de contrôle en utilisant le PFILE. Cependant, il existe une autre méthode, en utilisant le SPFILE.

Multiplexage en utilisant le SPFILE

Etape 1 : ajout du nouveau fichier de contrôle dans le paramètre CONTROL_FILES du PFILE par la commande suivante :

```
ALTER SYSTEM SET CONTROL_FILES ==("E:\app\DTR\oradata\MikaTestGood\control01.ctl",  
"E:\app\DTR\oradata\MikaTestGood\control02.ctl",  
"E:\app\DTR\oradata\MikaTestGood\control03.ctl", "C:\MESFICHIERSDECONTROLES\CONTROL0  
4.CTL") SCOPE =SPFILE;
```

Il est intéressant de remarquer qu'avec cette méthode nous modifions le paramètre CONTROL_FILE alors que la base n'est pas arrêtée.

Etape 2 : arrêt de la base (pour que les changements puissent prendre effet)

Shutdown immediate ;

Les étapes 3 et 4 sont identiques à celles de la méthode par PFILE (copie du fichier de contrôle et démarrage de la base de données)

Sauvegarde d'un fichier de contrôle

Deux méthodes de sauvegarde sont possibles :

Soit la sauvegarde du fichier en fichier binaire :

```
ALTER DATABASE BACKUP CONTROLFILE TO votre_fichier ;
```

Soit sauvegarder le fichier en fichier texte dans le répertoire USER_DUMP_DEST si tué dans notre exemple à l'adresse suivante :

E:\app\DTR\admin\MikaTestGood\

Nous devons préciser que cette méthode (fichier texte) est intéressante car elle permet de modifier ce fichier pour reconstruire un nouveau fichier de contrôle.

Par défaut, Oracle recommande de réaliser une sauvegarde du fichier de contrôle lors de chaque modification de la structure de la base comme par exemple l'ajout, le renommage ou encore la suppression d'un fichier de données.

Création d'un fichier de contrôle

Plusieurs situations peuvent amener à la création d'un fichier de contrôle. En effet, si l'ensemble des fichiers de contrôles ont été perdus ou encore corrompus, si le nom de la base de données a changé, si des paramètres comme MAXDATAFILES, MAXLOGFILES ou encore MAXLOGHISTORY ont été modifiés mais encore lors d'un déplacement d'une base de données vers une autre machine et que les emplacements des datafiles et des fichiers Redo logs est différent des emplacements originaux.

Nous allons voir les étapes de création d'un fichier de contrôle :

Etape 1 : Connexion à la base avec les privilèges nécessaires

Vous devez être connecté à la base avec le privilège SYSDBA afin d'opérer les opérations suivantes.

Lancez SQL*Plus avec l'option /nolog

```
Sqlplus /nolog
```

Puis connecter en tant qu'utilisateur sys avec le privilège SYSDBA par cette commande :

```
Connect SYS AS SYSDBA
```

Etape 2 : lister l'ensemble des fichiers de données ainsi que les fichiers Redo log online

Pour cela nous allons utiliser les vues suivantes :

- V\$LOGFILE
- V\$DATAFILE
- V\$PARAMETER

Pour lister l'ensemble des fichiers Redo Log, on utilise la commande suivante :

```
SELECT MEMBER FROM V$LOGFILE;
```

On obtient donc pour notre base de données exemple :

```
MEMBER
-----
E:\APP\DTR\ORADATA\MIKATESTGOOD\REDO03.LOG
E:\APP\DTR\ORADATA\MIKATESTGOOD\REDO02.LOG
E:\APP\DTR\ORADATA\MIKATESTGOOD\REDO01.LOG
```

Nous avons donc trois fichiers Red Log.

Pour afficher la liste des fichiers de données (datafiles) associées aux espaces disque logique de la base, utilisez la commande suivante :

```
SELECT NAME FROM V$DATAFILE;
```

Pour notre base de données exemple nous obtenons le résultat suivant :

```
NAME
-----
E:\APP\DTR\ORADATA\MIKATESTGOOD\SYSTEM01.DBF
E:\APP\DTR\ORADATA\MIKATESTGOOD\SYSAUX01.DBF
E:\APP\DTR\ORADATA\MIKATESTGOOD\UNDOTBS01.DBF
E:\APP\DTR\ORADATA\MIKATESTGOOD\USERS01.DBF
E:\APP\DTR\ORADATA\MIKATESTGOOD\EXAMPLE01.DBF
E:\APP\DTR\ORADATA\MIKATESTGOOD\POPO01.DBF
```

Il y a six fichiers de données associés aux six Tablespace de la base.

Pour afficher la liste des fichiers de contrôle, nous utilisons la commande précitée :

```
select value from v$parameter where name='control_files';
```

Et nous obtenons le résultat suivant :

```
VALUE
-----
E:\APP\DTR\ORADATA\MIKATESTGOOD\CONTROL01.CTL,
E:\APP\DTR\ORADATA\MIKATESTGOOD\CONTROL02.CTL,
E:\APP\DTR\ORADATA\MIKATESTGOOD\CONTROL03.CTL
```

Etape 3 : arrêt de la base :

```
shutdown ;
```

Etape 4 : sauvegarder l'ensemble des fichiers de données et fichiers Red Log

Etape 5 : démarrage de la base mais sans la monter

```
Startup nomount
```

Etape 6 : création du fichier de contrôle en utilisant la commande CREATE CONTROL FILE

Note : si l'un des groupes de fichiers Redo Log a été perdu ou si la base a été renommée, il faut utiliser la clause RESETLOGS, sinon il suffit d'utiliser la clause NORESETLOGS. Cette dernière option fait qu'Oracle compare le numéro de groupe de Redo Log spécifié dans la commande CREATE CONTROL FILE avec celui qui est inscrit dans l'en-tête du fichier Redo Log.

Voici un exemple de création d'un fichier de contrôle pour notre base de données exemple :

```
CREATE CONTROLFILE SET DATABASE MIKATESTGOOD

LOGFILE

GROUP 1 ('E:\APP\DTR\ORADATA\MIKATESTGOOD\REDO01.LOG')
GROUP 2 ('E:\APP\DTR\ORADATA\MIKATESTGOOD\REDO02.LOG')
GROUP 3 ('E:\APP\DTR\ORADATA\MIKATESTGOOD\REDO03.LOG')

NORESETLOGS

DATAFILE
    'E:\APP\DTR\ORADATA\MIKATESTGOOD\SYSTEM01.DBF',
    'E:\APP\DTR\ORADATA\MIKATESTGOOD\SYSAUX01.DBF',
    'E:\APP\DTR\ORADATA\MIKATESTGOOD\UNDOTBS01.DBF',
    'E:\APP\DTR\ORADATA\MIKATESTGOOD\USERS01.DBF',
    'E:\APP\DTR\ORADATA\MIKATESTGOOD\EXAMPLE01.DBF',
    'E:\APP\DTR\ORADATA\MIKATESTGOOD\POPO01.DBF'

MAXLOGFILES 16

MAXLOGMEMBERS 3

MAXLOGHISTORY 292

MAXDATAFILES 100

MAXINSTANCES 8

NOARCHIVELOG
```

Etape7 : sauvegarder votre nouveau fichier de contrôle

Etape 8 : Modification du PFILE

Il vous faut modifier le paramètre CONTROL_FILES de votre PFILE en ajoutant le nouveau fichier de contrôle. Il est important de préciser que si le nom de la base a été changé, il ne faut pas oublier de modifier également le paramètre DB_NAME de ce même fichier.

Par la suite, si la base de données n'avait pas besoin d'être restaurée, vous pouvez ouvrir la base avec la commande :

```
ALTER DATABASE OPEN ;
```

Déplacement d'un fichier de contrôle

Nous avons, dans notre base de données exemple, les fichiers de contrôle suivants :

```
VALUE
-----
E:\APP\DTR\ORADATA\MIKATESTGOOD\CONTROL01.CTL,
E:\APP\DTR\ORADATA\MIKATESTGOOD\CONTROL02.CTL,
E:\APP\DTR\ORADATA\MIKATESTGOOD\CONTROL03.CTL
```

Partons du principe que nous désirons déplacer le fichier CONTROL03.CTL vers l'emplacement

```
C:\MESFICHIERSDECONTROLES\.
```

Suivez les étapes suivantes pour réaliser cette opération :

Etape 1 : arrêt de la base

```
Shutdown immediate
```

Etape 2 : déplacement du fichier de contrôle

Coupez et collez le fichier de contrôle CONTROL03.CTL au nouvel emplacement (C:\MESFICHIERSDECONTROLES\).

Etape 3 : modification du PFILE

Vous devez mettre à jour le paramètre CONTROL_FILES du PFILE :

```
CONTROL_FILES = (E:\APP\DTR\ORADATA\MIKATESTGOOD\CONTROL01.CTL,
E:\APP\DTR\ORADATA\MIKATESTGOOD\CONTROL02.CTL,
C:\MESFICHIERSDECONTROLES\CONTROL03.CTL)
```

Etape 4 : démarrage de la base

```
Startup
```

Votre fichier de contrôle a été déplacé.

Suppression d'un fichier de contrôle :

Nous avons, dans notre base de données exemple, les fichiers de contrôle suivants :

```
VALUE
-----
E:\APP\DTR\ORADATA\MIKATESTGOOD\CONTROL01.CTL,
E:\APP\DTR\ORADATA\MIKATESTGOOD\CONTROL02.CTL,
E:\APP\DTR\ORADATA\MIKATESTGOOD\CONTROL03.CTL
```

Partons du principe que nous désirons supprimer le fichier de contrôle CONTROL03.CTL.

Suivez les étapes suivantes pour réaliser cette opération :

Etape 1 : arrêt de la base

Shutdown immediate

Etape 2 : modification du PFILE

Vous devez mettre à jour le paramètre CONTROL_FILES du PFILE :

```
CONTROL_FILES = (E:\APP\DTR\ORADATA\MIKATESTGOOD\CONTROL01.CTL,
E:\APP\DTR\ORADATA\MIKATESTGOOD\CONTROL02.CTL)
```

Nous avons enlevé les informations concernant le fichier CONTROL03.CTL

Etape 3 : suppression du fichier de contrôle

Il vous faut également supprimer le fichier « physique » de contrôle au sein de votre système d'exploitation, à l'emplacement :

E:\APP\DTR\ORADATA\MIKATESTGOOD\

Etape 4 : démarrage de la base

Startup

Le fichier de contrôle CONTROL03.CTL a été correctement supprimé.

Restauration d'un fichier de contrôle à partir d'un autre fichier valide

Nous avons, toujours, dans notre base de données exemple, les fichiers de contrôle suivants :

```
VALUE
-----
E:\APP\DTR\ORADATA\MIKATESTGOOD\CONTROL01.CTL,
E:\APP\DTR\ORADATA\MIKATESTGOOD\CONTROL02.CTL,
E:\APP\DTR\ORADATA\MIKATESTGOOD\CONTROL03.CTL
```

Partons du principe que le fichier CONTROL03.CTL est corrompu.

Il faut arrêter la base avec l'instruction SHUTDOWN IMMEDIATE

Il suffit de copier l'un des deux autres fichiers de contrôle en le renommant CONTROL03 .CTL dans le même répertoire.

Redémarrer la base avec la commande STARTUP.

Le fichier de contrôle CONTROL03.CTL a été restauré.

Online Redo Log Files

Chaque base de données Oracle dispose d'un ensemble de deux ou plusieurs fichiers Redo Log. Un ensemble de fichier Redo Log est communément appelé Redo Log pour la base de données. Ce dernier est composé d'entrée Redo qui sont appelés redo records.

Le Redo Log stocke une copie des changements effectués dans la base de données. Si une panne requiert un fichier de données afin d'être restaurée à partir d'une sauvegarde et les changements récents sur les données ne sont pas présents sur le fichier de données restauré, ces changements peuvent être obtenus à partir du Redo Log. Aussi, le travail n'est jamais perdu.

Le Redo Log est utilisé afin de restaurer une base de données à la suite d'une panne hardware, software ou encore une panne de média.

Afin de protéger d'une panne impliquant le Redo Log lui-même, Oracle peut, comme les fichiers de contrôle, multiplexer le Redo Log. Ainsi, deux ou plusieurs copies identiques du Redo Log en activité (online redo log) peuvent être maintenues sur différents disques.

Le Redo Log est constitué de plusieurs groupes de fichiers Red Log. Un groupe est composé de fichiers Redo Log et de leurs copié multiplexées. Chaque copie identique est considérée comme un membre du groupe. Par ailleurs, chaque groupe est défini par un numéro, comme par exemple le Groupe 1.

L'objectif est de placer les membres d'un même groupe sur des disques différents. Plus précisément, il est intéressant de placer les différentes copies multiplexées sur des disques différents afin de pouvoir être efficace en cas de panne.

Le processus d'arrière plan de l'instance LGWR écrit les redo records du buffer mémoire vers un groupe Redo Log jusqu'à ce que les fichiers Redo Log de ce dernier aient atteint leur limite de

stockage, ou que vous ayez demandé changement de groupe. Ensuite, le processus écrit sur les fichiers du groupe suivant. Il procède de manière circulaire et de cette manière, les fichiers du groupe le plus récent sont réécrits par les redo record les plus récents. Cela permet de garder en mémoire le plus longtemps possible les changements qui ont été réalisés au sein de la base de données.

Archived Redo Log Files

Lorsque le Redo Log est archive, les fichiers Redo Log sont copiés vers un autre emplacement avec que de nouvelles informations soient écrites par-dessus (overwriting). Ces archives augmentent le nombre données Redo qui peuvent être sauvegardées et sont utilisées pour les restaurations. En effet, elles sont utilisées afin de restaurer une sauvegarde et d'appliquer l'ensemble des changements qui ont été réalisés sur la base entre la date de la sauvegarde et la date de la restauration.

Discussion autour des modes ARCHIVELOG et NOARCHIVELOG

L'archivage peut être réalisé uniquement si la base de données est lancée en mode ARCHIVELOG. Cela signifie qu'un groupe de fichiers Redo Log ne peut être réutilisé par le LGWR tant que le groupe est archivé. Par contre, si la base est lancée en mode NOARCHIVELOG, les groupes inactifs sont réutilisables immédiatement par le LGWR.

Aussi, le mode NOARCHIVELOG permet de protéger des pannes d'instance mais non des pannes de médias (de disque par exemple). Seuls les plus récents changements effectués sur la base de données, qui sont stockés les groupes Redo Log, sont disponible pour restaurer une instance. Cependant, pour restaurer une base de données en mode NOARCHIVELOG, il est uniquement possible d'utiliser les sauvegardes (backup) complètes de la base, réalisées lorsque la base est fermée.

Aussi, si vous décidez de créer une base en mode NOARCHIVELOG, il est nécessaire de réaliser des sauvegardes régulièrement.

L'archivage des Redo Log Files présente les avantages suivants :

- Une sauvegarde de la base de données, avec les fichiers online Redo logs et ceux archivés, garantit que vous pouvez restaurer l'ensemble des transactions réalisées sur la base dans le cas d'une panne du système ou d'un disque.
- Vous pouvez restaurer une base de données en utilisant une sauvegarde qui avait été réalisée pendant que la base était en activité (ouverte) et utilisée, aussi longtemps que vous disposez d'une copie des fichiers log archivés qui ont été écrits lors de la sauvegarde de la base.
- Vous pouvez également réaliser des sauvegardes en ligne de Espace disque logique (tablespace) et ainsi utiliser celles-ci pour restauration après une panne de média.

Avant de pouvoir archiver les fichiers Redo Log, vous devez spécifier l'emplacement dans le quel seront archivés ces derniers. Oracle recommande que les archives de logs soient stockés un espace de restauration rapide appelé Flash Recovery Area. Cette dernière est un emplacement dans lequel Oracle peut stocker et gérer les fichiers relatifs aux sauvegardes et restaurations. Il se distingue l'espace mémoire de la base de données (database area) qui est l'emplacement où sont stockés les fichiers courants de la base tels que les datafiles, fichiers de contrôle ou encore les fichiers online Redo log.

Log switchs et numéros de séquence du Redo Log

Un log switch correspond au moment à partir duquel la base cesse d'écrire dans l'un des fichiers Redo Log online et commence l'écriture dans un autre. D'ordinaire, un log switch intervient lorsque le fichier Redo log sur lequel la base écrit est plein. Cependant, il est possible de configurer les logs switch afin qu'ils interviennent de façon régulière sans que le fichier Red Log online ne soit forcément plein. Par ailleurs, il est également possible de forcer un log switch de façon manuelle.

A chaque fois qu'un log switch intervient, Oracle donne un nouveau numéro de séquence au nouveau fichier Redo Log sur lequel le processus LGWR va écrire. Lors de l'archivage, ce fichier garde ce numéro et donne par conséquent le prochain numéro de séquence disponible. Aussi, chaque fichier Redo Log online ou archivé est identifié par son numéro de séquence (log séquence). Par ailleurs, suite à une panne du système ou encore la restauration d'un média, Oracle utilise les fichiers Redo selon l'ordre croissant des numéros de séquence.

Gestion des Redo Log

Informations sur les Redo Logs

Afin d'afficher des informations sur les Redo Logs, nous allons utiliser les vues V\$THREAD, V\$LOG, V\$LOGFILE et V\$LOG_HISTORY.

La vue **V\$THREAD** permet de visualiser des informations sur le fichier Redo Log online. Commençons par une description de cette vue :

```
desc v$thread
```

Name	Null?	Type
-----	-----	-----
THREAD#		NUMBER
STATUS		VARCHAR2 (6)

ENABLED	VARCHAR2 (8)
GROUPS	NUMBER
INSTANCE	VARCHAR2 (80)
OPEN_TIME	DATE
CURRENT_GROUP#	NUMBER
SEQUENCE#	NUMBER
CHECKPOINT_CHANGE#	NUMBER
CHECKPOINT_TIME	DATE
ENABLE_CHANGE#	NUMBER
ENABLE_TIME	DATE
DISABLE_CHANGE#	NUMBER
DISABLE_TIME	DATE
LAST_REDO_SEQUENCE#	NUMBER
LAST_REDO_BLOCK	NUMBER
LAST_REDO_CHANGE#	NUMBER
LAST_REDO_TIME	DATE

Cette commande permet d’afficher les informations concernant le fichier Redo Log en cours

```
select * from v$thread;
```

THREAD#	STATUS	ENABLED	GROUPS	INSTANCE	OPEN_TIM	CURRENT_GROUP#	SEQUENCE#	CHECKPOINT_CHANGE#
1	OPEN	PUBLIC	3	mikatestgood	16/06/08	3	54	2343659

CHECKPOI	ENABLE_CHANGE#	ENABLE_T	DISABLE_CHANGE#	DISABLE_	LAST_REDO_SEQUENCE#	LAST_REDO_BLOCK	LAST_REDO_CHANGE#	LAST_RED
18/06/08	886308	13/06/08	0		54	23885	2353777	18/06/08

Dans cette table, on peut observer que le fichier Redo Log actif (online) appartient au groupe Redo Log 3, qu’il porte le numéro de séquence 54 ou encore qu’il a été ouvert le 16/08/06.

La vue **V\$LOG** a pour particularité de donner des informations en consultant non pas le dictionnaire de données, mais en lisant dans le fichier de contrôle.

Commençons tout d’abord par identifier les informations disponibles dans cette vue :

```
desc V$LOG
```

Name	Null?	Type
-----	-----	-----
GROUP#		NUMBER
THREAD#		NUMBER
SEQUENCE#		NUMBER
BYTES		NUMBER
MEMBERS		NUMBER
ARCHIVED		VARCHAR2 (3)
STATUS		VARCHAR2 (16)
FIRST_CHANGE#		NUMBER
FIRST_TIME		DATE

GROUP#	THREAD#	SEQUENCE#	BYTES	MEMBERS	ARC	STATUS	FIRST_CHANGE#	FIRST_TI
-----	-----	-----	-----	-----	---	-----	-----	-----
1	1	52	52428800	1	NO	INACTIVE	2266917	17/06/08
2	1	53	52428800	1	NO	INACTIVE	2304391	18/06/08
3	1	54	52428800	1	NO	CURRENT	2343659	18/06/08

Dans cette table, nous pouvons observer que la base de données dispose de trois groupes de Redo Log, constitués chacun d'un fichier Redo Log. Par ailleurs, on peut observer, qu'un seul fichier Redo Log est actif et que les autres sont inactifs, ce qui est normal. ARC signifie ARCHIVED et spécifie si le groupe a été archivé ou non (si l'archivage des fichiers Redo log a été activé).

La vue **V\$LOGFILE** est utilisée afin de visualiser les noms des membres des groupes Redo Log.

```
select * from v$logfile;
```

GROUP#	STATUS	TYPE	MEMBER	IS_RECOVERY_DEST_FILE
-----	-----	-----	-----	-----
3	ONLINE	ONLINE	E:\APP\DTR\ORADATA\MIKATESTGOOD\REDO03.LOG	NO
2	ONLINE	ONLINE	E:\APP\DTR\ORADATA\MIKATESTGOOD\REDO02.LOG	NO
1	ONLINE	ONLINE	E:\APP\DTR\ORADATA\MIKATESTGOOD\REDO01.LOG	NO

GROUP# correspond au numéro du groupe de Redo Log

STATUS peut prendre plusieurs valeurs : INVALID si le fichier n'est plus accessible, STALE si le fichier n'est complet, DELETED si il n'est plus utilisé. Par ailleurs, si le statut est vide, cela correspond au fait que le fichier est en cours d'utilisation.

MEMBER correspond au nom du fichier Red Log appartenant au groupe.

IS_RECOVERY_FILE peut prendre les valeurs YES ou NO. Cette colonne détermine si le fichier a été créé dans la flash recovery area ou non.

La vue **V\$LOG_HISTORY** permet de visualiser l'ensemble des informations contenues dans l'historique des fichiers Redo Log. Cette vue lit les informations sur le fichier de contrôle. A noter que la quantité maximale d'information que cette vue peut contenir dépend du paramètre d'initialisation MAXLOGHISTORY.

Donc, concernant notre base de données exemple :

```
select * from V$LOG_HISTORY;
```

RECID	STAMP	THREAD#	SEQUENCE#	FIRST_CHANGE#	FIRST_TI	NEXT_CHANGE#	RESETLOGS_CHANGE#	RESETLOG
1	657288871	1	1	886308	13/06/08	921442	886308	13/06/08
2	657289056	1	2	921442	13/06/08	933027	886308	13/06/08
3	657289484	1	3	933027	13/06/08	939289	886308	13/06/08
4	657289777	1	4	939289	13/06/08	944321	886308	13/06/08
5	657290081	1	5	944321	13/06/08	950018	886308	13/06/08
6	657290480	1	6	950018	13/06/08	954530	886308	13/06/08
7	657290938	1	7	954530	13/06/08	959092	886308	13/06/08
8	657291525	1	8	959092	13/06/08	964018	886308	13/06/08
9	657292182	1	9	964018	13/06/08	969427	886308	13/06/08
10	657292961	1	10	969427	13/06/08	975698	886308	13/06/08
11	657293801	1	11	975698	13/06/08	984378	886308	13/06/08

Création d'un groupe et des membres Redo

Création de groupes

Remarque préliminaire

Il est nécessaire, afin de créer un nouveau groupe Redo Log ou encore un membre de disposer du privilège système : ALTER DATABASE. Par ailleurs, la base de données peut disposer au maximum d'un nombre de groupes RedoLog égal à MAXLOGFILES.

Dans l'objectif de créer un nouveau groupe Redo, on utilise la commande ALTER DATABASE avec la clause ADD LOGFILE

Par exemple, si on l'on veut ajouter un groupe à notre base de données (cela est possible car MAXLOGFILES = 16 et nous ne disposons que deux trois groupes de Redo Log) :

```
ALTER DATABASE
ADD LOGFILE (' E:\APP\DTR\ORADATA\MIKATESTGOOD\ MonRedo01.log', '
E:\APP\DTR\ORADATA\MIKATESTGOOD\ MonRedo02.log') SIZE 500K;
```

On peut également donner le numéro du groupe en utilisant la clause GROUP :

```
ALTER DATABASE
ADD LOGFILE GROUP 4 (' E:\APP\DTR\ORADATA\MIKATESTGOOD\MonRedo01.log', '
E:\APP\DTR\ORADATA\MIKATESTGOOD\MonRedo02.log') SIZE 500K;
```

L'administration des groupes de fichiers Redo Log est facilitée par l'utilisation des numéros. Vous devez comprendre ces derniers entre 1 et MAXLOGFILES. Il est important également d'incrémenter les numéros de groupes de 1 en 1 afin d'éviter de consommer inutilement de l'espace dans les fichiers de contrôle.

Création de membres de groupes

Remarque préliminaire

Lors de la création d'un nouveau groupe Red Log, il est nécessaire de créer des membres pour ce groupe, d'un nombre variant de 1 à MAXLOGMEMBERS.

Par ailleurs, la création de membre de groupe Redo n'est pas induite uniquement par la création d'un nouveau groupe. En effet, il se peut, que suite à une panne système, des membres d'un groupe aient été supprimés. Par conséquent, la création de membres de groupe peut s'appliquer également à des groupes déjà existants.

Afin de créer un nouveau membre dans groupe Redo existant, la commande ALTER DATABASE est utilisée avec la clause ADD LOGFILE MEMBER.

Comme il a été explicité plus haut, la vue V\$LOGFILE est utilisée afin de visualiser les noms des membres des groupes Redo Log.

```
select * from v$logfile;
```

GROUP#	STATUS	TYPE	MEMBER	IS_RECOVERY_DEST_FILE
3	ONLINE		E:\APP\DTR\ORADATA\MIKATESTGOOD\REDO03.LOG	NO
2	ONLINE		E:\APP\DTR\ORADATA\MIKATESTGOOD\REDO02.LOG	NO
1	ONLINE		E:\APP\DTR\ORADATA\MIKATESTGOOD\REDO01.LOG	NO

Partons du principe que nous désirons rajouter un membre dans le groupe Redo 1.

```
ALTER DATABASE ADD LOGFILE MEMBER 'E:\APP\DTR\ORADATA\MIKATESTGOOD\REDO04.LOG' TO  
GROUP 1;
```

Il est indispensable d'indiquer le nom du fichier, cependant sa taille n'est pas nécessaire car cette dernière va être déterminée grâce à celles des autres fichiers du groupe.

Une autre méthode permet d'ajouter un membre à un groupe Redo Log. Toujours en utilisant la commande ALTER DATABASE, on spécifie le groupe cible par désignant l'ensemble des autres membres du groupe.

Partons du principe que le GROUP 1 dispose de deux membres Redo Log que sont :

```
E:\APP\DTR\ORADATA\MIKATESTGOOD\REDO01.LOG
```

```
E:\APP\DTR\ORADATA\MIKATESTGOOD\REDO04.LOG
```

Donc si l'on désire ajouter un troisième membre, que nous nommerons

```
E:\APP\DTR\ORADATA\MIKATESTGOOD\REDO05.LOG
```

Nous utilisons la commande suivante :

```
ALTER DATABASE ADD LOGFILE MEMBER 'E:\APP\DTR\ORADATA\MIKATESTGOOD\REDO05.LOG' TO  
( 'E:\APP\DTR\ORADATA\MIKATESTGOOD\REDO01.LOG' ,  
'E:\APP\DTR\ORADATA\MIKATESTGOOD\REDO04.LOG' ) ;
```

Nous spécifions donc le groupe 1 en désignant l'ensemble de ces membres dans la clause TO.

Déplacement et renommage des fichiers Redo Log

Ces procédures peuvent s'avérer nécessaires dans certaines situations. Par exemple, si le disque contenant les fichiers Redo Log actuellement utilisés doit être enlevé ou encore si les fichiers Redo Log ainsi que d'autres fichiers de données se trouvent sur le même disque et l'administrateur souhaite les séparer afin d'éviter une certaine contention.

Cette procédure se décompose en deux parties :

- Déplacement des fichiers Redo en utilisant le système d'exploitation

- Utilisation de la commande ALTER DATABASE pour donner à la base les nouveaux emplacements

Remarques

Afin de donner à la base les nouveaux emplacements des fichiers Redo Log, il faut disposer du privilège ALTER DATABASE, mais aussi, comme nous allons dans l'exemple qui suit, des privilèges systèmes afin de copier les fichiers dans le répertoire de destinations, mais aussi les privilèges nécessaires à l'ouverture et la sauvegarde d'une base de données.

Par ailleurs, il est important, avant de déplacer ces fichiers, de réaliser une sauvegarde complète de la base de données. De plus, après déplacement ou renommage des fichiers, il est judicieux de réaliser une sauvegarde du fichier de contrôle de la base.

Partons de l'exemple ci-dessous :

Nos fichiers Redo Log se trouvent sur un disque E: et dans l'objectif d'éviter la contention de données nous souhaitons déplacer les fichiers vers un disque B :.

Il est impératif de se connecter avec le privilège SYSDBA, pour disposer des privilèges nécessaires.

Comment faire ?

- Etape 1 : Arrêt de la base avec la commande `shutdown immediate`
- Etape 2 : copie des fichiers Redo Log vers le nouvel emplacement (Disque B :)

Donc sur le disque B :, nous avons désormais par exemple les fichiers suivants :

```
B:\MIKATESTGOOD\ MESREDOLOG \REDO03.LOG
B:\MIKATESTGOOD\ MESREDOLOG \REDO02.LOG
B:\MIKATESTGOOD\ MESREDOLOG \REDO01.LOG
```

- Etape 3 : démarrage de la base en mode mount mais sans l'ouvrir

```
STARTUP MOUNT
```

- Etape 4 : Donner les nouveaux emplacements à la base

```
ALTER DATABASE
RENAME FILE
'E:\APP\DTR\ORADATA\MIKATESTGOOD\REDO01.LOG' ,
'E:\APP\DTR\ORADATA\MIKATESTGOOD\REDO02.LOG' ,
'E:\APP\DTR\ORADATA\MIKATESTGOOD\REDO03.LOG'
TO
'B:\MIKATESTGOOD\ MESREDOLOG \REDO01.LOG' ,
' B:\MIKATESTGOOD\ MESREDOLOG \REDO02.LOG' ,
'B:\MIKATESTGOOD\ MESREDOLOG \REDO03.LOG' ;
```


- Etape 5 : Ouvrir la base, les modifications vont être prises en compte :

```
ALTER DATABASE OPEN ;
```

Suppression de groupes et membres Redo Log

Les groupes

Dans certaines situations, l'administrateur doit supprimer un groupe Redo Log pour, par exemple diminuer le nombre de groupes.

Cependant, avant toute opération de suppression d'un groupe Redo Log, il est important de prendre en compte certaines conditions :

Une instance Oracle doit disposer d'au minimum de deux groupes, quelque soit le nombre de membre dans chaque groupe.

La suppression d'un groupe Redo Log est possible uniquement si celui-ci est inactif. Par ailleurs, si l'administrateur désire supprimer le groupe actif, il doit au préalable forcer un log switch (cette opération est expliquée plus bas)

Vous pouvez utiliser la vue V\$LOG précitée afin d'afficher les informations relatives aux groupes :

```
SELECT GROUP#, ARCHIVED, STATUS FROM V$LOG;
```

```
GROUP# ARC STATUS
----- --
1 NO INACTIVE
2 NO INACTIVE
3 NO CURRENT
```

La suppression d'un groupe est réalisée par la commande ALTER DATABASE avec la clause DROP LOGFILE.

Prenons l'exemple où nous souhaitons supprimer le groupe 1.

```
ALTER DATABASE DROP LOGFILE GROUP 1 ;
```

Il vous faut ensuite supprimer les fichiers Redo Log de ce dernier groupe en utilisant les commandes de votre système d'exploitation.

Les membres

Dans certaines situations, l'administrateur doit supprimer un ou plusieurs membres Redo Log. Par exemple, si certains membres sont situés sur un disque défectueux.

La suppression d'un membre inactif, la commande ALTER DATABASE est utilisée avec la clause DROP LOGFILE MEMBER.

Par exemple, nous voulons supprimer le membre E:\APP\DTR\ORADATA\MIKATESTGOOD\REDO05.LOG que nous avons ajouté au groupe 1 précédemment, nous utilisons la commande suivante :

```
ALTER DATABASE DROP LOGFILE MEMBER 'E:\APP\DTR\ORADATA\MIKATESTGOOD\REDO05.LOG'
```

Il vous suffit ensuite d'effacer le fichier Redo Log du disque en utilisant les commandes du système d'exploitation.

Note : si vous désirez supprimer un membre d'un groupe qui est actif, vous devez au préalable forcer le log switch.

Forcer les log switch

Afin de réaliser cette opération, l'utilisateur doit disposer du privilège ALTER SYSTEM. La commande permettant de réaliser cette opération est ALTER SYSTEM avec la clause SWITCH LOGFILE.

Prenons un exemple sur notre base de données :

On vérifie les statuts

```
SELECT GROUP#, ARCHIVED, STATUS FROM V$LOG;
```

```
GROUP# ARC STATUS
----- --
1 NO INACTIVE
2 NO INACTIVE
3 NO CURRENT
```

Ensuite on force le log switch par la commande suivante :

```
ALTER SYSTEM SWITCH LOGFILE ;
```

On vérifie à nouveau les statuts :

```

GROUP# ARC STATUS
-----
1 NO CURRENT
2 NO INACTIVE
3 NO INACTIVE

```

Le nouveau groupe courant est donc désormais le groupe 1.

Rollback Segments

Les segments Rollback étaient utilisés pour suivre les informations undo pour les bases de données relatives aux versions précédentes d'Oracle. Désormais, il est préféré la gestion des informations undo par une Tablespace Undo. Ces notions seront explicitées plus tard.

Fichiers de données

Les fichiers de données sont les fichiers du système d'exploitation qui stockent les données dans la base de données. Les données sont écrites dans ces fichiers dans un format propriétaire d'Oracle par conséquent elles ne peuvent être lues par d'autres applications. Tempfiles est une classe spéciale de fichiers de données qui sont associés uniquement avec les espaces disque logique temporaire.

Les fichiers de données peuvent être répartis selon différents types :

- Segment

Un segment contient un type spécifique d'objet de la base de données. Par exemple, une table est stockée dans un segment de table et un index est stocké dans un segment d'index. Un fichier de données peut contenir plusieurs segments.

- Extent

Un extent est un ensemble contigu de blocs de données dans un segment. Oracle alloue de l'espace mémoire pour les segments au sein des extents qui le composent. Lorsque les extents d'un segment sont pleins, la base de données alloue un autre extent pour ce segment.

- DataBlock

Un bloc de données, aussi appelé bloc de base de données (database block) est la plus petite unité de stockage pour la base de données. Un extent est constitué de plusieurs blocs de données contigus. La base de données utilise par défaut une taille de bloc de données à la création de la base. D'ailleurs, après que celle-ci ait été créée, de modifier la taille des blocs de données. Cependant, il est tout de même possible de créer un espace disque logique avec une taille de bloc de données différente de celle par défaut.

Espace disques logiques (Tablespace)

Une base de données est divisée en unités de stockage logique qui sont appelées espaces disque logiques (Tablespace en anglais), qui regroupent les structures logiques reliées telles que les tables, les vues, et autres objets de la base de données. Par exemple, tous les objets d'application peuvent être regroupés dans le même espace disque logique dans le but de simplifier les opérations de maintenance.

Un espace disque logique est composé d'un ou plusieurs fichiers de données. Les objets de la base de données qui sont assignés à un espace disque logique sont stockés dans les fichiers de données de celui-ci.

Par ailleurs, un espace disque logique vous permet de localiser physiquement les données de la base de données. En effet, lors de la création d'un espace disque logique, vous devez spécifier un emplacement (répertoire) de stockage pour les fichiers de données. Aussi, tous les objets d'un schéma sont placés dans ce répertoire physique.

Quelques espaces logiques qui sont inclus dans une base de données :

EXAMPLE : cet espace logique contient les schémas exemples qui sont inclus avec Oracle. La plupart des documentations officielles sont basées sur ces exemples

SYSTEM : cet espace logique est automatiquement créé lors de la création d'une base de données. Oracle l'utilise afin de gérer la base. Il contient le dictionnaire de données mais aussi d'autres tables et vues qui contiennent des informations concernant l'administration de la base de données. Ces dernières sont incluses sans le schéma SYS et peuvent être accédées uniquement par l'utilisateur SYS ou encore un autre utilisateur avec les privilèges requis.

SYSAUX : c'est un espace disque logique auxiliaire à SYSTEM. Certains composants et produits qui utilisaient l'espace disque logique SYSTEM dans des versions d'Oracle inférieure à 10g utilisent désormais cet espace disque logique. L'utilisation de SYSAUX réduit le lancement de l'espace disque logique SYSTEM et réduit également la maintenance. En effet, étant donné qu'il y a plusieurs espaces disque logique pour la maintenance. Il est important de préciser que toutes les bases de données issues de version antérieure à Oracle 10g doivent disposer d'un espace disque logique SYSAUX.

TEMP : il stocke temporairement les données générées lorsque que les requêtes SQL sont exécutées. Par exemple, cet espace disque logique est utilisé lors d'une opération de tri. Chaque base de données doit disposer d'un espace disque logique temporaire qui est assigné aux utilisateurs comme leur espace disque logique temporaire. Il est défini comme l'espace disque logique temporaire par défaut. Lors de la création d'un compte utilisateur, si aucun espace logique temporaire n'est spécifié, TEMP sera assigné à l'utilisateur.

UNDOTBS1 : Il s'agit de l'espace disque logique utilisé par la base de données afin de stocker les informations Undo. Chaque base de données doit avoir ce type d'espace disque logique.

USERS : cet espace disque logique est utilisé afin de stocker les objets des utilisateurs permanents ainsi que leurs données. Egalement, chaque base de données doit disposer de ce type de tablespace. En effet, en cas d'absence, les objets des utilisateurs seront créés dans l'espace disque logique SYSTEM, ce qui est fortement déconseillé. Dans une base de données préconfigurée, USERS est désigné comme l'espace disque logique par défaut pour les nouveaux utilisateurs.

Types d'espace disque logique

Permanent : ce type est utilisé pour stocker des données permanentes comme par exemple les données systèmes, utilisateurs ou encore données d'application. Il est assigné à chaque utilisateur un espace disque logique permanent (par défaut USER)

Undo : ce type est utilisé pour créer un espace disque logique de type Undo. Bien que vous puissiez créer plus d'un espace disque logique Undo, uniquement un seul se doit d'être actif.

Temporary : ce type est utilisé pour les espaces disque logique temporaire comme celui par défaut TEMP. Il est possible de créer d'autres espaces disque logique temporaire que celui par défaut. Par exemple, si vous désirez créer un autre groupe complet d'espaces disque logique. Les fichiers physiques qui composent une tablespace temporaire sont appelés tempfiles, contrairement aux datafiles pour les autres tablespace.

Statuts d'espace disque logique

Vous pouvez définir un espace disque logique selon les statuts suivants : Read/ Write, Read Only ou Offline (pour éventuellement réaliser des opérations de maintenance).

Espaces disques logique localement gérés

Deux types de gestion peuvent être identifiés : la gestion par dictionnaire ou local.

Lorsqu'un espace disque logique est géré par le dictionnaire, ce dernier demande à Oracle de stocker les informations inhérentes à l'allocation d'espace dans le dictionnaire des données. Cette action a pour conséquence d'induire une charge supplémentaire pour toutes les opérations possibles sur les objets d'un espace disque logique.

Lorsqu'un espace disque logique est géré localement (locally managed), ce dernier stocke l'ensemble des informations d'allocation d'espace en entête de l'espace disque logique. Cette méthode présente des avantages tels que le fait d'éviter les contentions sur le dictionnaire de données, la simplification de la gestion d'espace dans un espace disque logique qui est automatisée. Il est important de préciser que lors de la création de la base de données, le type de l'espace disque logique SYSTEM a un impact déterminant sur les autres. En effet, si ce dernier est LOCALLY MANAGED, les autres espaces disques logiques de la base devront l'être aussi.

Il existe deux méthodes afin de gérer les extents dans ce dernier mode : UNIFORM SIZE et AUTOALLOCATE.

Le premier demande à Oracle de créer des extents de taille identique alors que le deuxième impose à Oracle de créer des extents de taille de plus en plus grande en fonction du nombre d'extents créés. Cette méthode ne peut pas être appliquée pour un espace disque logique temporaire.

La première méthode permet un suivi simple de l'allocation des extents. Le deuxième a pour avantage de permettre au DBA de mieux gérer les systèmes qu'il maîtrise difficilement dans le sens où il n'est pas maître des objets qui sont créés dans la base.

Par ailleurs, en termes de performance (utilisation des ressources systèmes), Oracle recommande d'allouer des extents de taille relativement importante plutôt que de nombreux extents de taille faible.

Gestion des espaces disque logique

Création d'un Tablespace : Instruction CREATE TABLESPACE

Voici avant tout quelques clauses relatives à cette instruction :

- **LOGGING**: permet d'indiquer si la création d'objet dans l'espace disque logique doit faire l'objet d'une inscription dans les redo logs.
- **FORCE LOGGING**: permet de forcer le LOGGING même sur les objets ayant l'option NOLOGGING.
Cette option est invalide pour les espaces disque logique de type TEMPORARY ou UNDO.
- **ONLINE ou OFFLINE**: détermine la disponibilité de l'espace disque logique après sa création.
- **PERMANENT** : permet de créer un espace disque logique permanent.

- **TEMPORARY**: permet de créer un espace disque logique temporaire géré par le dictionnaire. Afin de créer un espace disque logique géré localement, la commande CREATE TEMPORARY TABLESPACE doit être utilisée.
- **LOCAL ou DICTIONARY** : détermine le type de gestion de l'espace disque logique.
- **AUTOEXTEND** : cette clause permet que lorsque l'espace disque logique a atteint sa taille maximale, il peut s'étendre de lui-même d'une taille qui est fixée par l'administrateur.

Dans cet exemple, on créé un espace disque logique d'une taille de 10Mo nommé MonEspaceDisqueLogique, géré localement. La gestion des extents est en mode UNIFORM et leur taille est 128Ko.

```
CREATE TABLESPACE MonEspaceDisqueLogique
DATAFILE 'e:\app\DTR\oradata\DataBaseTest\MonTableSpace01.dbf' SIZE 10M
EXTENT MANAGEMENT LOCAL UNIFORM SIZE 128K;
```

Prenons l'exemple suivant, plus complexe, qui provient de partie création de base de données sans le DBCA :

```
CREATE TABLESPACE POPO LOGGING
DATAFILE 'e:\app\DTR\oradata\MikaTestGood\popo01.dbf'
SIZE 500M REUSE AUTOEXTEND ON NEXT 1280K MAXSIZE UNLIMITED
EXTENT MANAGEMENT LOCAL;

CREATE TABLESPACE apps_tbs LOGGING
DATAFILE 'E:\app\DTR\oradata\DataBaseTest\apps01.dbf'
SIZE 500M REUSE AUTOEXTEND ON NEXT 1280K MAXSIZE UNLIMITED
EXTENT MANAGEMENT LOCAL;
```

Ce Tablespace est nommé apps_tbs et sa taille est de 500Mo. Il peut être réutilisé (REUSE) et s'il atteint sa taille maximum, il s'étendra de 1280Ko. Sa taille maximum est illimitée. Par ailleurs, il est géré localement.

Nous voulons créer un Tablespace Undo :

```
CREATE UNDO TABLESPACE undotbs
DATAFILE 'E:\app\DTR\oradata\DataBaseTest\undotbs.dbf'
SIZE 20M AUTOEXTEND ON;
```

Modification d'un tablespace : instruction ALTER TABLESPACE

Cette instruction permet de modifier un espace disque logique au travers de l'ensemble des clauses qui sont spécifiées lors de la création.

Prenons l'exemple suivant :

Nous voulons ajouter un datafile à un espace disque logique :

```
ALTER TABLESPACE POPO  
ADD DATAFILE 'e:\app\DTR\oradata\MikaTestGood\popo02.dbf' SIZE 20M;
```

Et pour la retirer :

```
ALTER TABLESPACE POPO  
DROP DATAFILE 'e:\app\DTR\oradata\MikaTestGood\popo02.dbf';
```

Nous voulons diminuer la taille du tablespace précédent :

Pour ce faire nous avons besoin de changer la taille du datafile associé. Cela concerne la base de données avant l'espace disque logique. Par conséquent, nous utilisons la commande ALTER DATABASE

```
ALTER DATABASE DATAFILE 'E:\app\DTR\oradata\MikaTestGood\popo01.dbf' resize 300M ;
```

Nous voulons activer l'option AUTOEXTEND du Tablespace :

```
ALTER DATABASE DATAFILE 'e:\app\DTR\oradata\MikaTestGood\popo01.dbf' AUTOEXTEND ON NEXT 10M;
```

Nous voulons changer le statut de l'espace disque logique :

```
ALTER TABLESPACE POPO OFFLINE ;
```

Cette commande peut être utile pour éviter que les données de l'espace disque logique soient accessibles lors d'une maintenance.

Suppression d'un Tablespace : instruction DROP TABLESPACE

Voici avant tout quelques clauses relatives à cette instruction :

INCLUDING CONTENTS : cette option permet de supprimer le contenu des l'espace disque logique. A noter que si cette option n'est pas activée, il est impossible de supprimer des espaces disques logique contenant des objets.

AND DATAFILES : cette clause permet de supprimer également les fichiers physiques associés.

CASCADE CONSTRAINTS : cette clause donne la possibilité de supprimer les contraintes d'intégrité référentielles vers des objets d'autres espace disque logique, ainsi que les contraintes uniques de l'espace disque logique supprimé.

L'exemple suivant permet de supprimer l'espace disque logique ainsi que ses contraintes et ses contenu :


```
DROP TABLESPACE POPO INCLUDING CONTENTS CASCADE CONSTRAINTS;
```

Visualisation des Tablespace :

La requête suivante permet d'afficher l'identifiant, le nom, le statut et le nom des fichiers de données d'un tablespace:

```
select
    FILE_ID,
    FILE_NAME,
    STATUS,
    TABLESPACE_NAME
from
    DBA_DATA_FILES
order by FILE_ID;
```

Ce qui donne le résultat suivant :

FILE_ID	FILE_NAME	STATUS	TABLESPACE_NAME
1	E:\APP\DTR\ORADATA\MIKATESTGOOD\SYSTEM01.DBF	AVAILABLE	SYSTEM
2	E:\APP\DTR\ORADATA\MIKATESTGOOD\SYSAUX01.DBF	AVAILABLE	SYSAUX
3	E:\APP\DTR\ORADATA\MIKATESTGOOD\UNDOTBS01.DBF	AVAILABLE	UNDOTBS1
4	E:\APP\DTR\ORADATA\MIKATESTGOOD\USERS01.DBF	AVAILABLE	USERS
5	E:\APP\DTR\ORADATA\MIKATESTGOOD\EXAMPLE01.DBF	AVAILABLE	EXAMPLE
6	E:\APP\DTR\ORADATA\MIKATESTGOOD\POPO01.DBF	AVAILABLE	POPO

Cette commande permet d'afficher les noms des tablespace, la quantité de mémoire allouée, utilisée et libre :

```
SELECT A.tablespace_Name, A.Alloue, B.Occupe, C.Libre
FROM (select tablespace_name, sum(bytes)/1024/1024 AS ALLOUE from
dba_data_files group by tablespace_name) a,
(select tablespace_name, Sum(bytes)/1024/1024 AS OCCUPE from
dba_segments group by tablespace_name) b,
(select tablespace_name, Sum(bytes)/1024/1024 AS LIBRE from
dba_free_space group by tablespace_name) c
WHERE B.tablespace_Name = A.tablespace_Name
AND C.Tablespace_Name = B.Tablespace_Name;
```

TABLESPACE_NAME	ALLOUE	OCCUPE	LIBRE
SYSAUX	716,6875	681,75	34,875
UNDOTBS1	185	16,25	168,6875
USERS	5	4,9375	1,125
SYSTEM	700	691	8,9375
EXAMPLE	100	77,3125	22,625

Données UNDO

Lorsqu'une transaction modifie des données de la base, Oracle réalise au préalable une copie des données avant la modification. Cette dernière est appelée données Undo. La sauvegarde de ces informations est nécessaire pour les raisons suivantes :

- Afin d'annuler les changements, qui n'ont pas fait l'objet de commit, sur la base de données dans le cas d'une opération de Roll Back. Cette dernière peut être soit le fait qu'un utilisateur décide d'annuler les changements effectués sur la base lors d'une transaction erronée soit faire partie d'une opération de restauration.
- Afin d'obtenir une consistance de lecture ce qui signifie que chaque utilisateur peut avoir une vue consistante des données de la base. Par exemple, si un utilisateur lance une requête à 10h du matin et cette dernière opère pendant 15 minutes, les résultats vont refléter l'état des données à 10h, sans prendre en compte les éventuelles opérations des autres utilisateurs telles que des UPDATE ou INSERT sur les données lors de l'exécution de la requête.

Le paramètre d'initialisation lié à la gestion Undo est le suivant :

UNDO_MANAGEMENT : si sa valeur est AUTO ou nulle, alors la gestion automatique des Undo est activée. A contrario, si sa valeur est MANUAL, Oracle enclenche le mode manuel de gestion des Undo.

Espace disque logique Undo (Tablespace Undo)

Dans le cas d'une gestion automatique d'annulation (UNDO), les données Undo sont stockées dans un espace disque logique Undo. Ce type de tablespace dispose de propriétés supplémentaires par rapport à un Tablespace standard. Il est possible de disposer de plusieurs espaces disque logique Undo mais seulement un seul peut être actif pour une instance Oracle.

Lorsqu'une base de données est créée via le DBCA, ce dernier crée un tablespace Undo autoextensible nommé UNDOTBS1, qui peut s'étendre jusqu'à une taille de 35Go.

La commande SQL permettant de créer un tablespace UNDO est la suivante :

```
CREATE UNDO TABLESPACE nomTablespace  
DATAFILE 'emplacement du datafile'
```

Période de rétention Undo

Oracle s'assure que les données Undo correspondantes à une transaction en cours ne soient pas écrasées tant que la transaction n'a pas fait l'objet d'un commit (période de rétention). Par contre, après ce dernier, l'espace mémoire utilisé par ces données peut être soit réutilisé soit écrasé.

Dans ce dernier cas, ces données peuvent être écrasées si l'espace mémoire du tablespace devient insuffisante.

Par défaut, Oracle optimise de façon automatique la période de rétention Undo sur la base de la taille du tablespace Undo et de l'activité du système. Cependant, il est possible de définir une période minimum de rétention en définissant le paramètre d'initialisation UNDO_RETENTION dans le PFILE. Cette définition a pour impact les éléments suivants :

Si le tablespace Undo dispose d'une taille fixe, la définition du paramètre précédent n'a aucun impact. Oracle optimise toujours de façon automatique la période de rétention.

Pour un tablespace Undo qui dispose de l'option AUTOEXTEND, la base de données tente d'honorer la période minimum de rétention spécifiée dans le paramètre UNDO_RETENTION. Lorsque que l'espace mémoire dans l'espace disque logique est faible, ce dernier s'étend de façon automatique. Si la clause MAXSIZE est spécifiée pour un tablespace auto-extensible, lorsque la taille maximum a été atteinte, la base de données commence à écraser les données Undo qui ne sont pas expirées.

Gestion des données Undo

Informations sur les données Undo

Les vues V\$UNDOSTAT, V\$ROLLSTAT, V\$TRANSACTION permettent d'obtenir des informations sur les fichiers Undo.

V\$UNDOSTAT : cette vue contient des statistiques pour la visualisation et l'optimisation d'un espace Undo. Vous pouvez l'utiliser pour estimer la quantité d'espace Undo requise pour les tâches qui sont effectuées sur la base. Par ailleurs, la base de données utilise ces statistiques afin d'optimiser l'usage des Undo dans le système. Il est important de préciser que cette vue est significative lorsque la gestion Undo est réalisée de façon automatique.

V\$ROLLSTAT : cette vue est également utilisée dans le cadre d'une gestion Undo automatique. Elle reflète les comportements des segments Undo dans le tablespace.

V\$TRANSACTION : cette vue contient des informations concernant les segments Undo.

DBA_UNDO_EXTENTS : elle contient le statut et la taille de chaque extent du tablespace Undo.

DBA_HIST_UNDOSTAT : cette vue contient des aperçus des informations contenues dans V\$UNDOSTAT.

Suivi des périodes de rétention optimisées

Il est possible d'obtenir la période de rétention courante en visualisant la colonne TUNED_UNDORETENTION de la vue V\$UNDOSTAT. Cette vue contient une ligne pour chaque collection de statistiques (collectées à intervalles de 10 minutes) des quatre précédents jours (au-delà de quatre jours, les informations sont disponibles la vue DBA_HIST_UNDOSTAT).

La période de rétention est donnée en secondes.

Cette requête SQL, issue de la documentation Oracle, permet d'afficher les informations :

```
select to_char(begin_time, 'DD-MON-RR HH24:MI') begin_time,
to_char(end_time, 'DD-MON-RR HH24:MI') end_time, tuned_undoretention
from v$undostat order by end_time;
```

BEGIN_TIME	END_TIME	TUNED_UNDORETENTION
-----	-----	-----
18-JUIN -08 11:13	18-JUIN -08 11:23	1859
18-JUIN -08 11:23	18-JUIN -08 11:33	1262
18-JUIN -08 11:33	18-JUIN -08 11:43	1870
18-JUIN -08 11:43	18-JUIN -08 11:53	1212
18-JUIN -08 11:53	18-JUIN -08 12:03	1819
18-JUIN -08 12:03	18-JUIN -08 12:13	1222
18-JUIN -08 12:13	18-JUIN -08 12:23	1829
18-JUIN -08 12:23	18-JUIN -08 12:33	1232
18-JUIN -08 12:33	18-JUIN -08 12:43	1839
18-JUIN -08 12:43	18-JUIN -08 12:53	1242
18-JUIN -08 12:53	18-JUIN -08 13:03	1849

D'après ces informations, la période de rétention est en moyenne de 1566 secondes donc environ 26 minutes.

Définition de la période de rétention Undo minimum

Comme il a été explicité précédemment, il est possible de définir cette période en donnant une valeur au paramètre d'initialisation UNDO_RETENTION.

Pour ce faire vous avez le choix entre deux méthodes :

Changer la valeur de UNDO_RETENTION dans le PFILE

Utiliser, à n'importe quel moment, la commande ALTER SYSTEM :

```
ALTER SYSTEM SET UNDO_RETENTION = 1800 ;
```

Le changement du paramètre UNDO_RETENTION est immédiate mais la période de rétention ne peut être honorée uniquement s'il y a assez d'espace mémoire dans le tablespace Undo.

Quelques exemples :

Création d'un tablespace undo

```
CREATE UNDO TABLESPACE UNDOTABLESPACE  
  
DATAFILE E:\APP\DTR\ORADATA\MIKATESTGOOD\montablespaceundo.dbf' size 100M;
```

Changement du tablespace Undo utilise par la base :

Nous devons modifier le paramètre d'initialisation UNDO_TABLESPACE contenu dans le PFILE. Ce paramètre contient le nom du tablespace UNDO utilisé.

```
ALTER SYSTEM  
SET UNDO_TABLESPACE =UNDOTABLESPACE ;
```

La base de données peut être ouverte pendant que l'opération de changement (switch) est réalisée. De plus, les transactions des utilisateurs sont également réalisées. Lorsque l'opération de changement a été réalisée avec succès, l'ensemble des transactions débutées après le lancement de la commande de changement sont assignés dans les tables de transactions dans le nouveau tablespace.

L'opération de changement n'attend pas sur les transactions de l'ancien tablespace Undo afin de réaliser des commit. En effet, s'il y a des transactions en attente dans l'ancien tablespace Undo, ce dernier entre dans le mode PENDING OFFLINE (statut). Dans ce statut, les transactions existantes continuent d'être exécutées mais les enregistrements undo des transactions des nouveaux utilisateurs ne sont plus stockés dans l'ancien tablespace.

Un espace disque logique Undo peut se retrouver dans le mode PENDING OFFLINE même avant que l'opération de changement n'ait été réalisée. Par ailleurs, un tablespace dans ce mode ne peut être utilisé par une autre instance Oracle ni même effacée.

Après que toutes les transactions aient fait l'objet d'un commit, le tablespace passe du mode PENDING OFFLINE au mode OFFLINE. Ainsi, cet espace disque logique est éventuellement disponible pour les autres instances Oracle.

Modification du tablespace utilisé par la base

Partons du principe que l'on désire rajouter un fichier de données au tablespace UNDOTABLESPACE.

```
ALTER TABLESPACE UNDOTABLESPACE  
ADD DATAFILE ' E:\APP\DTR\ORADATA\MIKATESTGOOD\montablespaceundo2.dbf' SIZE 100M;
```

Nous avons donc rajouté un fichier de données au tablespace UNDOTABLESPACE,

Il est important de préciser que vous également utiliser les commandes ALTER DATABASE pour modifier la taille d'un fichier de données.

GESTION DES UTILISATEURS

Informations sur les utilisateurs

Les vues qui permettent d'avoir des informations sur les utilisateurs sont DBA_USERS et DBA_TS_QUOTAS. De plus, les vues USER_USERS et USER_TS_QUOTAS permettent d'avoir des informations sur l'utilisateur courant.

Descriptif de la vue DBA_USERS

- USERNAME : login de l'utilisateur
- USER_ID : identifiant de l'utilisateur
- PASSWORD : mot de passe encrypté de l'utilisateur
- ACCOUNT_STATUS : statut du compte (OPEN : ouvert, EXPIRED : l'utilisateur doit changer son mot de passe, LOCKED (TIMED) : compte verrouillé pour une période données, LOCKED : compte verrouillé de manière définitive)
- LOCK_DATE : date à laquelle le compte a été verrouillé.
- EXPIRY_DATE : date à laquelle le mot de passe doit être changé.
- DEFAULT_TABLESPACE : tablespace par défaut de l'utilisateur
- TEMPORARY_TABLESPACE : tablespace temporaire de l'utilisateur
- CREATED : date de création de l'utilisateur
- PROFILE : nom du profil qui a été assigné à l'utilisateur

Par exemple, pour avoir la liste des utilisateurs ainsi que leur statut, nous utilisons la commande suivante :

```
SELECT USERNAME, ACCOUNT_STATUS FROM DBA_USERS ;
```

Descriptif de la vue DBA_TS_QUOTAS

- TABLESPACE_NAME : nom de l'espace disque logique
- USERNAME : nom de l'utilisateur concerné
- BYTES : quantité d'octets qui sont alloués pour l'utilisateur sur le tablespace.
- MAX_BYTES : quota alloué sur le tablespace. (Note : ce champ prend la valeur -1 si il n'y pas de quota (UNLIMITED))
- BLOCKS : nombre de blocs qui sont alloués pour l'utilisateur sur l'espace disque logique.
- MAX_BLOCKS : nombre de blocs maximum que l'utilisateur peut posséder sur le tablespace.

Par exemple, affichons la liste des utilisateurs avec leur tablespace et quota associés :

```
SELECT TABLESPACE_NAME, USERNAME, MAX_BYTES FROM DBA_TS_QUOTAS;
```

Création d'utilisateur : CREATE USER

Nous allons voir dans ce chapitre les méthodes qui permettent la création et la configuration d'un utilisateur d'une base de données, compte avec lequel ce dernier pourra se connecter et réaliser des actions en fonction des droits et privilèges qui lui sont accordés.

Par ailleurs, il est intéressant de préciser que , sous Oracle, la création d'un utilisateur implique de prime à bord que celui-ci ne dispose d'aucun droits, ce qui est, a contrario, pas le cas dans d'autres bases de données. La gestion des privilèges sera abordée dans le chapitre suivant.

Notion de schéma

Un schéma est un ensemble d'objets Oracle tels que des tables, indexes, clusters et d'autres encore qui sont associés à un utilisateur bien précis. Lors de la création d'un utilisateur, un schéma est créé également et lui est associé. Un utilisateur ne peut être associé à qu'un seul schéma, la réciproque étant vraie.

Notion de user

Concrètement parlant, un utilisateur correspond à un nom de login auquel sont associés des privilèges. Il est stocké dans le dictionnaire de données et un espace de stockage lui est alloué pour ses objets Oracle qui seront stockés dans le schéma qui est associé à l'utilisateur.

Dans l'environnement Oracle, un utilisateur peut être assimilé à son schéma.

Choix d'un nom d'utilisateur

La première étape de création d'un utilisateur réside dans le fait de lui assigner un nom d'utilisateur ou encore login. A ce propos, il est intéressant, dans une optique de gestion des ajouts d'utilisateurs, de mettre en place une politique de nommage.

Par exemple, il est possible d'utiliser uniquement la première lettre du prénom et de la concaténer au nom de famille.

Jacques Dupont -> jdupont

Par ailleurs, il faut prendre connaissance des restrictions relatives aux login sous Oracle.

- Le nombre de caractères du login ne doit pas excéder 30
- Le login peut contenir les lettres [a-z] et les chiffres [0-9] ainsi que les caractères `_`, `$`, `#`. De façon commune, les caractères accentués sont à proscrire.
- Le login doit commencer par une lettre uniquement. Cependant, dans le cas de l'utilisation unique de chiffre pour les noms de login, il faudra faire précéder les chiffres par le caractère `'`.

Note : il faut attacher une importance toute particulière à l'utilisation du caractère `"` avec les logins, car ceux-ci deviennent sensibles à la casse.

Aussi, pour reprendre notre exemple précédent,

JDUPONT sera différent de jdupont.

Authentification de l'utilisateur

L'utilisateur est identifié par un mot de passe qui est stocké dans la base de données. Cet élément induit que le fait que la base de données soit ouverte est nécessaire afin qu'il puisse s'identifier et se connecter.

La clause IDENTIFIED BY <mot de passe> est utilisée pour définir le mot de passe d'un utilisateur.

Les mots de passe sous Oracle, comme les login, sont soumis à des restrictions :

- Il doit commencer par une lettre
- Sa taille est fixée à un maximum de 30 caractères
- Il peut être composé de chiffres, lettres de l'alphabet et des symboles suivants : `_`, `#`, `$` (les deux derniers étant recommandés par Oracle).

Par exemple, on décide de créer l'utilisateur jdupont, identifié par le mot de passe MonMotDePasse

```
CREATE USER jdupont IDENTIFIED BY MonMotDePasse
```

Utilisateurs et Tablespace

En termes de sécurité de la base de données, il est important de restreindre le domaine d'action de l'utilisateur en choisissant les espace disque logique que ce dernier va utiliser.

Comme il a été précité, il est fortement déconseillé de permettre à un utilisateur d'utiliser le tablespace SYSTEM. Aussi, est-il important d'attribuer à l'utilisateur d'autres tablespace.

Cette étape est nécessaire dans le sens où l'administrateur va définir l'espace disque logique de données et l'espace disque logique temporaire que l'utilisateur va utiliser. En effet, Oracle assigne par défaut le tablespace SYSTEM à l'utilisateur si aucun espace disque logique n'est défini pour ce dernier.

Définition de quota.

Après avoir choisi les tablespace que l'utilisateur va utiliser, il est possible de définir l'espace qui lui est alloué sur chaque espace disque logique.

Il existe deux options possibles pour les quotas, à savoir la taille (en Ko ou Mo) et l'option Unlimited.

Par ailleurs, il est important de préciser que chaque nouvel utilisateur dispose d'un quota null sur l'ensemble des tablespace, donc il est nécessaire de le définir.

La commande CREATE USER

Commençons par un exemple :

```
CREATE USER Mika
  IDENTIFIED BY mikapwd
  DEFAULT TABLESPACE USERS
  QUOTA 20M ON USERS
  TEMPORARY TABLESPACE TEMP
  QUOTA 5M ON TEMP
  PASSWORD EXPIRE;
```

A cette commande est associé un ensemble de clauses que nous allons expliciter ci-dessous :

IDENTIFIED BY password :

Permet de déterminer le mot de passe de l'utilisateur

DEFAULT TABLESPACE nom de l'espace disque logique :

Assigne un tablespace par défaut à l'utilisateur

TEMPORARY TABLESPACE nom de l'espace disque logique :

Assigne un tablespace temporaire à l'utilisateur

QUOTA options ON nom de l'espace disque logique :

Définition du quota d'espace assigné à l'utilisateur sur un tablespace.

PROFILE nom du profil :

Attribue un profil qui a pour objectif de limiter les ressources système de l'utilisateur.

PASSWORD EXPIRE :

Cette clause fait expirer le mot de passe de l'utilisateur dans le but que ce dernier le change dès sa première connexion.

ACCOUNT LOCK / UNLOCK :

Active ou désactive un compte utilisateur.

Aussi, dans l'exemple ci-dessus, l'utilisateur a pour nom de login Mika et mot de passe mikapwd. Son tablespace par défaut est USER, sur lequel il dispose d'un quota de 20Mo et son tablespace temporaire est TEMP sur lequel il dispose d'un espace mémoire de 5 Mo. Par ailleurs, il devra changer son mot de passe lors de sa première connexion.

Par la suite il est nécessaire d'assigner à l'utilisateur des privilèges, cela sera explicité dans le chapitre suivant.

Modification d'un utilisateur : ALTER USER

Modification du mot de passe

Il est possible de modifier le mot de passe d'un utilisateur en utilisant la commande suivante :

```
ALTER USER Mika IDENTIFIED BY nouveaumotdepasse ;
```

Ainsi, l'utilisateur que nous avons créé précédemment, est désormais identifié par le mot de passe : *nouveaumotdepasse*.

Modification des quotas

Partons du principe que nous voulons changer la quantité d'espace assignée à l'utilisateur Mika sur les tablespaces USERS et TEMP.

```
ALTER USER Mika QUOTA 30M ON USERS QUOTA 10M ON TEMP ;
```

L'utilisateur dispose désormais d'un quota de 30 Mo sur le tablespace USERS et d'un quota de 10 Mo sur le tablespace temporaire TEMP.

Par ailleurs, il est important de préciser que si un utilisateur dispose d'un quota de 5 Mo sur un tablespace et qu'il a créé des tables sur ce dernier, si l'administrateur change son quota à 0 Mo, ce n'induit pas le vidage des tables pour autant. Ce la signifie uniquement que ses tables ne seront plus en mesure d'allouer des extents additionnels.

Modification du tablespace par défaut

Partons du principe que l'administrateur veut changer les tablespace qui sont assignées à l'utilisateur Mika :

```
ALTER USER Mika DEFAULT TABLESPACE USERS2 TEMPORARY TABLESPACE TEMP2 ;
```

Mika a désormais USERS2 comme tablespace par défaut, et TEMP2 comme tablespace temporaire.

Modification d'un statut

Dans certaines circonstances, il peut être nécessaire de verrouiller un compte utilisateur, afin que ce dernier ne puisse pas être utilisé de manière frauduleuse.

Pour verrouiller le compte de l'utilisateur Mika, par exemple, on utilise la commande suivante :

```
ALTER USER Mika ACCOUNT LOCK ;
```

Pour le déverrouiller :

```
ALTER USER Mika ACCOUNT UNLOCK;
```

Par ailleurs, il est possible d'utiliser la commande ALTER USER avec l'ensemble des clauses spécifiées dans la partie traitant la commande CREATE USER.

Suppression d'un utilisateur : commande DROP USER

Nous allons voir comment supprimer un utilisateur ainsi que son schéma. Il est important de préciser qu'un utilisateur connecté à la base ne peut pas être supprimé.

```
drop user mop;

drop user mop

*

ERROR at line 1:

ORA-01940: impossible de supprimer un utilisateur qui est connecté
```

Suppression d'un utilisateur avec un schéma vide

Il s'agit du cas où l'utilisateur existe mais il n'a pas créé d'objets Oracle. Aussi, pour le supprimer, il faut utiliser la commande DROP USER.

Si son schéma n'est pas vide, on obtient le message d'erreur suivant :

```
DROP USER John;

DROP USER John
*
ERREUR à la ligne 1 :
ORA-01922: CASCADE à spécifier pour supprimer 'John'
```

Supprimons, par exemple l'utilisateur Mika :

```
DROP USER Mika ;

Utilisateur supprimé.
```

Suppression d'un utilisateur ainsi que son schéma

Afin de supprimer un utilisateur ainsi que l'ensemble des objets contenus dans son schéma, il faut utiliser la clause CASCADE de la commande DROP USER.

Par ailleurs, si le schéma contient des objets de type table, Oracle va effacer l'ensemble des contraintes d'intégrité ainsi que toutes les contraintes d'intégrité contenues dans les schémas des autres utilisateurs qui font référence aux contraintes UNIQUE et PRIMARY KEY du schéma que l'on souhaite supprimer.

De plus, Oracle efface également l'ensemble des indexes liés aux colonnes des tables, l'ensemble des triggers et types de données.

Oracle invalide mais ne supprime pas les objets des autres schémas qui font référence au schéma qui va être supprimé.

Cependant, lors de l'exécution de la requête DROP USER avec la clause CASCADE, Oracle ne supprime pas les rôles qui ont été créés par l'utilisateur.

Par exemple, nous souhaitons supprimer l'utilisateur Mika et son schéma associé :

```
DROP USER Mika CASCADE ;
```

GESTION DES RÔLES ET PRIVILEGES

L'objectif principal des rôles et des privilèges sont de mettre en place une politique de sécurité d'accès aux données de la base. En effet, ces deux notions protègent les données en accordant (ou retirant) des privilèges à un utilisateur précis ou bien un groupe d'utilisateurs.

La notion de rôle peut être définie par un regroupement de privilèges.

Parmi les privilèges, nous pouvons distinguer deux types différents :

- Les privilèges système, qui accordent les droits de création, modification, suppression ou encore exécution d'ensembles d'objets.
Par exemple, le privilège CREATE TABLE, permet à l'utilisateur de créer des tables.
- Les privilèges objet, qui accordent les droits de manipulations sur des objets.

Par exemple, les privilèges SELECT et INSERT accordés sur la table Mika.TableTest accordent les droits de sélection et d'insertion de lignes dans la table (appartenant à l'utilisateur Mika) à l'utilisateur qui les a reçus.

Assignation de privilèges système

Lors de la création d'un utilisateur avec la commande CREATE USER, ce dernier ne dispose d'aucuns droits de par le fait qu'aucun privilège ne lui a été accordé. Par exemple, il ne peut même pas se connecter à la base de données. Aussi, est-il nécessaire de lui attribuer des privilèges.

Pour ce faire, il faut utiliser la commande GRANT qui a la syntaxe suivante :

```
GRANT <privilège système><rôle><ALLPRIVILEGES> TO <utilisateur> <role> <PUBLIC>
```

rôle : correspond à un rôle qui a été défini au préalable.

ALL PRIVILEGES : symbolise l'ensemble des privilèges systèmes excepté SELECT ANY DICTIONARY. Il est important de préciser que cette option accorde des droits quasi illimités à l'utilisateur et donc peut induire des risques de sécurité au niveau de la base de données.

PUBLIC : accordent les privilèges spécifiés à l'ensemble des utilisateurs

Afin qu'un utilisateur puisse se connecter à la base de données, le privilège système CREATE SESSION doit lui être accordé.

Par exemple, accordons ce privilège à l'utilisateur Mika :

```
GRANT CREATE SESSION TO Mika ;
```

Par la suite, l'administrateur doit lui accorder le droit de création de table :

```
GRANT CREATE TABLE TO Mika ;
```

Nous pouvons également lui assigner le droit de création de vues :

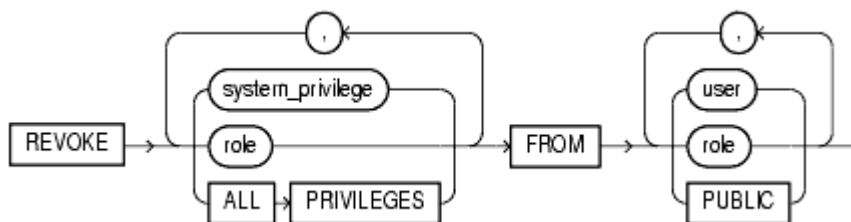
```
GRANT CREATE VIEW TO Mika ;
```

L'ensemble de ces commandes peut être regroupé comme dans l'exemple suivant :

```
GRANT CREATE SESSION, CREATE TABLE, CREATE VIEW TO Mika ;
```

Retrait de privilèges système

La commande REVOKE permet de retirer les privilèges accordés à un utilisateur, ou encore à un rôle. La syntaxe de cette commande est la suite :



Il est aisé de remarquer que cette commande possède des arguments similaires à la commande GRANT.

Afin de supprimer un privilège, il est nécessaire de disposer du droit induit par la clause ADMIN OPTIONS.

Le rôle DBA permet à l'utilisateur de retirer l'ensemble des privilèges systèmes (ou rôles). Il en est de même des rôles CONNECT, RESOURCE et DBA. Cependant, l'utilisateur disposant de ce rôle ne peut pas retirer les privilèges qu'il n'a pas attribués lui-même.

Remarque importante : lorsque des privilèges sont retirés à un utilisateur, son schéma ainsi que les objets qu'il contient ne sont pas supprimés.

Assignation de privilèges objets

La commande GRANT est toujours utilisée dans ce cas, mais de façon différente :

```
GRANT <privilège_objet><ALL PRIVILEGES> ON <schéma><.><objet> TO  
<utilisateur><rôle><PUBLIC><WITH GRANT OPTION><WITH HIERARCHY OPTION>
```

privilège_objet : correspond à un privilège objet

ALL PRIVILEGES : correspond à l'ensemble des privilèges qui sont accordés à l'utilisateur qui exécute l'instruction

Schéma : représente le nom d'un schéma

Objet : correspond au nom d'un objet du schéma sélectionné.

WITH GRANT OPTIONS : permet à un utilisateur d'assigner à son tour les privilèges qu'il a reçu.

WITH HIERARCHY OPTION : accorde les privilèges à l'ensemble des sous-objets

Par exemple, si l'on souhaite accorder à l'utilisateur Mika les droits de sélection, insertion, modification et suppression sur les lignes d'une table exemple nommée TableExemple qui appartient à autre utilisateur que nous nommerons John, il suffit d'exécuter la commande suivante :

```
GRANT SELECT, INSERT, UPDATE, DELETE ON John.TableExemple TO Mika ;
```

Il est important de préciser qu'afin de supprimer ou mettre à jour les lignes d'une table, les privilèges DELETE et UPDATE ne sont pas suffisants. En effet, il est nécessaire d'accorder le privilège SELECT.

Par ailleurs, un utilisateur ne peut pas attribuer de droits sur objet qui ne lui appartient pas, même s'il dispose des droits DBA.

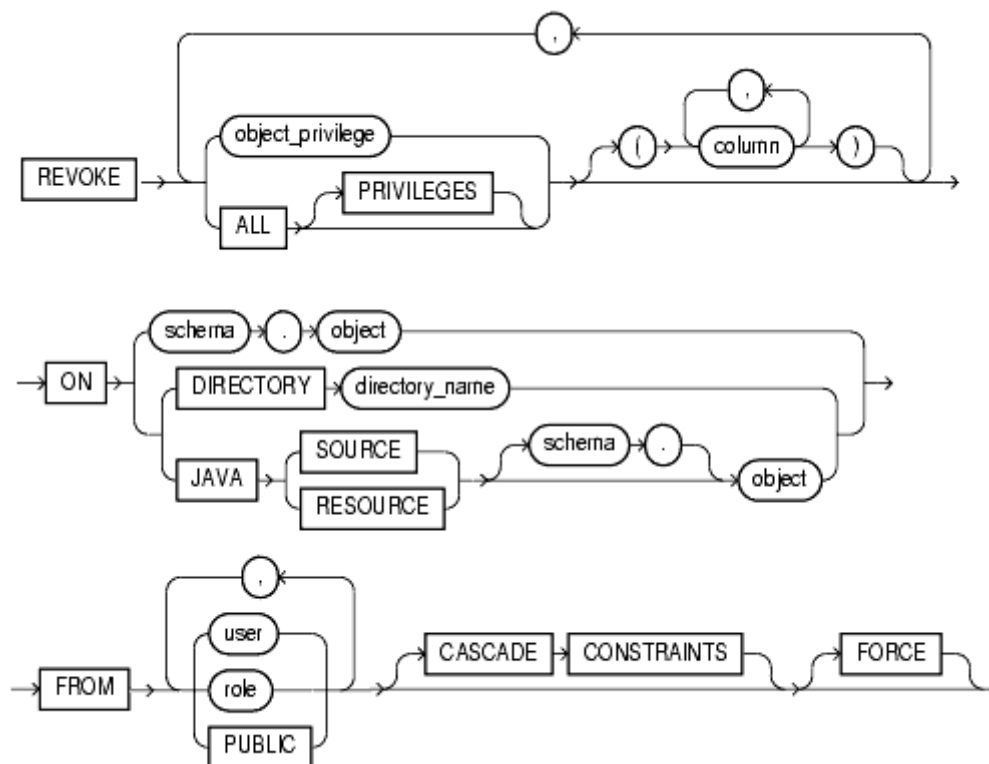
Il est intéressant de mettre en exergue quelques principes relatifs aux privilèges :

- Un utilisateur dispose de l'ensemble des droits possibles sur les objets qu'il possède.
- Un utilisateur peut donner les droits qu'il a reçu, mais uniquement ceux-ci.

- Un utilisateur ayant reçu le privilège WITH GRANT OPTION peut à son tour l'accorder à un autre utilisateur.

Retrait de privilèges objet

Tout comme les privilèges système, les privilèges objets peuvent être révoqués par l'utilisation de la commande REVOKE. Cependant, sa syntaxe, appliquée aux privilèges objet est légèrement plus complexe :



CASCADE CONSTRAINTS : cette option permet de supprimer l'ensemble des contraintes d'intégrités référentielles déterminées par l'utilisateur. Il est important de préciser que cette option ne s'applique uniquement sur la suppression de privilège de type REFERENCES.

FORCE : cette option permet de révoquer le privilège EXECUTE.

Création de rôles

Dans le cas où la liste des privilèges à attribuer à des utilisateurs est importante, l'utilisation de l'instruction GRANT devient rapidement fastidieuse.

Prenons l'exemple d'une promotion d'étudiants suivant des cours de base de données. Ils doivent réaliser des travaux pratiques sur une base de données et par conséquent ils doivent bénéficier des certain nombre de privilèges, identiques pour chaque étudiant.

Il est intéressant de regrouper l'ensemble de ces privilèges dans un ensemble afin de faciliter leur attribution. Nous arrivons donc à la notion de rôle.

L'instruction permettant de créer un rôle est la suivante : CREATE ROLE avec cette syntaxe :

```
CREATE ROLE <nomrôle>
```

Avec un ensemble de clauses :

NOT IDENTIFIED : cette clause stipule qu'il n'est pas nécessaire de taper un mot de passe pour activer ce rôle.

IDENTIFIED BY motdepasse : un mot de passe est nécessaire pour activer le rôle

Partons de l'exemple précité. Nous souhaitons créer un rôle Etudiants qui permet d'attribuer les droits de sélection, insertion, modification et suppression sur des tables exemples (TABEX1, TABEX2, TABEX3) appartenant à l'utilisateur PROFESSEUR.

Nous devons tout d'abord créer le rôle, sans identification par mot de passe (ce n'est pas nécessaire) :

```
CREATE ROLE Etudiant ;
```

Cependant, ce rôle est vide, il faut donc le remplir avec des commandes GRANT :

```
GRANT SELECT, INSERT, UPDATE, DELETE ON PROFESSEUR.TABEX1 TO Etudiant;  
GRANT SELECT, INSERT, UPDATE, DELETE ON PROFESSEUR.TABEX2 TO Etudiant;  
GRANT SELECT, INSERT, UPDATE, DELETE ON PROFESSEUR.TABEX3 TO Etudiant;
```

Le rôle est désormais opérationnel. Nous pouvons donc l'attribuer à un étudiant nommé par exemple John :

```
GRANT Etudiant TO John ;
```

Sous Oracle, il existe trois rôles prédéfinis :

- CONNECT
- RESOURCE
- DBA

La vue **DBA_SYS_PRIVS** permet de visualiser les droits assignés à un rôle.

Donc pour CONNECT, nous avons :

```
SELECT * FROM DBA_SYS_PRIVS WHERE grantee='CONNECT' ;
```

GRANTEE	PRIVILEGE	ADM
CONNECT	CREATE SESSION	NO

Ce rôle permet uniquement l'ouverture d'une session.

Si l'option ADMIN OPTION a la valeur NO, ce la signifie que le privilège associé ne peut pas être accordé à un autre rôle ou encore à un autre utilisateur.

Regardons maintenant les privilèges systèmes qui composent le rôle RESOURCE :

```
SELECT * FROM DBA_SYS_PRIVS WHERE grantee='RESOURCE' ;
```

GRANTEE	PRIVILEGE	ADM
RESOURCE	CREATE TRIGGER	NO
RESOURCE	CREATE SEQUENCE	NO
RESOURCE	CREATE TYPE	NO
RESOURCE	CREATE PROCEDURE	NO
RESOURCE	CREATE CLUSTER	NO
RESOURCE	CREATE OPERATOR	NO
RESOURCE	CREATE INDEXTYPE	NO
RESOURCE	CREATE TABLE	NO

Ce rôle accorde les droits de création de triggers, de séquence, de type, de procédures, de cluster, de opérateur, de type d'indexes, de tables.

Cependant, il est important de préciser que le rôle RESOURCE octroie un privilège UNLIMITED QUOTA aux utilisateurs. Il est donc à utiliser et assigner avec précautions.

Concernant le rôle, il n'est pas vraiment pertinent de lister l'ensemble des privilèges (un très grand nombre !). Cependant, ce rôle regroupe tous les privilèges système pour la gestion des utilisateurs et de leurs tables

Généralement, il est déconseillé d'attribuer ces rôles par défaut aux utilisateurs car ils leurs accordent trop de droits. Il est préférable de créer des rôles « sur-mesure ».

Informations relatives aux rôles

Il est possible de lister l'ensemble des rôles en utilisant la vue DBA_ROLES ;

```
select * from DBA_ROLES ;
```

Comme nous l'avons vu précédemment, la liste des privilèges système que contient un rôle est visible par les vues **DBA_SYS_PRIVS** et **USER_SYS_PRIVS**.

Par ailleurs, il peut être intéressant de visualiser les rôles attribués à un utilisateur. Pour ce faire, nous utilisons les vues **DBA_ROLE_PRIVS** et **USER_ROLE_PRIVS**.

```
select * from DBA_ROLE_PRIVS where grantee = <nom_utilisateur> ;
```

Pour lister l'ensemble des privileges objets accordés à un utilisateur, il faut utiliser les vues **DBA_TAB_PRIVS**, **ALL_TAB_PRIVS** et **USER_TAB_PRIVS**.

Durant une session :

Afin de lister l'ensemble des rôles accordés à un utilisateur au cours de sa session, il faut utiliser la vue **SESSION_ROLES**, et pour afficher, dans le même cas, ses privilèges, on utilise la vue **SESSION_PRIVS**.

Remarque importante :

Si un rôle a été attribué à un utilisateur, ce dernier ne lui permet pas de créer des vues, des procédures, des fonctions de packages ni même de clés étrangères au travers de code PL/SQL dynamique.

Pour permettre la création de ces objets par l'intermédiaire de PL/SQL, l'utilisateur doit disposer de droits qui lui ont été attribués par l'instruction GRANT.

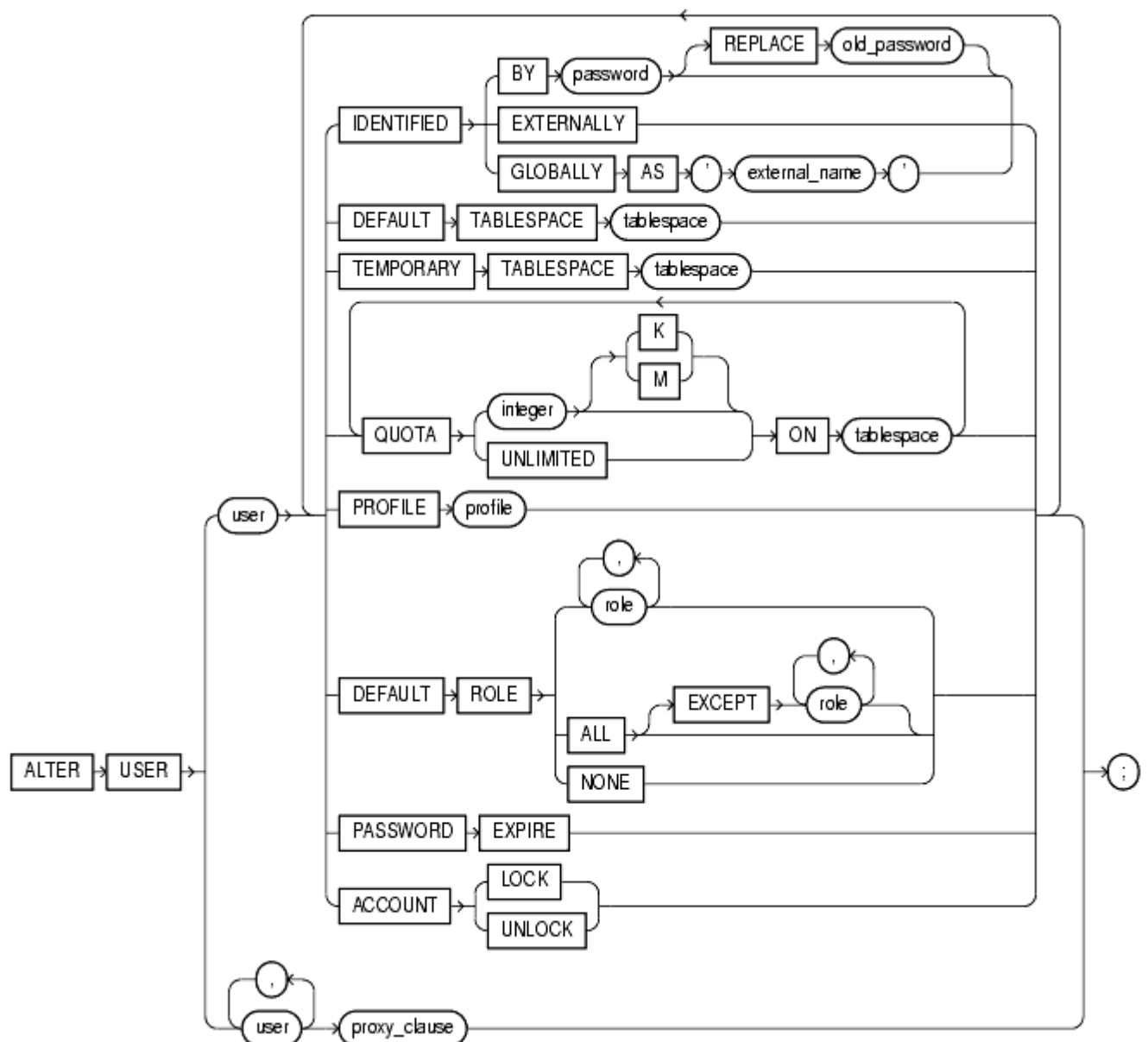
Modification de rôle

La commande ALTER ROLE permet de modifier un rôle. Par exemple, nous voulons désormais que le rôle Etudiant dispose d'un mot de passe pour pouvoir l'activer :

```
ALTER ROLE Etudiant IDENTIFIED BY MotDePasse ;
```

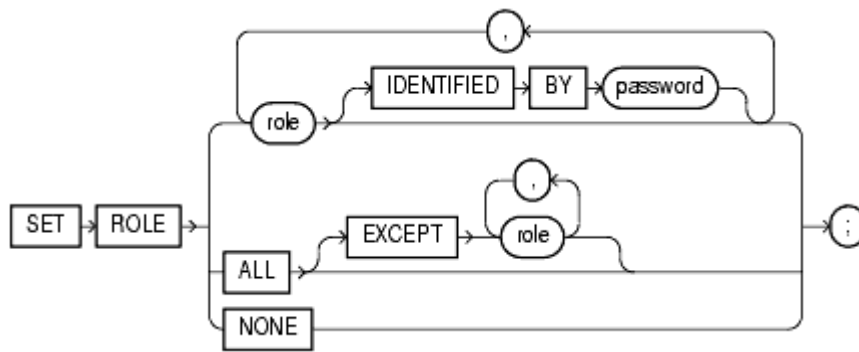
Activation et désactivation de rôles

La commande ALTER USER est utilisée pour activer (désactiver) un rôle. Le schéma suivant nous montre sa syntaxe.



- DEFAULT ROLE indique les rôles accordés par défaut à la connexion
- ALL cette clause assigne l'ensemble des rôles
- ALL EXCEPT assigne tous les rôles à l'exception des rôles spécifiés
- NONE n'accorde aucun rôle.

Par ailleurs, afin d'activer un rôle qui n'est pas prédéfini, l'instruction SET ROLE doit être utilisée. Sa syntaxe est la suivante :



Donc pour activer le rôle Etudiant que nous avons précédemment créer, il suffit de taper la commande suivante :

```
SET ROLE Etudiant ;
```

Suppression d'un rôle

Un rôle peut être supprimé en utilisant la commande suivante : DROP ROLE ;

Par exemple :

```
DROP ROLE Etudiant ;
```

Ainsi, le rôle étudiant et l'ensemble des privilèges qui le constituent sont effacés de la base. Ce rôle est aussi retiré à tous les utilisateurs concernés.

Liste des privilèges système

Privilège système	Description
CLUSTERS	
CREATE CLUSTER	Création de clusters dans le schéma
CREATE ANY CLUSTER	Création de clusters dans n'importe quel schéma
ALTER ANY CLUSTER	Modification de clusters dans n'importe quel schéma
DROP ANY CLUSTER	Suppression de clusters dans n'importe quel schéma
CONTEXTS	
CREATE ANY CONTEXT	Créations de tous context
DROP ANY CONTEXT	Suppression de tous context
DATABASE	
ALTER DATABASE	Modification de la base
ALTER SYSTEM	Autorise l'instruction ALTER SYSTEM
AUDIT SYSTEM	Autorise les instructions AUDIT sql_statements
DATABASE LINKS:	
CREATE DATABASE LINK	Création de liens de base de données dans le schéma
CREATE PUBLIC DATABASE LINK	Création de liens de base de données pour le groupe PUBLIC
DROP PUBLIC DATABASE LINK	Suppression de liens de base de données pour le groupe PUBLIC
DEBUGGING	
DEBUG CONNECT SESSION	Connexion dans la session en cours au débbugger utilisant le protocole Java Debug Wire Protocol (JDWP)
DEBUG ANY PROCEDURE	Deboguage de tout code PL/SQL ou Java dans n'importe quel objet de la base
DIMENSIONS	

CREATE DIMENSION	Création de dimensions dans le schéma
CREATE ANY DIMENSION	Création de dimensions dans n'importe quel schéma
ALTER ANY DIMENSION	Modification de dimensions dans n'importe quel schéma
DROP ANY DIMENSION	Suppression de dimensions dans n'importe quel schéma
DIRECTORIES	
CREATE ANY DIRECTORY	Création d'objets DIRECTORY
DROP ANY DIRECTORY	Suppression d'objets DIRECTORY
INDEXTYPES	
CREATE INDEXTYPE	Création d'indextype dans le schéma
CREATE ANY INDEXTYPE	Création d'indextype dans n'importe quel schéma
ALTER ANY INDEXTYPE	Modification d'indextype dans n'importe quel schéma
DROP ANY INDEXTYPE	Suppression d'indextype dans n'importe quel schéma
EXECUTE ANY INDEXTYPE	Référence un indextype dans n'importe quel schéma
INDEXES	
CREATE ANY INDEX	Création d'index dans n'importe quel schéma
ALTER ANY INDEX	Modification d'index dans n'importe quel schéma
DROP ANY INDEX	Suppression d'index dans n'importe quel schéma
QUERY REWRITE	Autorise la réécriture via une vue matérialisée dans le schéma
GLOBAL QUERY REWRITE	Autorise la réécriture via une vue matérialisée dans n'importe quel schéma
LIBRARIES	
CREATE LIBRARY	Création de bibliothèques de fonctions/procédures externes dans le schéma
CREATE ANY LIBRARY	Création de bibliothèques de fonctions/procédures externes dans n'importe quel schéma
DROP ANY LIBRARY	Suppression de bibliothèques de fonctions/procédures externes dans n'importe quel schéma

MATERIALIZED VIEWS	
CREATE MATERIALIZED VIEW	Création de vues matérialisées dans le schéma
CREATE ANY MATERIALIZED VIEW	Création de vues matérialisées dans n'importe quel schéma
ALTER ANY MATERIALIZED VIEW	Modification de vues matérialisées dans n'importe quel schéma
DROP ANY MATERIALIZED VIEW	Suppression de vues matérialisées dans n'importe quel schéma
QUERY REWRITE	Autorise la réécriture via une vue matérialisée dans le schéma
GLOBAL QUERY REWRITE	Autorise la réécriture via une vue matérialisée dans n'importe quel schéma
ON COMMIT REFRESH	Création de vues matérialisées de type refresh-on-commit sur n'importe quelle table
FLASHBACK ANY TABLE	Autorise une requête FlashBack sur n'importe quelle table, vue ou vue matérialisée dans n'importe quel schéma
OPERATORS	
CREATE OPERATOR	Création d'un opérateur dans le schéma
CREATE ANY OPERATOR	Création d'un opérateur dans n'importe quel schéma
DROP ANY OPERATOR	Suppression d'un opérateur dans n'importe quel schéma
EXECUTE ANY OPERATOR	Référence un opérateur dans n'importe quel schéma
OUTLINES	
CREATE ANY OUTLINE	Création d' outlines publics
ALTER ANY OUTLINE	Modification d' outlines publics
DROP ANY OUTLINE	Suppression d' outlines publics
PROCEDURES	
CREATE PROCEDURE	Création de fonctions, procédures et packages dans le schéma
CREATE ANY PROCEDURE	Création de fonctions, procédures et packages dans n'importe quel schéma

ALTER ANY PROCEDURE	Modification de fonctions, procédures et packages dans n'importe quel schéma
DROP ANY PROCEDURE	Suppression de fonctions, procédures et packages dans n'importe quel schéma
EXECUTE ANY PROCEDURE	Exécution de fonctions, procédures et package de n'importe quel schéma
PROFILS	
CREATE PROFILE	Création de profils
ALTER PROFILE	Modification de profils
DROP PROFILE	Suppression de profils
ROLES	
CREATE ROLE	Création de rôles
ALTER ANY ROLE	Modification de n'importe quel rôle
DROP ANY ROLE	Suppression de n'importe quel rôle
GRANT ANY ROLE	Assignation de n'importe quel rôle
ROLLBACK SEGMENTS	
CREATE ROLLBACK SEGMENT	Création de segments de rollback
ALTER ROLLBACK SEGMENT	Modification de segments de rollback
DROP ROLLBACK SEGMENT	Suppression de segments de rollback
SEQUENCES	
CREATE SEQUENCE	Création de séquences dans le schéma
CREATE ANY SEQUENCE	Création de séquences dans n'importe quel schéma
ALTER ANY SEQUENCE	Modification de n'importe quelle séquence
DROP ANY SEQUENCE	Suppression de séquences dans n'importe quel schéma
SELECT ANY SEQUENCE	Interrogation de séquences dans n'importe quel schéma

SESSIONS	
CREATE SESSION	Connexion à la base
ALTER RESOURCE COST	Application de coûts pour les ressources de la session
ALTER SESSION	Permet l'instruction ALTER SESSION
RESTRICTED SESSION	Connexion restreinte à l'instance
SYNONYMS	
CREATE SYNONYM	Création de synonymes dans le schéma
CREATE ANY SYNONYM	Création de synonymes dans n'importe quel schéma
CREATE PUBLIC SYNONYM	Création de synonymes publics
DROP ANY SYNONYM	Suppression de synonymes dans le schéma
DROP PUBLIC SYNONYM	Suppression de synonymes publics
TABLES	
CREATE TABLE	Création de tables dans le schéma
CREATE ANY TABLE	Création de tables dans n'importe quel schéma
ALTER ANY TABLE	Modification de tables ou vues dans n'importe quel schéma
BACKUP ANY TABLE	Autorise l'utilisation de l'outil Export
DELETE ANY TABLE	Suppression de lignes des tables de n'importe quel schéma
DROP ANY TABLE	Suppression ou troncature de tables dans n'importe quel schéma
INSERT ANY TABLE	Insertion de lignes dans les tables de n'importe quel schéma
LOCK ANY TABLE	Vérouillage des tables ou vues de n'importe quel schéma
SELECT ANY TABLE	Interrogation des tables, vues ou vues matérialisées de n'importe quel schéma
FLASHBACK ANY TABLE	Autorise les requêtes FlashBack sur n'importe quelle table de n'importe quel schéma
UPDATE ANY TABLE	Mise à jour de lignes dans les tables de n'importe quel schéma
TABLESPACES	

CREATE TABLESPACE	Création de tablespace
ALTER TABLESPACE	Modification de tablespace
DROP TABLESPACE	Suppression de tablespace
MANAGE TABLESPACE	Autorise la mise en ligne/hors ligne des tablespace
UNLIMITED TABLESPACE	Quota illimité sur le tablespace
TRIGGERS	
CREATE TRIGGER	Création de déclencheurs dans le schéma
CREATE ANY TRIGGER	Création de déclencheurs dans n'importe quel schéma
ALTER ANY TRIGGER	Activation, désactivation, compilation de déclencheurs dans n'importe quel schéma
DROP ANY TRIGGER	Suppression de déclencheurs dans n'importe quel schéma
ADMINISTER DATABASE TRIGGER	Création de déclencheurs de base de données
TYPES	
CREATE TYPE	Création d'objets et de corps d'objets dans le schéma
CREATE ANY TYPE	Création d'objets et de corps d'objets dans n'importe quel schéma
ALTER ANY TYPE	Modification d'objets et de corps d'objets dans n'importe quel schéma
DROP ANY TYPE	Suppression d'objets et de corps d'objets dans n'importe quel schéma
EXECUTE ANY TYPE	Utilisation d'objets et de corps d'objets dans n'importe quel schéma
UNDER ANY TYPE	Création de sous-types sous des types non finaux
VIEWS	
CREATE VIEW	Création de vues dans le schéma
CREATE ANY VIEW	Création de vues dans n'importe quel schéma
DROP ANY VIEW	Suppression de vues dans n'importe quel schéma
UNDER ANY VIEW	Création de sous-vues sous n'importe quelle vue objet
Autres	
ANALYZE ANY	Analyse de tables, clusters et index dans n'importe quel schéma

AUDIT ANY	Audit de n'importe quel objet de n'importe quel schéma
COMMENT ANY TABLE	Pose de commentaires sur les tables et vues de n'importe quel schéma
FORCE ANY TRANSACTION	Force le commit ou rollback de n'importe quelle transaction distribuée douteuse
FORCE TRANSACTION	Force le commit ou rollback de n'importe quelle transaction distribuée douteuse
GRANT ANY OBJECT PRIVILEGE	Assigne n'importe quel privilège objet
GRANT ANY PRIVILEGE	Assigne n'importe quel privilège système
RESUMABLE	Active l'allocation d'espace pour les instructions résumables
SELECT ANY DICTIONARY	Interrogation de n'importe quel objet dans le dictionnaire du schéma SYS
SYSDBA	Autorise les opérations STARTUP et SHUTDOWN ALTER DATABASE: open, mount, back up, ou changement du jeu de caractères CREATE DATABASE ARCHIVELOG et RECOVERY CREATE SPFILE Inclue le privilège RESTRICTED SESSION
SYSOPER	Autorise les opérations STARTUP et SHUTDOWN ALTER DATABASE OPEN MOUNT BACKUP ARCHIVELOG et RECOVERY CREATE SPFILE Inclue le privilège RESTRICTED SESSION

Liste des privilèges objet

Privilège objet	Description
TABLES	
ALTER	Modifier la définition d'une table
DELETE	Supprimer des lignes de la table

DEBUG	Accès via un debugger
INDEX	Création d'un index sur la table
INSERT	Insertion de lignes dans la table
REFERENCES	Création d'une contrainte d'intégrité
SELECT	Interrogation d'une table
UPDATE	Mise à jour de la table
VIEWS	
DEBUG	Accès via un debugger
DELETE	Suppression de lignes
INSERT	Insertion de lignes
REFERENCES	Définition de contrainte d'intégrité
SELECT	Interrogation de la vue
UNDER	Création d'une sous-vue
UPDATE	Mise à jour de la vue
SEQUENCES	
ALTER	Modification de la définition d'une séquence
SELECT	Interrogation d'une séquence
PROCEDURES	
DEBUG	Accès via un debugger
EXECUTE	Compilation, exécution d'une procédure, d'une fonction ou d'un package
MATERIALIZED VIEWS	
ON COMMIT REFRESH	Création d'une vue matérialisée de type refresh-on-commit sur la table
QUERY REWRITE	Vue matérialisée en query rewrite
SELECT	Interrogation d'une vue matérialisée
DIRECTORIES	
READ	Droits de lecture sur le répertoire

WRITE	Droits d'écriture sur le répertoire
LIBRARIES	
EXECUTE	Utilisation d'un objet et de ses méthodes
OBJECT TYPES	
DEBUG	Accès via un debugger
EXECUTE	Utilisation d'un objet et de ses méthodes
UNDER	Création d'un sous-type
INDEXTYPES	
EXECUTE	Utilisation d'un indextype
OPERATORS	
EXECUTE	Utilisation d'un opérateur

Sauvegarde et restauration de base de données

LES UTILITAIRES EXPORT ET IMPORT

Remarque préliminaire

Il est important de préciser que nous allons étudier une méthode logique de sauvegarde et de restauration d'une base de données. Cette dernière permet de réaliser une sauvegarde du contenu logique d'une base dans un fichier dump qui est un fichier de transfert Oracle au format binaire. Ce fichier pourra donc être utilisé dans le but de créer à nouveau les objets qu'il contient.

Les différents modes d'export et d'import

Niveau base de données complète

Ce mode est complet de par le fait que ce type d'exportation prend en compte tous les objets de la base de données. En effet, toutes les informations inhérentes aux structures de la base comme par exemple les définitions de tablespace sont comprises. A l'occasion de l'insertion des données, tous les objets sont importés et créés dans la nouvelle base de données.

Le paramètre FULL correspond à ce mode d'exportation et d'importation.

Niveau utilisateur

Ce mode permet d'exporter l'ensemble des objets appartenant à un utilisateur spécifique tels que les tables, fonctions ou encore déclencheurs.

Le paramètre OWNER permet de spécifier les utilisateurs que l'on souhaite exporter. Le paramètre FROMUSER spécifie l'utilisateur que l'on désire importer à partir du fichier dump. Le paramètre TOUSER indique le schéma destinataire.

Niveau table

Ce mode permet d'exporter uniquement des tables en particulier ainsi que leurs objets associés (index, privilèges ou encore contraintes). Il est important de préciser que les tables se doivent d'être nommées grâce au paramètre TABLES.

Niveau tablespace

Ce mode permet d'exporter et d'importer les métadonnées relatives à des tablespaces spécifiques ainsi que les objets qu'ils contiennent.

Privilèges pré-requis

Actions	Privilège ou rôle nécessaire
Exporter son propre schéma	CREATE SESSION
Exporter d'autres schémas	SYSDBA, EXP_FULL_DATABASE et DBA
Exporter la base entière ou tablespaces	EXP_FULL_DATABASE
Importer un objet du fichier DUMP	IMP_FULL_DATABASE

Les principaux paramètres de contrôle

Paramètres d'export

Paramètre	Description	Valeur par défaut
Userid	chaîne de connexion à la base de données	
Buffer	Taille du buffer de transfert	4096
Compress	Compression des extents en un seul	Y
Constraints	Export des contraintes	Y
File	Nom du fichier DUMP	expdat.dmp
Log	Nom du fichier de sortie du compte-rendu, pour voir les erreurs en particulier	
Full	Export de toute la base	N
Grants	Export des privilèges	Y
Help	Affiche la liste des paramètres supportés, aucun DUMP généré.	N
Indexes	Export des index	Y
Owner	Utilisateur(s) à exporter	userid
Parfile	Fichier contenant les paramètres d'export	
Recordlength	Taille des enregistrements (migration SE)	
Record	Indique que l'export doit stocker les informations sur les exports et les objets exportés	Y
Inctype	Export incrémental (COMPLETE, CUMULATIVE, INCREMENTAL)	

Rows	Export des lignes	
Query	Définit une condition de filtre pour exporter un sous-ensemble	
Tables	Table(s) à exporter	
Consistent	Positionne sa session en " READ ONLY " le temps de l'export. Cela permet donc de préserver la cohérence des données exportées.	N
Direct	Chargement direct par tableau	N
Statistics	Analyse des objets exportés	ESTIMATE
Feedback	Affiche la progression de l'export tous les n enregistrements	N
Point_in_time_Recover	Indique si on autorise l'export des tablespaces	N
Recover_Tablespaces	Liste des tablespaces à sauvegarder	
Volsize	Nombre d'octets à écrire sur chaque volume bande	

Paramètres d'import

Paramètre	Description	Valeur par défaut
Userid	chaîne de connexion à la base de données	
Buffer	Taille du buffer de transfert	10240
Commit	Ecriture régulière des blocs de données	N
File	Nom du fichier DUMP	expdat.dmp
Log	Nom du fichier de sortie du compte-rendu, pour voir les erreurs en particulier	
Fromuser	Utilisateur à exporter vers TOUSER	

Full	Import de tout le contenu du DUMP	N
Grants	Export des privilèges	Y
Help	Affiche la liste des paramètres supportés, aucun DUMP généré.	N
Ignore	Ignore les erreurs et continue l'import	N
Indexes	Import des index	Y
Parfile	Fichier contenant les paramètres d'import	
Recordlength	Taille des enregistrements (migration SE)	
Rows	Import des données	Y
Destroy	Détruit les objets s'ils existent avant de les importer	N
Show	Liste le contenu du fichier d'export, aucune opération n'est effectuée dans la base.	N
Tables	Table(s) à importer	
Touser	Utilisateur destinataire	
Charset	Code alphabet du pays de référence s'il est différent de celui de la création de base	NLS_LANG
Point_in_time_recover	Indique si on autorise l'import des tablespaces	N
Skip_unusable_indexes	Permet de repousser la reconstruction de l'index après l'insertion des données dont ils dépendent	N
Analyze	Exécute la commande ANALYZE dans le fichier dump	Y
Feedback	Affiche la progression de l'import tous les n enregistrements	0
Volsize	Nombre d'octets dans le fichier pour	

chaque volume bande

Méthode pour exporter et importer

Export FULL des objets

```
C:\> exp userid=system/manager file=c:\backup\export_full.dump  
log=c:\control\export_full.log full=y rows=n
```

Ici, on se connecte à la base en tant que SYSTEM (**userid=system/manager**) et on exporte toute la base (**full=y**) sans les données (**rows=n**). On sauvegarde la sortie dans le fichier de log (**log=c:\control\export_full.log**).

Voici quelques exemples supplémentaires :

Export du schéma Mika

```
C:\> exp userid=system/manager file=c:\backup\export_full.dump  
log=c:\control\export_full.log owner=Mika
```

Export de la table TEST de l'utilisateur Mika

```
C:\> exp userid=system/manager file=c:\backup\export_full.dump  
log=c:\control\export_full.log tables=Mika.Test
```

Export du tablespace USER

```
C:\> exp userid=system/manager file=c:\backup\export_full.dump  
log=c:\control\export_full.log tablespaces=user
```

Les exemples suivants permettent d'importer les fichiers générés dans les exemples précédents.

Les exemples suivants vont vous permettre d'importer les éléments sauvegardés dans les précédents exemples.

Import du schéma Mika

```
C:\> imp userid=scott/tiger file=c:\backup\export_full.dump  
log=c:\control\export_full.log owner=Mika
```

Import du schéma Mika dans le schéma TESTSCHEMA

```
C:\> imp userid=system/manager file=c:\backup\export_full.dump  
log=c:\control\export_full.log fromuser=Mika touser=testschema
```

Import de la table TEST de l'utilisateur Mika dans NOUVEAUSCHEMA

```
C:\> imp userid=system/manager file=c:\backup\export_full.dump  
log=c:\control\export_full.log fromuser=Mika touser=NOUVEAUSCHEMA  
tables=mika.test
```

Import de tous les schémas inclus dans le DUMP

```
C:\> imp userid=system/manager file=c:\backup\export_full.dump  
log=c:\control\export_full.log
```

SAUVEGARDE A FROID

Introduction

L'expression « sauvegarde à froid » signifie que la base est arrêtée afin de procéder à une sauvegarde. Dans ce cas, l'activité de la base de données est interrompue et par conséquent les fichiers peuvent être sauvegardés sans aucune corruption de données.

Sauvegarde

Le script suivant permet de sauver les fichiers de la base de données sous Windows. Il génère le script de sauvegarde des fichiers (datafiles, logfiles, controlfile et tempfile) qui copie ces fichiers dans le répertoire défini par la variable **repertoire**.

Il devra être adapté selon vos chemins personnels et surtout l'OS (copy sera remplacé par cp pour Unix par exemple).

```
-- Variables d'environnement de SQL*Plus de formatage de l'affichage  
Set feedback off  
Set Linesize 200  
Set Heading off  
Set Pagesize 0  
Set Trimspool off  
Set Verify off  
define repertoire ='c:\archive' -- répertoire de destination des
```

```

fichiers sauvegardés
define fichier_control=c:\control_backup.sql -- définition du
fichier de sortie
spool &fichier_control
select 'host copy ' || name || ' &repertoire ' from v$datafile
order by 1 ;
select 'host copy ' || member || ' &repertoire ' from v$logfile
order by 1 ;
select 'host copy ' || name || ' &repertoire ' from v$controlfile
order by 1 ;
select 'host copy ' || name || ' &repertoire ' from v$tempfile
order by 1 ;
spool off
-- Fermeture de la base de données pour avoir des fichiers
synchronisés
shutdown immediate
@@fichier_control
startup

```

SAUVEGARDE A CHAUD

Introduction

Comme nous venons de le voir, effectuer une restauration ou récupération d'une base de données implique que la base soit arrêtée, cependant dans des environnements à haute disponibilité cela est parfois impossible. Une solution dans ce cas existe: "Hot backup" ou sauvegarde à chaud. Le problème est le suivant, dans le cas d'une haute disponibilité, forcément l'état des fichiers change constamment : des modifications sont apportés dans les fichiers de données, des informations de contrôle sont écrites dans les fichiers de contrôle, et des informations de reprise sont consignées dans les fichiers REDO, lesquels peuvent également être archivés. Pour effectuer une sauvegarde dans ce cas, la solution est de placer chaque tablespace dans le mode de sauvegarde et de sauvegarder les fichiers de données, puis de rétablir le tablespace dans le mode normal. On sauvegarde donc des fichiers incohérents puisque les SCN, contenus dans les entêtes de fichiers de données sont différents. Il faudra effectuer une récupération de support pour rendre ces différents SCN cohérents et pouvoir ouvrir la base.

Etant donné que les changements qui ont lieu durant la phase de sauvegarde d'un tablespace donnent lieu à une consignment dans le flux de reprise, la base doit être en mode ARCHIVELOG. Les tablespaces en mode lecture seule (" alter tablespace tools read only ") ne peuvent pas être sauvegardé car la base ne peut pas les modifier, pas plus que ceux gérés localement. Dans ce cas là, la solution consiste à les recréer.

Sauvegarde

Tout d'abord nous allons commencer par créer un script de sauvegarde à chaud de la base de données. Ce script que nous appellerons sauvegarde_chaud.sql nous aidera dans le traitement des tablespaces et des fichiers de données.

```
SET feedback off pagesize 0 heading off verify off linesize 100
trimspool on
PROMPT veuillez entrer le chemin du répertoire destinataire des
sauvegardes
ACCEPT repertoire
PROMPT veuillez entrer le chemin du premier fichier
ACCEPT fichier
PROMPT veuillez entrer le chemin du second fichier
ACCEPT spool
SPOOL &fichier
PROMPT spool &spool ;;
PROMPT archive log list ;;
PROMPT alter system switch logfile ;;
SELECT ' alter tablespace ' || tablespace_name || ' begin backup  ;
'
    FROM dba_tablespaces
    WHERE status NOT IN ('READ ONLY', 'INVALID', 'OFFLINE');
SELECT ' host copy ' || file_name || ' &repertoire '
    FROM dba_data_files
    WHERE tablespace_name NOT IN (
        SELECT tablespace_name
        FROM dba_tablespaces
        WHERE status IN
            ('READ ONLY', 'INVALID',
'OFFLINE'));
SELECT ' alter tablespace ' || tablespace_name || ' end backup  ; '
    FROM dba_tablespaces
    WHERE status NOT IN ('READ ONLY', 'INVALID', 'OFFLINE');
PROMPT alter database backup controlfile to
'&repertoire\control.ctl' REUSE ;;
PROMPT alter system switch logfile ;;
PROMPT archive log list ;;
PROMPT spool off ;;
SPOOL off;
@&fichier
```

Le script précédant doit générer et exécuter un autre script de la forme :

```
spool c:\oracle\sauvegarde\hot_backup.sql ;
archive log list ;
alter system switch logfile ;
    alter tablespace SYSTEM begin backup  ;
    alter tablespace RBS begin backup  ;
```

```

alter tablespace USERS begin backup ;
alter tablespace TEMP begin backup ;
alter tablespace TOOLS begin backup ;
alter tablespace INDX begin backup ;
alter tablespace DRSYS begin backup ;
host copy C:\ORACLE\ORADATA\BD0\USERS01.DBF c:\oracle\sauvegarde
host copy C:\ORACLE\ORADATA\BD0\DR01.DBF c:\oracle\sauvegarde
host copy C:\ORACLE\ORADATA\BD0\TOOLS01.DBF c:\oracle\sauvegarde
host copy C:\ORACLE\ORADATA\BD0\INDX01.DBF c:\oracle\sauvegarde
host copy C:\ORACLE\ORADATA\BD0\RBS01.DBF c:\oracle\sauvegarde
host copy C:\ORACLE\ORADATA\BD0\TEMP01.DBF c:\oracle\sauvegarde
host copy C:\ORACLE\ORADATA\BD0\SYSTEM01.DBF c:\oracle\sauvegarde
alter tablespace SYSTEM end backup ;
alter tablespace RBS end backup ;
alter tablespace USERS end backup ;
alter tablespace TEMP end backup ;
alter tablespace TOOLS end backup ;
alter tablespace INDX end backup ;
alter tablespace DRSYS end backup ;
alter database backup controlfile to
'c:\oracle\sauvegarde\control.ctl' REUSE ;
alter system switch logfile ;
archive log list ;
spool off ;

```

Lorsque le tablespace est en mode sauvegarde, si une modification intervient sur un objet de celui-ci, des reprises seront créées, ce qui surchargera la phase de récupération lorsque le tablespace sera à nouveau placé dans le mode normal : donc veillez à effectuer cette opération lorsque le serveur est le moins sollicité.

Lorsqu'on positionne le mode sauvegarde, on peut alors copier le fichier de données, on peut effectuer cette copie sans au préalable avoir placé le tablespace dans ce mode, cependant la sauvegarde qu'on obtiendra sera inutilisable.

Pour vérifier que le mode sauvegarde est bien activé, vérifiez le fichier des alertes (l'altération d'un tablespace y est consignée) ou consultez la vue **v\$backup** :

```

SELECT b.tablespace_name ,
       b.file#,
       b.status,
       b.change#   SCN,
       b.time horodate
FROM v$backup b
ORDER BY tablespace_name;

```

TABSPACE_NAME	FILE#	STATUS	CHANGE#	TIME
DRSYS	7	NOT ACTIVE	524024	05/11/03
INDX	6	NOT ACTIVE	524023	05/11/03
RBS	2	NOT ACTIVE	524019	05/11/03
SYSTEM	1	NOT ACTIVE	524018	05/11/03
TEMP	4	NOT ACTIVE	524021	05/11/03
TOOLS	5	ACTIVE	1085795	20/11/03
USERS	3	NOT ACTIVE	524020	05/11/03

Cette vue nous apprend, que le fichier 5 du tablespace TOOLS est en phase de récupération, et qu'il a été mis dans cet état le 20/11/03.

RESTAURATION DES DONNEES

Restauration des fichiers de données

La restauration de la base de données est particulièrement simple, elle est réalisée par les étapes suivantes :

- Arrêt de la base de données
- Suppression des datafiles, logfiles, controlfiles et tempfiles
- Restauration des fichiers
- Démarrage de la base de données

Si la sauvegarde à froid s'est bien déroulée alors le démarrage suite à la restauration des fichiers ne doit poser aucun problème.

Restauration totale de la base de données

Restaurer la base de données à l'instant de la sauvegarde peut être salubre mais c'est souvent insuffisant, la perte des données devant être la plus petite possible.

Afin de récupérer le maximum de données il convient donc de passer la base de données en mode **ARCHIVELOG**.

Restauration complète à partir de la sauvegarde à froid

Je vous propose un cas pratique pour illustrer le modus operandi de cette restauration. Imaginons la perte du datafile `c:\oracle\oradata\BD0\users01.dbf`. La perte d'un datafile provoque immédiatement l'erreur suivante et arrête la base de données :

```
ORA-01157: impossible d'identifier ou de verrouiller le fichier de
données 3 -
voir le fichier de trace DBWR
ORA-01110: fichier de données 3 : 'C:\ORACLE\ORADATA\BD0\USERS01.DBF'
```

Windows a l'avantage de bloquer la suppression d'un fichier accédé par un processus, limitant ainsi les risques de suppression accidentelle des datafiles. Néanmoins, cette fonctionnalité ne nous met pas à l'abri d'une panne matérielle.

Pour le test, vous devrez donc arrêter la base avant de supprimer le datafile.

La première étape consiste alors à restaurer le fichier de notre dernière sauvegarde complète (la vue **v\$recover_files** permet de retrouver le(s) fichier(s) à restaurer). Une fois que le fichier est à nouveau disponible il faut le resynchroniser. C'est à dire qu'il faut rejouer les redos jusqu'à ce que les données soient complètement restaurées et que le numéro SCN du tablespace corresponde au numéro SCN des autres tablespaces de la base de données.

Ensuite on démarre la base en prenant soin de lancer un RECOVER :

```
SQL> STARTUP MOUNT;  
SQL> RECOVER DATABASE  
media recovery completed  
SQL> ALTER DATABASE OPEN;
```