Architecture du Système Oracle 10g

Faculté Polydisciplinaire de Ouarzazate (IGE 2012/2013)

Mohamed NEMICHE

Table des matières

Introduction à l'Architecture et Installation Oracle

Objectifs de chapitre :

- Installation d'Oracle
- Comprendre l'architecture d'un serveur de BD Oracle
- Démarrage et arrêt d'une instance et d'une base de données Oracle
- Création d'une base de données opérationnelle
- Gestion des fichiers d'une base de données Oracle
- Gestion de la structure logique (tablespaces, segments, extents et blocs)

Bases de données relationnelles

- Collection de données opérationnelles enregistrées sur un support adressable et utilisées par les systèmes et les applications
- Les données doivent être structurées indépendamment d'une application particulière
- Elles doivent être cohérentes (contraintes), non redondantes (formes normales) et accessibles simultanément par plusieurs utilisateurs

SGBD : Système de Gestion de bases de données

Un système de gestion de base de données (SGBD) est un ensemble de programmes qui permettent la gestion et l'accès à une base de données.

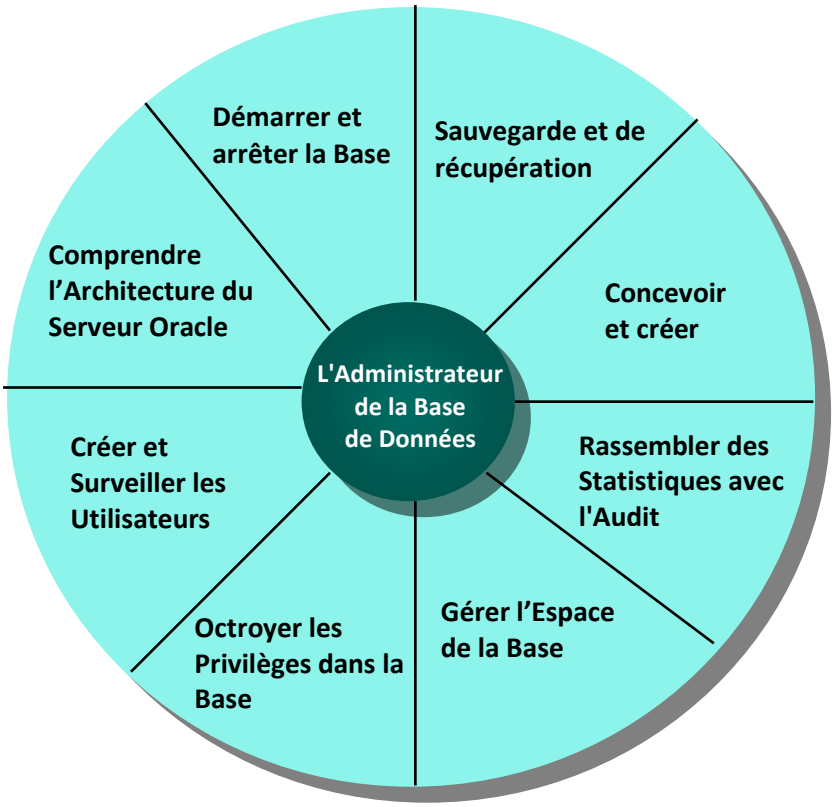
- Un SGBD possède son propre système de fichier.
- Un SGBD assure la reprise en cas de panne.
- Un SGBD doit permettre la sauvegarde et la restauration d'une BD.
- Un SGBD doit permettre une gestion des rôles et droits.
- Une des fonctions importante des SGBD modernes est d'autoriser les utilisateurs d'effectuer des opérations simultanées (concurrentes) sur des données partagées de la BD. Si ces op ne sont pas sous contrôle, les accès interfèrent tôt ou tard les uns avec les autres et la BD devient incohérente. Pour éviter cela, le SGBD met en place un protocole de contrôle de simultanéité (ou de concurrence) qui empêche les accès à la BD d'interférer.

Transaction

Une transaction est un ensemble de modifications de la base qui forme un tout indivisible. Il faut effectuer ces modifications entièrement ou pas du tout, sous peine de laisser la base dans un état incohérent.

- Action **atomique** : entièrement ou pas du tout
- Préservant la **consistance** de la BD
- Comme si l'usager était **isolé** sur la BD : ses résultats intermédiaires (état temporairement incohérent) sont masqués aux autres transactions.
- A effet **durable** sur la BD, une fois terminées comme prévu
    - les effets d'une transaction globalement terminée ne peuvent pas être détruits ultérieurement par une quelconque défaillance.
        - **ModèleACIDde transactions**

Les tâches de l'administrateur de la Base de données



- **Dans la phase de « conception »**
    - définition du schéma conceptuel de la base
    - règles de gestion, cohérence des informations
    - volumétrie

- **Dans la phase de maintenance**
    - Planification et création des BD
        - ❖ Gestion des structures physiques
        - ❖ Gestion des structures logiques
    - Gestion de la sécurité, des utilisateurs
    - Sauvegarde et restauration
    - Optimisation de la base de données
        - ❖ optimisation de requêtes
    - Administration du réseau
- Grandes fonctions de DBA :
    - installer le SGBD et les applications clientes
    - Créer la base de données en faisant des choix au niveau physique
    - Gérer les utilisateurs
    - Assurer la cohérence et la sécurité des données
    - Echanger des données avec l'extérieur
    - Améliorer les performances
        - ❖ gestion des ressources mémoires
        - ❖ gestion des temps de réponses

Généralités

- Tendances actuelles
    - progiciels intégrés
        - ❖ minimise les besoins en administration

          … sans pour autant les supprimer
    - amélioration des outils d'administration par les fournisseurs de SGBD
        - ❖ Notion d'Assistant
            - o pour la création des bases, la sauvegarde/restauration, ...
    - A terme, vers des BD qui s'autoadministrent

Architecture Client Serveur

- L'architecture client/serveur désigne un mode de communication entre plusieurs ordinateurs à doubles niveaux d'hiérarchie.
- Le logiciel client peut envoyer des requêtes à un serveur via un protocole de communication à travers un support (réseau).
- Le serveur est initialement passif à l'écoute des requêtes clients sur un port déterminé. dès qu'une requête lui parvient, il décide de la traiter ou de la mettre en attente et envoie une réponse.
- Oracle est un SGBD doté d'une architecture Client/Serveur.

Histoire d'Oracle

- Software Development Laboratories (SDL) a été créé en 1977.
- En 1979, SDL change de nom en devenant Relational Software, Inc. (RSI) et introduit son produit Oracle V2 comme base de données relationnelle.
    - ❖ La version 2 ne supportait pas les transactions mais implémentait les fonctionnalités SQL basiques de requête et jointure. Il n'y a jamais eu de version 1, pour des raisons de marketing, la première version a été la version 2. Celle-ci fonctionnait uniquement sur les systèmes Digital VAX/VMS.
- En 1983, RSI devient Oracle Corporation pour être plus représentative de son produit phare.
    - ❖ La version 3 d'Oracle, entièrement ré-écrite en langage de programmation C, est publiée.
    - ❖ Supporte les transactions grâce aux fonctionnalités de *commit* et *rollback*.
    - ❖ Unix est supportée dans cette version
- En 1984, la version 4 d'Oracle apparaît, supportant la cohérence en lecture (*read consistency*).
- Début 1985, Oracle commence à intégrer le modèle client-serveur, avec l'arrivée des réseaux au milieu des années 1980.
- En 1988, Oracle met sur le marché son ERP - Oracle Financials basé sur la base de données relationnelle Oracle.
    - ❖ Oracle version 6 supporte le PL/SQL
    - ❖ le verrouillage de lignes (*row-level locking*)
    - ❖ les sauvegardes à chaud (*hot backups*, lorsque la base de données est ouverte).

- En 1992, la version 7 d'Oracle supporte les contraintes d'intégrité, les procédures stockées et les déclencheurs (*triggers*).
- En 1995, acquisition d'un puissant moteur multidimensionnel, commercialisé sous le nom d'Oracle Express.
- En 1997, la version 8 introduit le développement orienté objet et les applications multimédia.
- En 1999, la version 8i est publiée dans le but d'affiner ses applications avec Internet. La base de données comporte nativement une machine virtuelle Java.
- En 2001, Oracle 9i
- En 2004, la version 10g est publiée.
- En 2005, vers la fin novembre, une version complètement gratuite est publiée, la « Oracle Database 10g Express Edition ».
- Septembre 2009, sortie de Oracle 11g Release 2

Généralités

- Oracle 10g est commercialisé selon trois gammes (Edition) :
    - ❖ Edition Standard (Standard Edition)
    - ❖ Edition Entreprise (Entreprise Edition)
    - ❖ Edition Personnelle (Personal Edition)
- Oracle Express Edition
- Oracle 10g Database est un SGBD qui fonctionne sur de nombreuse plates-formes Unix(dont Linux) et également dans l'environnement Windows
    - ❖ Oracle propose donc une organisation de la mémoire et des ressources disque la plus indépendante possible de l'environnement système. C'est un SGBD qui offre une architecture très ouverte.

L'architecture Oracle comporte plusieurs composants principaux :

- Serveur Oracle : comporte plusieurs fichiers, processus et structures mémoire. Le serveur Oracle est constitué d'une instance oracle et d'une base oracle
    - ❖ Instance Oracle : L'instance Oracle comprend une région de la mémoire appelée La SGA (System Global Area), ainsi que les processus d'arrière-plan utilisé pour gérer la base de données
- La zone mémoire du programme (**PGA**)

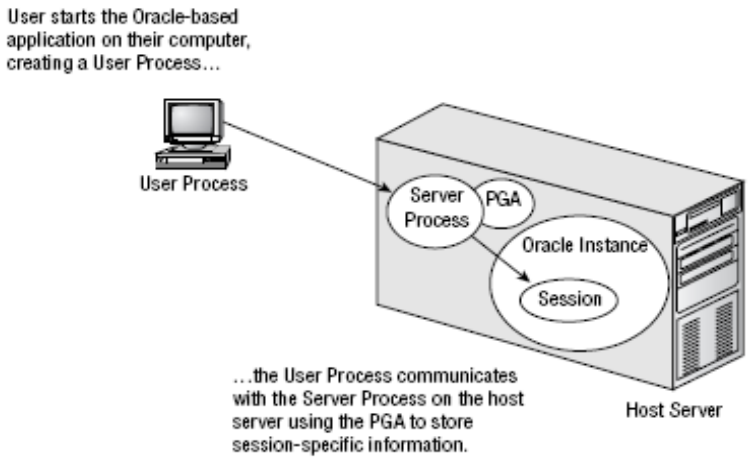
- ❖ Zone mémoire utilisée par un seul processus serveur à la différence de la SGA qui est partagée par tous les processus serveurs
- ❖ PGA contient :
  - o une zone de tri
  - o des informations sur la session
  - o l'état du curseur
  - o …

- Base de données Oracle :
  - ❖ Structure physique :
    - o Fichiers de données, Fichiers redo log, Fichiers de contrôle.
    - o Autres fichiers importants : (fichier de paramètres, fichier de mots de passe)
  - ❖ Structure logique
    - o Tablespace, segment, extent, bloc
- Les Processus serveurs : gèrent les requêtes des utilisateurs provenant des connexions à la base de données ; ils sont chargés de :
  - ❖ la communication entre la SGA et le processus utilisateur.
  - ❖ analyser, d'exécuter les requêtes SQL des utilisateurs, de lire les fichiers de données, de placer les blocs de données correspondants dans la SGA et de Renvoyer les résultats des commandes SQL au processus utilisateur.
- Le serveur oracle supporte
  - ❖ SQL (LDD , LMD, LCD)
  - ❖ PL/SQL
  - ❖ Autres langages de programmation (Pro*C …)
- **Connexion à un serveur Oracle**
  - ❖ Une connexion est un chemin de communication entre un processus utilisateur et un processus serveur. Il existe trois types de connexions grâce auxquelles un utilisateur peut accéder à un Serveur Oracle :
    - o Connexion locale : Selon cette méthode, un utilisateur est directement connecté sur la machine faisant office de Serveur Oracle.
    - o Connexion Deux Tiers : Ce type de connexion est couramment nommé "Connexion Client Serveur", un utilisateur se connecte à partir d'une machine directement connectée à un Serveur Oracle.
    - o Connexion Multi Tiers : Dans une architecture multi tiers, la machine de l'utilisateur se connecte à un Serveur applicatif

(Par exemple un Serveur Web) qui lui-même va se connecter au serveur Oracle pour récupérer les données issues de la base de données.

- Session : une session est une connexion spécifique d'un utilisateur à un serveur oracle.
- La session démarre lorsque l'utilisateur est authentifié par le serveur oracle et se termine lorsque l'utilisateur se déconnecte ou en cas de déconnexion anormale.

FIGURE 1.12 The relationship between User and Server processes

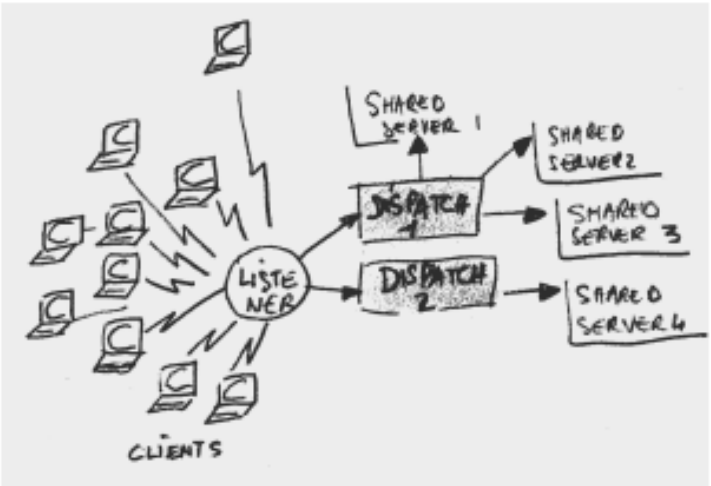


The Server Process communicates with the Oracle instance on behalf of the user. The Oracle instance is examined in the next section.

- Oracle supporte deux modes de fonctionnement :
    1. **Serveur dédié** : chaque fois qu'un utilisateur se connecte, il est pris en charge par un processus serveur.
        - ❖ Si 100 utilisateurs se connectent, 100 processus serveurs sont créés de même
        - ❖ Avantage :
            - Une commande SQL est tout de suite et directement prise en compte par un processus serveur
        - ❖ Inconvénient :
            - Chaque processus serveur occupe une zone mémoire et utilise la CPU
        - ❖ Meilleure configuration (recommandée et utilisée par la bcp de DBA), si les ressources matérielles le permettent.

2. **Serveur Partagé** : c'est un groupe de processus serveurs qui s'occupent d'un grand nombre de processus utilisateurs.

   - ❖ Les processus utilisateurs sont alloués à un processus DISPACHER, celui-ci met les requêtes utilisateurs dans une fille d'attente, et le processus serveur exécute toutes les requêtes, une par une
   - ❖ Avantage :
     - Réduire la charge de la CPU et utilise moins de mémoire
   - ❖ Inconvénient :+
     - Lors de forte utilisations de la BDD, il risque d'y avoir des temps d'attente (performance)



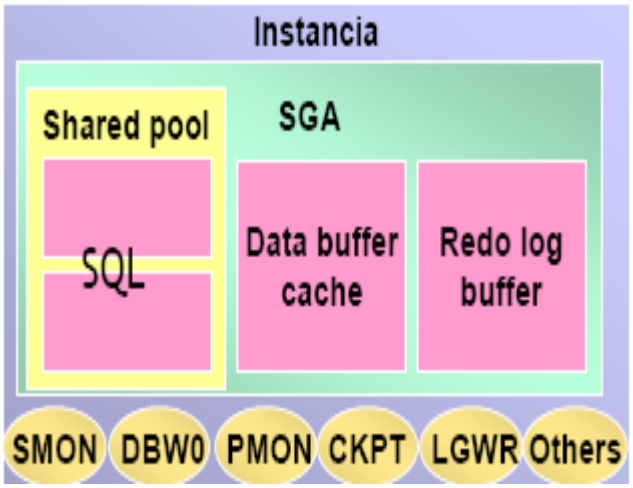
Chapitre 1

Architecture – Instance

I. Architecture – Instance

- Serveur Oracle = instance Oracle + base de données Oracle
  - ❖ Instance Oracle :
    - c'est un moyen pour accéder à une base de données Oracle (ouvre une unique base de données)

1. Structure Mémoire (SGA)



2. Processus en arrière-plan

L'utilisation de la mémoire par Oracle 10g

- Dans tout système informatique, l'utilisation de mémoire est synonyme de performance.
- la mémoire partagée SGA (*System Global Area):*
- la mémoire allouée pour chaque programme PGA (*Program Global Area) ;*
- Les données auxquelles on accède et qui sont manipulées en mémoire le sont beaucoup plus rapidement que sur disque.
- Il est important de bien comprendre ces éléments, car ils interviennent dans les opérations d'amélioration des performances.

I. 1 Architecture - Instance : SGA

- La SGA (*System Global Area) représente la zone mémoire déterminante d'une instance, tant* par sa taille que par son rôle.
  - ❖ C'est elle qui assure le partage des données entre les utilisateurs.

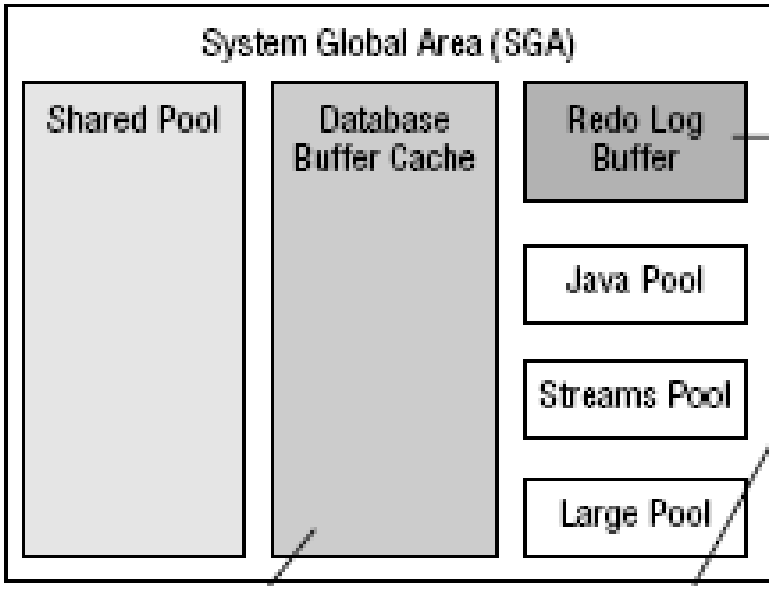
- Oracle utilise la mémoire SGA comme buffer intermédiaire (plus rapide que le disque) pour l'échange de données entres processus

❖ Elle est divisée trois composants obligatoires :

- shared pool
- Database buffer cache (Le cache de données)
- redo log buffer (Le cache de reprise)

❖ Et de trois composants optionnels

- Java pool
- Large pool
- Streams pool



Database Buffer cache

- Est utilisé pour stocker des blocs de données en mémoire afin d'accélérer l'interrogation et/ou la modification
- Aucune modification est faite directement sur les données du disque
  - ❖ Oracle lit les données suite à la demande d'un *processus utilisateur et ensuite valide les* modifications sur le disque
- Il utilise un algorithme nommé LRU mois récemment utilisés (Least-Recently Used) pour déterminer les données à libérer du cache

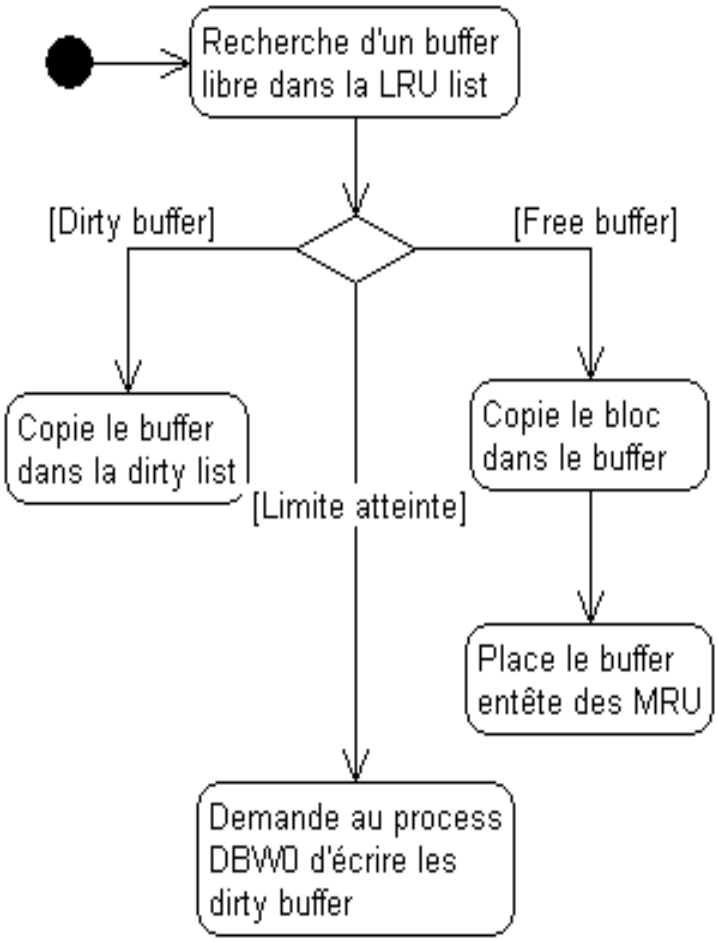
Organisation du buffer cache

- Le buffer cache est organisé en 2 listes :

- La dirty list
- La liste LRU (Least Recently Used)

- La dirty list contient les buffers qui ont été modifiés mais ne sont pas encore écrits sur le disque
- La liste LRU contient les buffers libres (qui n'ont pas été modifiés), les « pinned » buffers (qui sont en cours de modification, les dirty buffers qui n'ont pas encore été déplacés dans la dirty list

Database Buffer cache (recherche d'un buffer libre)



Database Buffer cache

- Vues système utilisées
  - V$SGA ;
  - V$PARAMETER ;

Buffer Redo Log

- C’est un buffer circulaire qui stocke les modifications réalisées sur la base de données avec les opérations: insert, delete, update, create, alter y drop.
- Permet à oracle de reconstruire les modifications des données en cas de panne
- L'information Redo reste dans le buffer Redo log jusqu'à ce qu‘oracle la stocke sur le disque
- Sa taille est définie par LOG_BUFFER

Shared Pool

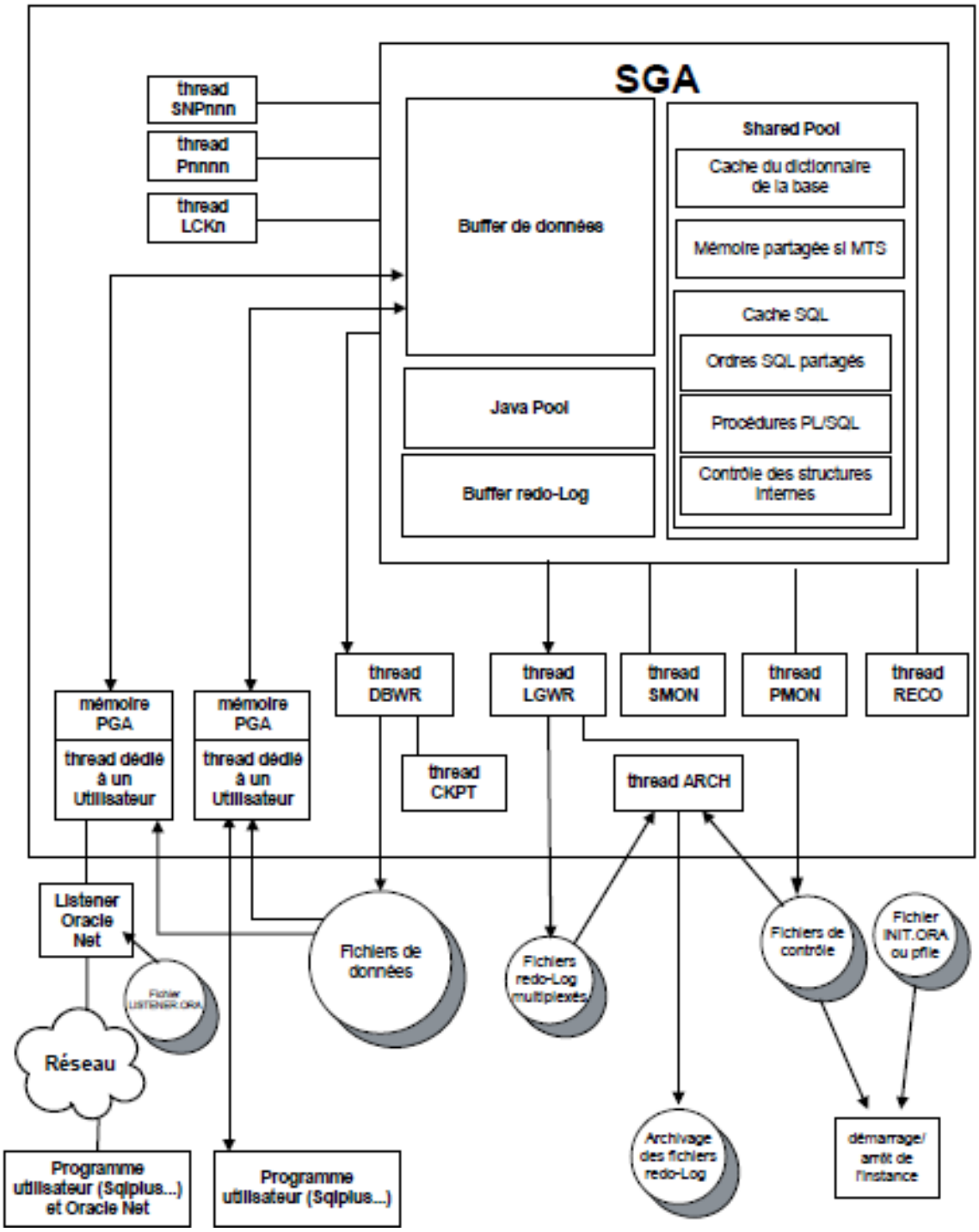
- Permet de stocker plusieurs éléments cruciaux pour la gestion des données :
  - ❖ Library cache : permet d'analyse l'ordre d'exécution d'une requête SQL et de définir un plan d'exécution.
  - ❖ Dictionary cache : stocke les données en provenance du dictionnaire afin d'accélérer l'accès au dictionnaire (nom d'utilisateurs, privilèges, etc.).
  - ❖ SQL area : stocke les requêtes SQL les plus récemment utilisées par les utilisateurs de base de données.
  - ❖ Si la même requête est ré-exécutée, le serveur n'analyse pas son ordre. Cela permet d'améliorer la performance des applications
- Le fonctionnement d’une base Oracle est assuré par un ensemble de processus imbriqués qui réalisent de nombreuses actions. Pour plus de simplicité, nous avons regroupé les processus en deux familles :
  - ❖ les indispensables,
  - ❖ les optionnels.

I.2 Architecture - Instance : Processus en arrière-plan

- Les processus indispensables sont présents dès qu’une base Oracle fonctionne. Ils sont requis pour en assurer le fonctionnement minimal. Si l’un d’eux s’arrête, la base de données n’est plus opérationnelle.
- D’autres processus peuvent être lancés pour assurer des fonctions complémentaires, comme l’archivage des redo log. Si l’un de ces processus optionnels n’est pas démarré, cela ne met pas en cause le fonctionnement global de la base de données. Seule la tâche assurée par ce processus optionnel ne sera pas réalisée.

Les processus indispensables

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



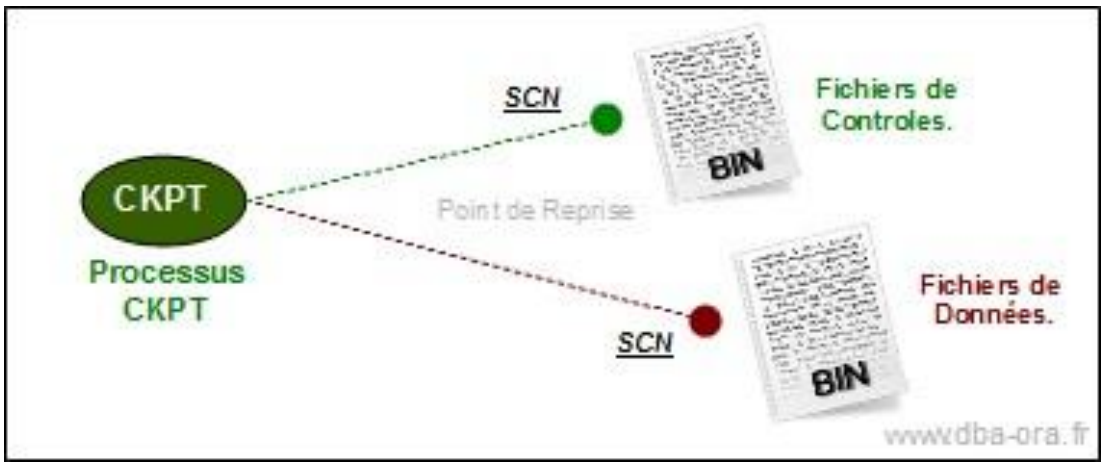
- DBWR (Data Base WRiter)
  - ❖ Ecrit les blocs modifiés dans le cache de données sur les disques.
  - ❖ DBWn se déclenchera lors des événements suivants :
    - Lorsque le nombre de bloc *dirty* atteint une certaine limite

- Lorsqu'un processus sera à la recherche de blocs libres dans le *Database Buffer Cache*, et qu'il ne sera pas en mesure d'en trouver.
- Lors de timeouts (environ toutes les 3 secondes par défaut)
- Lors d'un *checkpoint*.

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

- LOGWR (LoG WRiter)
  - ❖ Ecrit dans les fichiers Redo Log le contenu du cache Redo Log
  - ❖ Le processus LGWR transcrit les informations contenues dans le *REDO LOG Buffer* vers les fichiers *REDOLOG FILE* quand :
    - une transaction s'est terminée avec un COMMIT
    - le REDO LOG Buffer est au 1/3 plein
    - Avant que DBWn n'écrive le contenu du Database Buffer Cache dans les fichiers du disque dur

- CKPT (CHECKPOINT)
  - ❖ **Checkpoint** inscrit les informations de point de reprise dans les fichiers de Controles et dans l'entête de chaque fichier de données. C'est ce point de reprise (**SCN**) qui permet de rendre cohérent les fichiers de controles et les fichiers de données, indispensable pour un processus de récupération.



- CKPT (CHECKPOINT)

- ❖ Le processus Checkpoint n'écrit pas les blocs sur le disque, c'est le role du processus Database Writer DBWn.
    - Provoque l'activation du DBWR pour écrire les blocs modifiés depuis le dernier point de contrôle,
- ❖ Les numéros SCN enregistrés dans les fichiers garantissent que toutes les modifications apportées aux blocs de base de données avant un numéro SCN ont été écrites sur le disque.
    - En cas d'arrêt anormal de l'instance, ce SCN marque le début des données à utiliser pour la récupération de l'instance.

- CKPT (CHECKPOINT)
    - ❖ Le processus Log Writer (LGWR) n'écrit pas dans le fichier de journalisation (REDO LOG) tant que le processus CKPT n'a pas synchronisé, car tant que cette synchronisation n'est pas faite, le fichier de journalisation contient des données nécessaires à une éventuelle récupération après défaillance de l'instance.
    - ❖ A savoir qu'une synchronisation se déclenche aussi à chaque basculement de REDO LOG, lors de l'arrêt de la base de données, lors de la mise hors ligne d'un Tablespace.
- SMON (System MONitor)
    - ❖ **System Monitor** surveille la base de données lors de son démarrage c'est à dire faire la récupération de l'instance après un arrêt anormal. Lors du démarrage de l'instance Oracle, il vérifie si le dernier arrêt a été correctement effectué.
    - ❖ **SMON** est aussi chargé de:
        - libérer les segments temporaires
        - compacter l'espace contigu dans les Tablespaces
        - surveiller la base de données.
- Le processus SMON en action lors d'un arrêt brutal de la base de données
    - ❖ **SMON** lit les informations contenu dans les segments **UNDO** (données en attente de validation) puis les annule. (**ROLL BACK**)
    - ❖ **SMON** récupère dans les fichiers **REDO LOG** les enregistrements validés mais pas encore écrit dans les fichiers de données et les insère. (**ROLL FORWARD**).
    - ❖ **SMON** libère toute les ressources de la base de données (vérrous, segments temporaires).

- Le **processus System Monitor SMON** en action lors d'un fonctionnement normal de la base de données
    - ❖ **SMON** surveille l'activité de la base de données.
    - ❖ **SMON** recycle les segments temporaires et assure les espaces de tris.
    - ❖ **SMON** libère toute les ressources de la base de données (vérrous, segments temporaires).
    - ❖ **SMON** peut être appelé par d'autre Processus pour libérer de l'espace.
- PMON (Process Monitor)
    - ❖ Ce processus gère les ressources utilisateurs
    - ❖ Si un incident survient (arrêt brutal du poste client)
        - ○ PMON annule les transactions en cours
        - ○ Il supprime les verrous posés sur les tables et les lignes
        - ○ Et libère les ressources utilisées

I.3 Les processus optionnels

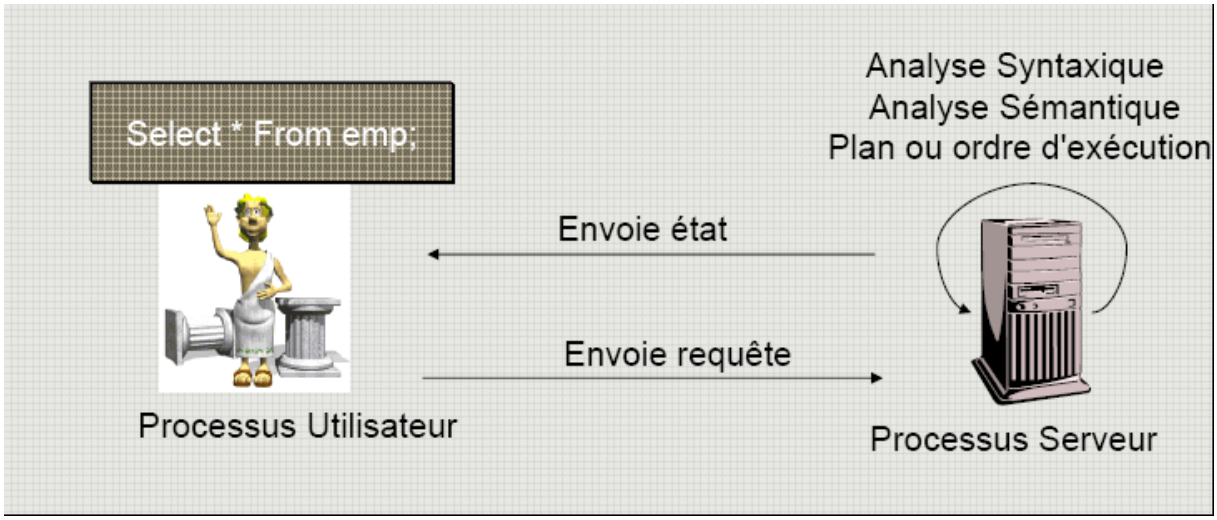
- Les processus détachés optionnels sont les suivants :
- Listener ou listener Net ;
- ARCH ou Archiver ;
- RECO ou Recover ;
- Dnnnn et Snnnn Dispatcher et Server ;
- ....

I.4 Les processus server

- Le rôle des processus server est fondamental. Dès que l'application demande l'accès à l'instance, ce sont eux qui vérifient que les données sont présentes en mémoire.
    - ❖ Dans l'affirmative, le processus server traite vos ordres SQL.
    - ❖ Dans la négative, les processus server lisent les fichiers de la base de données pour les « monter » en mémoire SGA, avant de traiter l'ordre SQL.
- Ce n'est pas un processus permanent de l'instance
    - ❖ le processus server est créé lors de chaque connexion
- Il assure la lecture des données contenues dans vos fichiers.
- Il assure le lien entre le processus client, les processus d'arrière plan, la SGA et les fichiers qui composent la base.

I.5 Traitement d'une requête d'interrogation

- Le processus client envoie la requête au processus serveur
- Le processus serveur effectue l'analyse syntaxique et sémantique de la requête
  - ❖ utilise la shared pool de la SGA pour compiler l'ordre
  - ❖ retourne l'état (analyse correcte ou incorrecte) au processus client.
- Exécution de la requête
- Récupération des résultats
  - ❖ le processus serveur envoie les lignes extraites par la requête (à partir du cache de données si les données y ont déjà été chargées)



1.6 Traitement d'une requête de MAJ

- Similaire à une requête d'interrogation pour la phase d'analyse
- Phase d'exécution différente :
  1. le processus serveur lit des blocs de données et de rollback à partir
     - ❖ des fichiers de données
     - ❖ du buffer cache de données
  2. Si des blocs ne sont pas déjà dans le cache, copie des blocs du disque dans le cache
  3. Mise à jour de verrou sur les données
  4. Enregistrement des modifications
     - ❖ à apporter au rollback (image avant)

- ❖ et aux données (nouvelles valeurs) dans le cache de reprise (redo log)

5. Mêmes opérations dans le cache de données
    - ❖ ces blocs sont marqués comme modifiés (différents des blocs stockés sur disque)

Chapitre 2

Architecture – Base de données : Structure Physique de stockage

II. Architecture – BDD : Structure physique de stockage

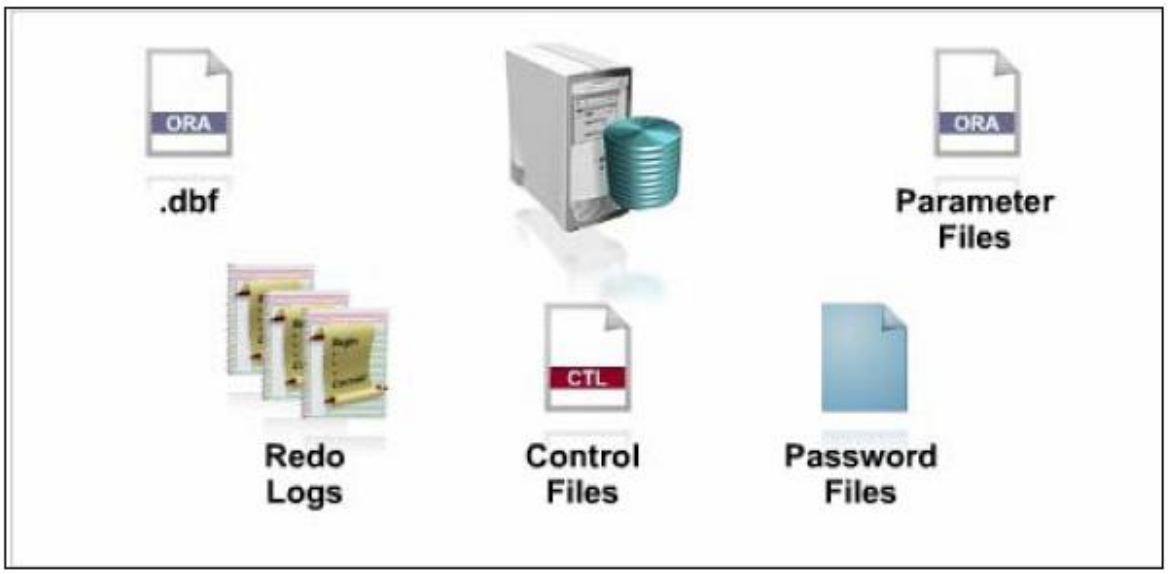
- Base de Données Oracle
  - ❖ Structure Physique de stockage
    - Fichiers de données
    - fichiers Redo Log
    - Fichiers de Contrôles
    - Autres fichiers
  - ❖ Structure Logique de stockage
    - Tablespace
    - Segment
    - Extent
    - Bloc de données

➔ Nécessaire de comprendre ces 2 niveaux de représentation

BD Oracle : Rappel

- Se compose de fichiers du SE
  - ❖ fichiers de données (>=2)
    - stocke le dictionnaire des données, les objets utilisateurs …
  - ❖ fichiers de reprise (redo log) (>=2)
    - Stocke toutes les modifications apportées à la base de données (pour la reconstruire en cas de panne)
  - ❖ fichiers de contrôle (>=1)
    - contient les informations nécessaires à la mise à jour et à la vérification de l'intégrité des données.
  - ❖ Autres fichiers :
    - fichier de paramètres, fichier mot de passe, fichiers de reprise archivés …

Architecture – structure Physique de la BDD



II.1 Fichiers de données

select * from dba_data_files;

Présentation du dictionnaire de données

- Le dictionnaire de données est un ensemble de tables et de vues où est stockée l'information sur la structure logique et physique de la BDD:
  - ❖ Les utilisateurs des bases de données
  - ❖ Noms et caractéristiques des objets stockés dans la base de donnée
  - ❖ Contraintes d'intégrités
  - ❖ Ressources physiques allouées à la base

- …

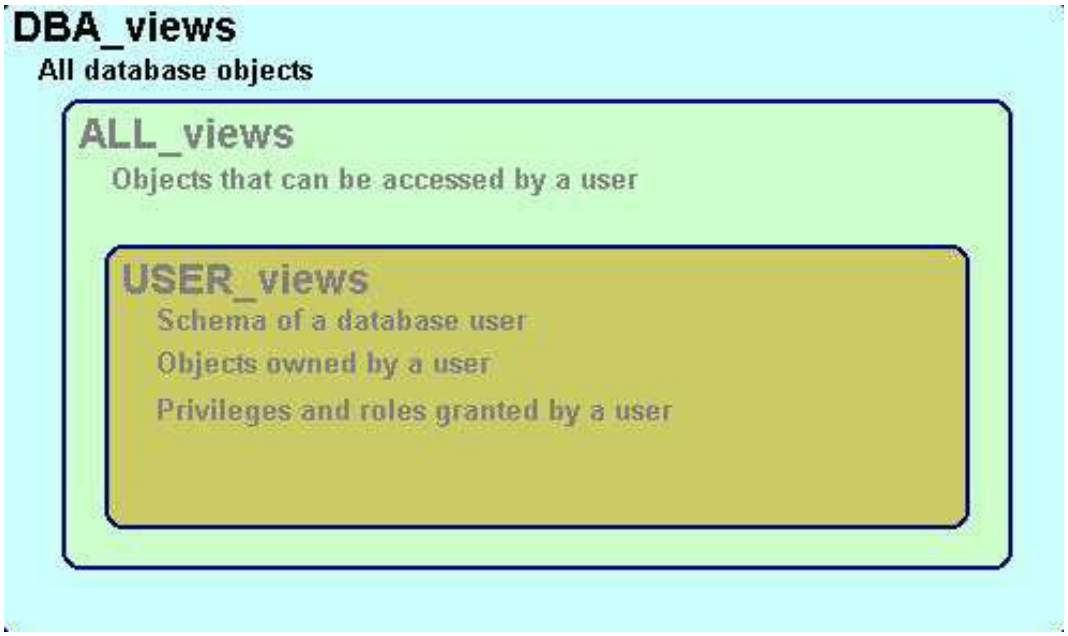
- Le dictionnaire de données est créé à la création de la BDD, mis à jour au fur et à mesure de la création d'objets.
- Les utilisateurs des bases de données, les développeurs d'applications, les administrateurs de bases de données et le serveur Oracle utilisent le dictionnaire de données comme une source centrale d'information associée à une base de données.
- Le dictionnaire de données possède deux composants :
  - les tables de base
  - les vues du dictionnaire de données

1. Les tables de base sont les tables réelles d'Oracle qui stockent les informations sur une base de données.
   - Ces tables sont créées avec le script sql.bsq. Ce script est stocké dans le répertoire ORACLE_HOME\rdbms\admin. Les informations stockées dans les tables de base sont lues et écrites par le serveur Oracle. Ces informations sont rarement accédées directement par les utilisateurs car ces informations sont normalisées et stockées sous une forme encodée.
   - Les utilisateurs ou administrateurs de bases de données ne doivent pas utiliser de commandes LMD, telles que INSERT, UPDATE et DELETE, pour mettre à jour les tables de base directement, à l'exception de la table de trace d'audit lorsque la fonctionnalité d'audit est utilisée.

1. Les vues du dictionnaire de données sont des vues sur les tables de base.
   - Elles sont crées par le script catalog.sql.
   - Les vues du dictionnaire de données simplifient et résument les informations contenues dans les tables de base.
   - Les vues du dictionnaire stockent également ces informations sous une forme que les utilisateurs de la base de données peuvent lire facilement.
   - Ces vues permettent au DBA de gérer et d'administrer la base de données.

- Les utilisateurs n'ont pas accès à ce dictionnaire éventuellement en le lecture à traves trois catégories des vues:
  1. USER_<vues> : Vues sur les objets créé par un utilisateur
     - Par exemple, la vue USER_TABLES contient les informations sur les tables appartenant à un utilisateur.
  2. ALL_<vues>: Vues sur les objets auxquels un utilisateur a accès (pas nécessairement qu'il a créés(à travers l'obtention publique ou explicite de rôles et de privilèges)

3. DBA_<vues>: Vues sur tous les objets de la base de données de tous les utilisateurs accessible par les administrateur(DBA) ou les utilisateurs qui possèdent le privilège système SELECT ANY TABLE.



Vues	Description
dictionary dict_columns	Vues générales
dba_tables dba_objects dba_lobs dba_tab_columns dba_constraints	Informations sur les objets, tels que les tables, les contraintes, les gros objets et les colonnes
dba_users dba_sys_privs dba_roles	Informations sur les privilèges et les rôles des utilisateurs
dba_extents dba_free_space dba_segments	Allocation d'espace pour les objets de la base de données
dba_rollback_segs dba_data_files dba_tablespaces	Structures générales de la base de données
dba_audit_trail dba_audit_objects dba_audit_obj_opts	Informations d'audit

II.2 Fichiers Redo Log

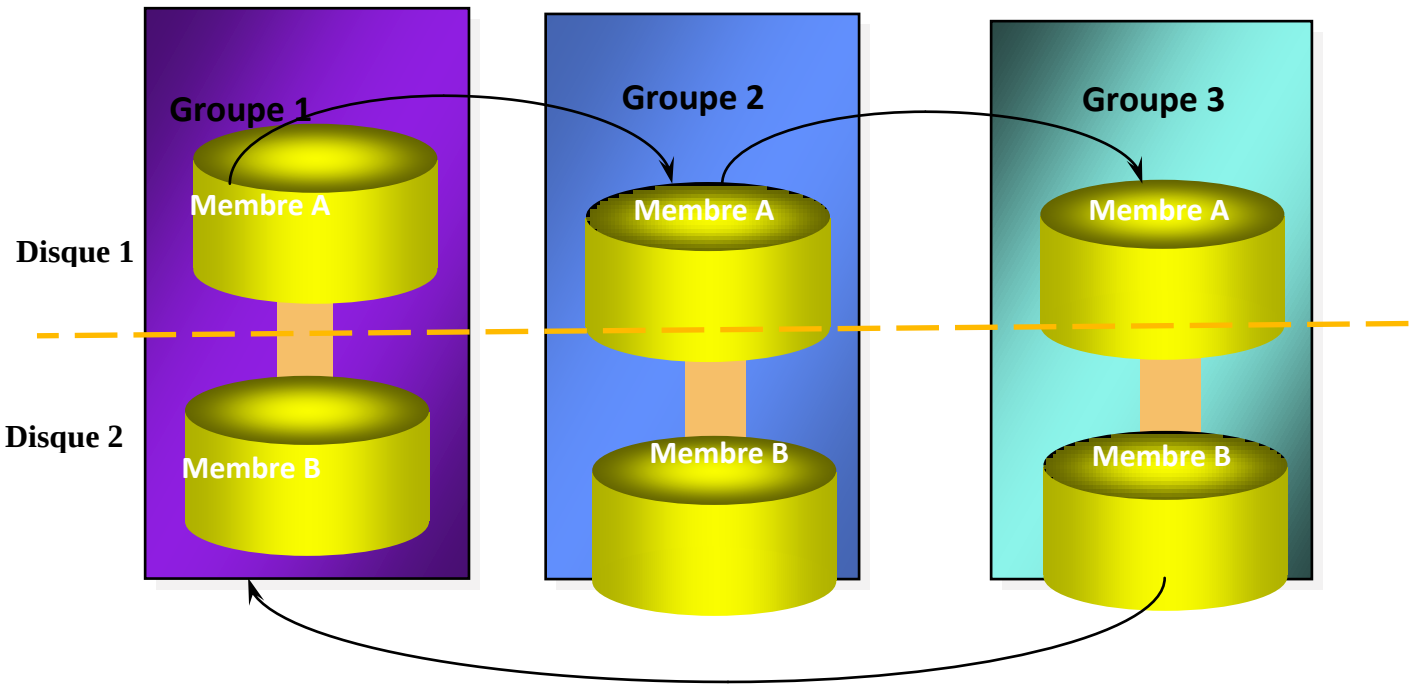
Fichiers Redo Log

- L'idée de base enregistrer toutes les modifications apportées aux données pour minimiser les problèmes liés aux pannes
- Le serveur Oracle met à jour les fichiers de reprise
  - ❖ buffer de reprise
  - ❖ processus LGWR
- Ne sont utilisés qu'en cas d'échec d'une instance pour restaurer des données validées non écrites dans les fichiers de données
- Groupes de fichiers de reprise :
  - ❖ un membre est un élément d'un groupe, i.e. un fichier de reprise
  - ❖ LGWR écrit simultanément sur chaque membre du groupe
  - ❖ tous Les membres d'un groupe redo log en ligne possèdent exactement les mêmes informations
  - ❖ Les membres d'un groupe sur des disques différents (au moins 2 groupes)
  - ❖ tous les membres d'un groupe possèdent un numéro de séquence log sert d'identifiant
  - ❖ le numéro courant est stocké dans le fichier de contrôle

    select * from v$logfile;

    select * from v$log;

Les Fichiers Redo Log Multiplexés



- Ajouter un groupe de fichiers de reprise online:

  ALTER DATABASE [*database]*

  ADD LOGFILE [GROUP *valeur] 'filename' [SIZE n[K|M]] [REUSE]*

  [,[GROUP *valeur] 'filename' [SIZE n[K|M]] [REUSE]]...;*

  Exemple

  SQL> ALTER DATABASE

  ADD LOGFILE GROUP 3

  ('c:\orant\database\logorcl3.ora') SIZE 1000K;

- **Ajout des membres redo log online**

  ❖ Des membres redo log peuvent être ajoutés grâce à la commande SQL suivante :

  ALTER DATABASE [*database]*

  ADD LOGFILE MEMBER '*filename' [REUSE]*

  TO GROUP *n;*

  Exemple

  SQL> ALTER DATABASE

ADD LOGFILE MEMBER 'e:\orant\database\log7borcl.ora'TO GROUP 7;

- **Suppression de groupes de fichiers redo log online**
  - ❖ Pour améliorer les performances de la base de données, il peut s'avérer nécessaire d'augmenter ou de diminuer la taille des groupes de fichiers redo log online. Pour changer la taille d'un groupe de fichiers redo log online, il faut créer un nouveau groupe de fichiers redo log online et ensuite supprimer le vieux groupe de fichiers redo log online.

    ALTER DATABASE [*database]*

    DROP LOGFILE GROUP *n;*

- Un certain nombre de restrictions sont à prendre en compte lors de la suppression de groupes redo log online :
  - ❖ L'instance doit avoir au moins deux groupes de fichiers redo log online. Un groupe de fichiers redo log online ne pourra pas être supprimé si il n'existe que deux groupes. Pour supprimer un groupe, il doit en rester au moins trois.
  - ❖ Un groupe redo log online actif ne peut pas être supprimé.
- Quand un groupe redo log est supprimé, les fichiers du système d'exploitation ne sont pas supprimés automatiquement. Il est donc nécessaire de supprimer manuellement les fichiers redo log online afin de garder un espace disque propre.
- **Emplacement des fichiers redo log online**
  - ❖ La disponibilité des fichiers redo log online détermine la performance d'une base de données Oracle.
    - Pour améliorer la performance de la base de données, les membres des groupes de fichiers redo log online, les fichiers log archivés et les fichiers de données doivent être stockés sur des disques séparés.
    - Le stockage des fichiers redo log et des fichiers de données sur différents disques permet de réduire de façon substantielle le risque de perdre à la fois les fichiers de données et les fichiers redo log online lors d'une défaillance du support.
- **Archivage de fichiers de redo log**
  - ❖ Une des décisions importantes qu'un DBA doit prendre est configurer la base de données dans le mode :
    - ARCHIVELOG
    - NOARCHIVELOG
  - ❖ **NOARCHIVELOG :**

- En mode, NOARCHIVELOG, les fichiers de redo log online sont réécrits à chaque fois qu'un fichier de redo log online est rempli et qu'un log switch est lancé.

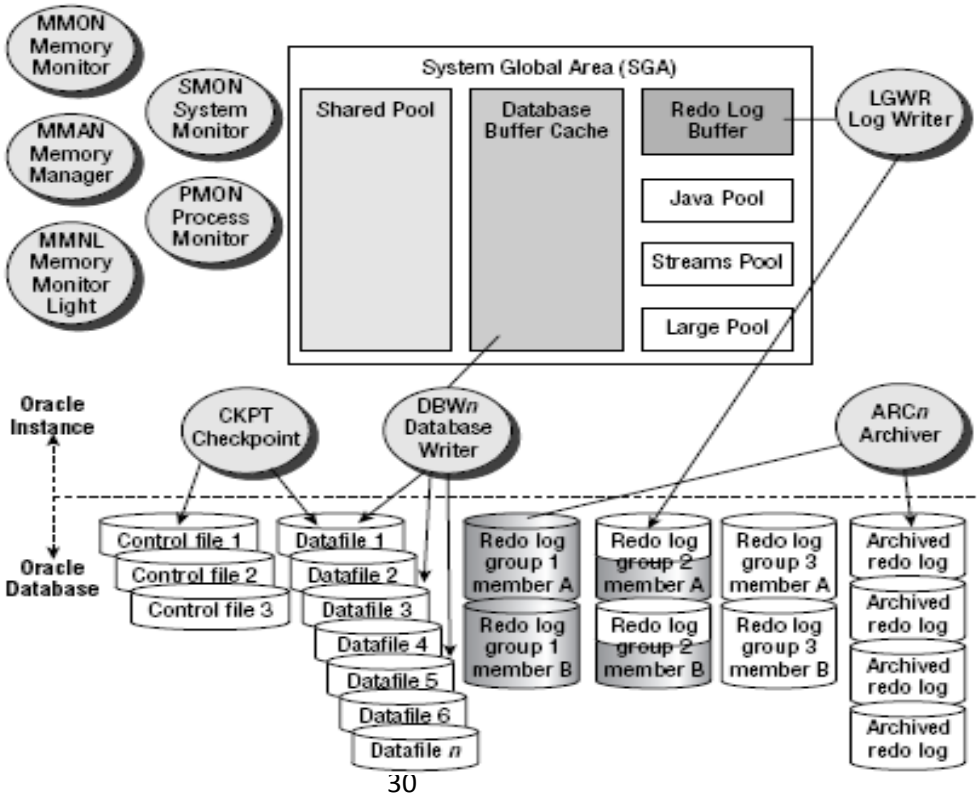
❖ **ARCHIVELOG :**

- Si la base de données est configurée dans le mode ARCHIVELOG, alors les groupes de redo remplis doivent être archivé. Comme toutes les modifications faites sur la base de données sont enregistrées dans les fichiers de redo log online, le DBA peut utiliser la sauvegarde présente sur le disque dur et les fichiers de redo log archivés pour restaurer la base de données sans perdre aucune donnée comité.

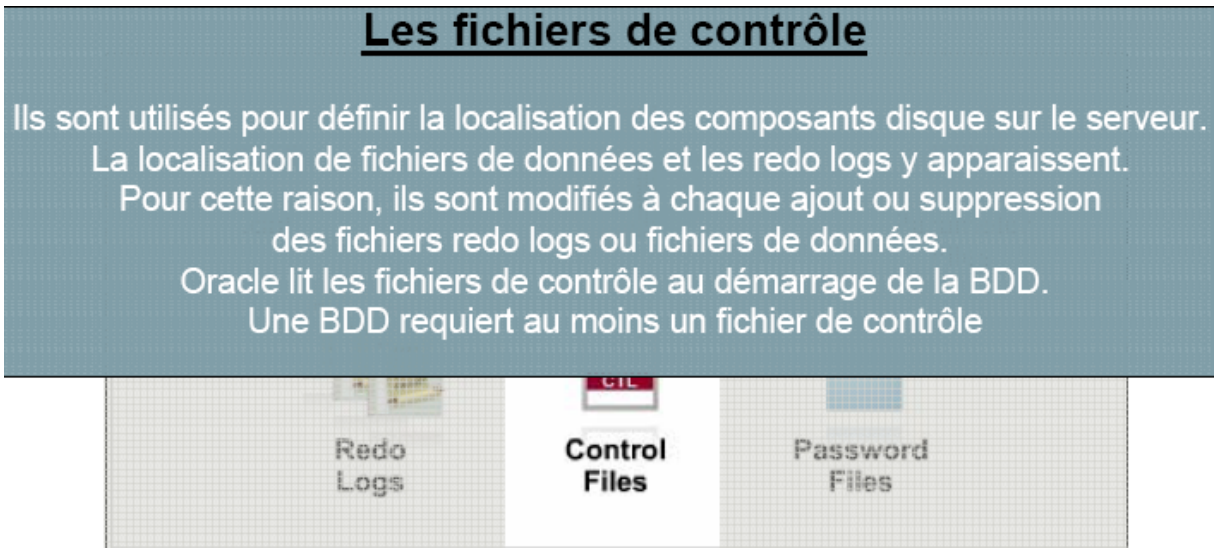
❖ On peut archiver les fichiers de redo log :

- Manuellement.
- Automatiquement : Méthode recommandée.
- Le paramètre d'initialisation LOG_ARCHIVE_START, lorsqu'il est à
  - ☐ TRUE indique que l'archivage se fait automatiquement.
  - ☐ A FALSE, indique que le DBA le fait manuellement.
- En mode automatique, l'archivage se fait grâce au processus ARCn, et manuellement avec des requêtes SQL.

FIGURE 1.11 The Oracle 10*g* architecture



II.3 Fichiers de Contrôles



- Fichier binaire
- Sert pour la base de données
  - ❖ lors du démarrage normal
    - A chaque fois qu'une instance monte une base de données, elle lit le fichier de contrôle pour localiser emplacement des fichiers de données et des fichiers de reprise
  - ❖ lors du démarrage après panne
    - Contient les informations nécessaires à la remise en état de la base de données
  - ❖ pour le bon fonctionnement
    - doit être disponible quand une base de données est montée ou ouverte
- Tous les fichiers de données et les fichiers log de la base de données sont identifiés dans le fichier de contrôle.
- Le nom de la base de données est répertorié dans le fichier de contrôle.
- Le fichier de contrôle est requis pour monter, ouvrir, et accéder à la base de données.
- Le numéro de séquence de log actuel. Cette information permet de synchroniser tous les fichiers appartenant à la base de données.
- La configuration recommandée est un minimum de deux fichiers de contrôle sur des disques différents (multiplixage).

- **Multiplexage des fichiers de contrôle**
    - ❖ Dans le but de prévenir une erreur dans un fichier de contrôle, il est fortement recommandé de multiplexer les fichiers de contrôles et de les stocker séparément sur des disques différents.
        - o Si un fichier de contrôle est perdu, une copie du fichier de contrôle peut être utilisé pour redémarrer l'instance. On peut multiplexer jusqu'à 8 fichiers de contrôles.
    - ❖ Modifier le SPFILE (Fichiers de paramètres) :

      ALTER SYSTEM SET control_files ='$HOME/ORADATA/u01/ctrl01.ctl','$HOME/ORADATA/u02/ctrl02.ctl' SCOPE = SPFILE ;
    - ❖ Eteindre la base de données.
    - ❖ Créer les fichiers de contrôles supplémentaires :

      cp $HOME/ORADATA/u01/ctrl01.ctl

      $HOME/ORADATA/u02/ctrl02.ctl
    - ❖ Redémarrer la base de données

Multiplexage du fichier de contrôle

- **Avec le init.ora (**Fichiers de paramètres) **:**
    - ❖ Eteindre la base de données.
    - ❖ Copier le fichier de contrôle existant sur un autre disque et changer sont nom :cp control01.ctl …/DISK3/control02.ctl
    - ❖ Ajouter le nouveau fichier de contrôle à init.ora :CONTROL_FILES = (/DISK1/control01.ctl,

/DISK3/control02.ctl)

    - ❖ Redémarrer la base de données.
    - ❖

Fichier de controles

- Requêtes sur le fichier de contrôle
    - ❖ Select * from V$CONTROLFILE;

Autres fichiers

- Fichiers de paramètres d'initialisation

Chapitre 3

Architecture – Base de données: Structure Logique de stockage

III. Structure logique de stockage

- Objectifs
  - ❖ Faire le lien entre structure physique et logique
    - o via les fichiers de données
  - ❖ Comprendre les moyens logiques de structuration
    - o Segment
  - ❖ Etude de deux segments particuliers
    - o Segment rollback
    - o Segment temporaires

III.1 Tablespace

- L'utilisateur, qu'il soit connecté par une application, ou directement à la BDD par SQL*Plus, ne voit que la structure logique de la BDD.
  - ❖ Quand il envoie une requête il voit que les tables et leurs descriptions
  - ❖ A aucun moment ne fait appel dans sa commande aux fichiers physiques
  - ❖ Le processus serveur qui réceptionne la commande va devoir savoir dans qu'il fichier sont stockées
  - ❖ Le relation entre les tables et les données stockées dans le fichier physique se fait par un objet logique : Le tablespace
  - ❖ Les données d'une base Oracle sont mémorisées dans une ou plusieurs unités logiques appelées tablespaces et physiquement dans des fichiers associés à ces tablespaces.
  - ❖ Un tablespace est la plus grande unité logique de stockage allouable dans oracle.
  - ❖ Un tablespace est composé d'au moins un fichier de données ayant une existence physique (stocké sur disque dur).
- Tout objet du schéma de base de données est crée dans un seul tablespace, mais peut s'étaler sur plusieurs fichiers de données.
- Les tablespaces peuvent être activés (online) ou désactivés (offline) individuellement pour d'éventuelles opérations de maintenance :
  - ❖ Les objets d'un tablespace offline ne sont plus accessibles.

Utilisation des Tablespaces

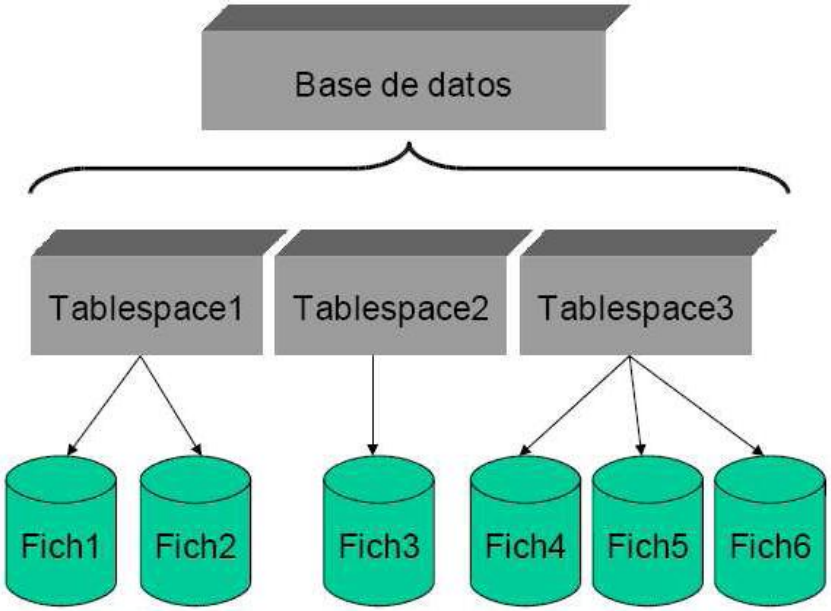
- Contrôle de l'allocation d'espace et affectation de quotas aux utilisateurs
- Contrôle de disponibilité des données par la mise online ou offline des tablespaces
- Distribution du stockage des données sur plusieurs dispositifs
    - ❖ pour améliorer les E/S
    - ❖ pour réduire la contention sur un seul disque
- Exécution de sauvegarde partielle
- Conservations de volumes importants de données statiques sur des dispositifs en lecture seule.

Tablespace

- Appartient à une seule base de données
- Chaque tablespace comprend un ou plusieurs fichiers SE
- Les tablespaces peuvent être mis on/off line lors de l'exécution de la base de données
    - ❖ excepté le tablespace SYSTEM
    - ❖ et un tablespace avec un rollback segment actif
- Possibilité de basculer entre le statut Read/Write et le statut Read seul.

Fichiers de données

- Informations quantitatives :
    - ❖ nombre maximum de fichier par tablespace = 1023
    - ❖ nombre maximum de tablespace par base de données : 64 000



III.2 Segments/Extend/Blocs de données

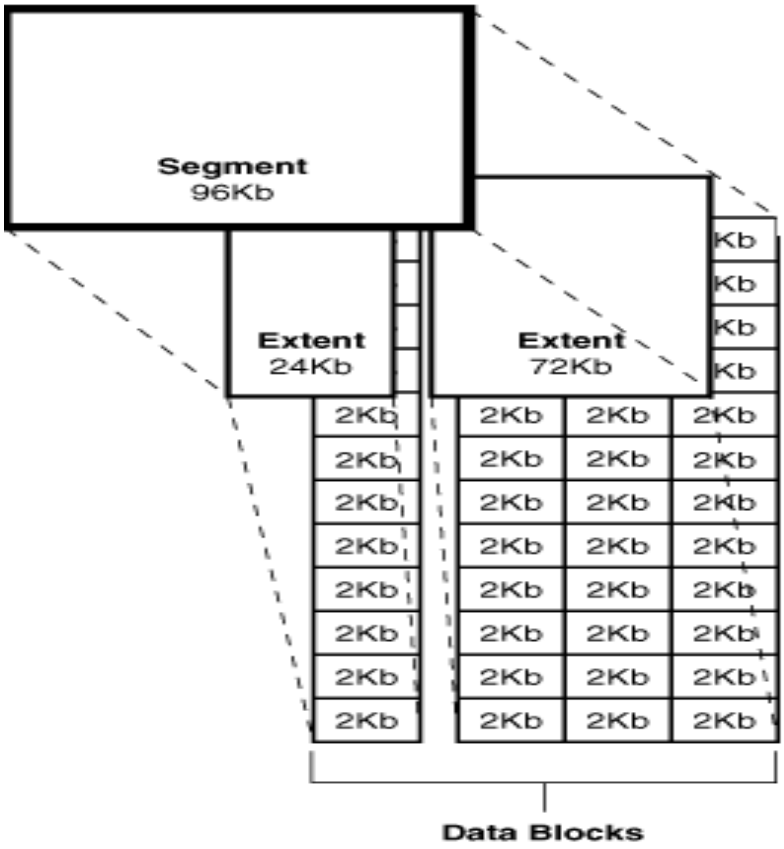
- ❖ Espace alloué à un type spécifique de structure logique de stockage dans un tablespace
    - segments de table, segments d'index, segment temporaire, rollback segment,…

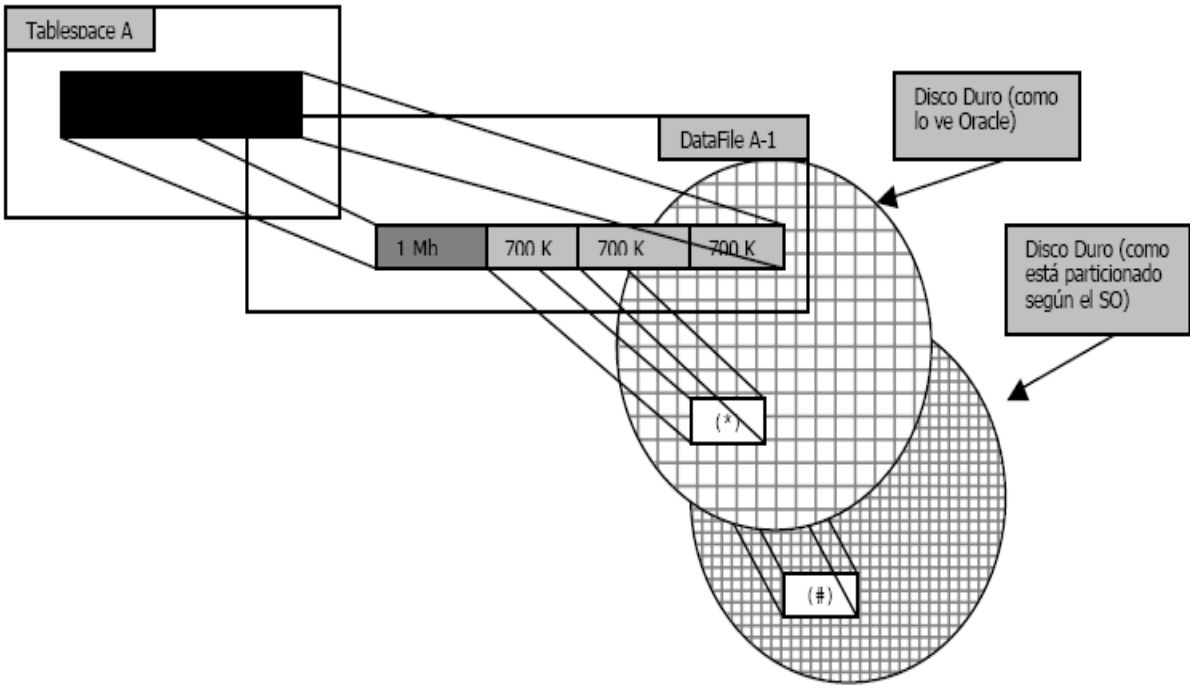
Extents

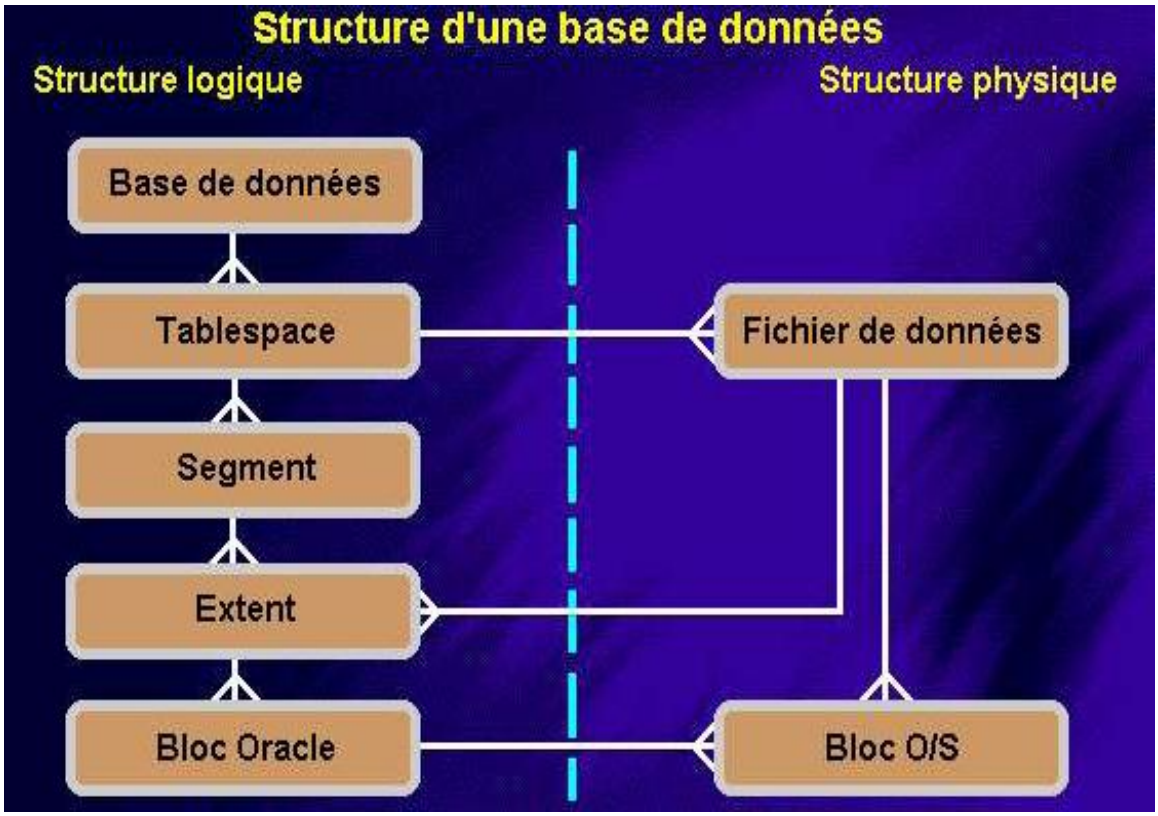
- ❖ Ensemble de blocs contigus.
- ❖ Chaque type de segment est composé d'un ou plusieurs extents

- **Blocs de données**
    - ❖ contient un ou plusieurs blocs de fichier physique alloué à partir d'un fichier de données existant
    - ❖ plus petite unité d'E/S
    - ❖ sa taille vaut DB_BLOC_SIZE

Segments,Extents & bloc (suite)







Les Tablespaces

- **Le tablespace System:**
  - ❖ Indispensable au fonctionnement de la base de donnée.
  - ❖ Il contient les informations du dictionnaire de données, les définitions des procédures stockées, des packages … .
  - ❖ Ne devrait pas contenir de données utilisateurs
- **Les tablespaces non system:**
  - ❖ Permettent plus de flexibilité dans l'administration de la BD.
  - ❖ Composées de segments de données, d'indexes, de rollback et de segments temporaires.

Création d'un tablespace

- Un tablespace est créé à l'aide de la commande
  - ❖ Create tablespace

    Create tablespace tbs_user

    Datafile 'c:/oracle/oradata/userdata01.dbf' size 100M

Autoextend on next 5M maxsize 200M;

- Syntaxe de création d'un tablespace :

  CREATE TABLESPACE *nom*

  DATAFILE spécification_fichier_data [,_]

  [MINIMUM EXTENT valeur [K|M]]

  [ONLINE|OFFLINE]

  [PERMANENT|TEMPORARY]

  [LOGGING|NOLOGGING]

  [EXTENT MANAGEMENT

  DICTIONARY

  |LOCAL [AUTOALLOCATE |UNIFORM [SIZE valeur [K|M]]]]

  [DEFAULT clause _ storage]

  [BLOCKSIZE valeur [K]]

- DATAFILE: spécifie le ou les fichiers de données liés au tablespace. M spécifie la taille en Mo ou Kbits.
- MINIMUM EXTENT: garantit que la taille de chaque extent du tablespace est un multiple de *valeur*
- LOGGING: option par défaut qui spécifie que toutes les tables, indexes et partitions dans le tablespace génèrent des infos de redo log.
- DEFAULT: définit les paramètres de stockage par défaut pour tous les objets créés dans le tablespace.
- OFFLINE: rend le tablespace indisponible immédiatement après la création.
- PERMANENT: spécifie que le tablespace peut être utilisé pour contenir des objets permanents.
- TEMPORARY: cette clause est utilisée pour les objets temporaires.
- MANAGEMENT DICTIONARY: le tablespace est géré avec le dictionnaire de données.
- MANAGEMENT LOCAL: le tablespace est géré localement et une partie de son espace est réservée pour sa gestion.
  - ❖ A la différence des tablespaces standards géré au niveau du dictionnaire de données, la gestion de l'espace physique (allocation / libération de blocs) se fait dans l'entête du fichier(s) du tablespace. Une table binaire d'allocation (bitmap) y est maintenue.

- ❖ Avantages:
  - pas de contention en mise a jour au niveau du dictionnaire
  - …

- AUTOALLOCATE: la taille des extents est gérée par oracle, c'est l'option par défaut
- UNIFORM: il est défini une taille uniforme des extents en k octets ou en M octets . La taille par défaut est de 1 méga Octet

Exemple 1 (tablesapace géré par le dictionnaire de donnée)

```
Create tablespace data
Datafile
  'd:\oracle\oradata\iges5\data01.dbf' size 30M,
  'e:\oracle\oradata\iges5\data02.dbf' size 20M
   autoextend on next 5M maxsize 60M
Extent management dictionary
Minimum extent 150K
Default storage (initial 300K
                 next 300K
pctincrease 0
                 minextents 1
          maxentents 50)
Online;
```

Exemple 2

```
Create tablespace indx
Datafile
  'd:\oracle\oradata\iges5\indx01.dbf' size 100M,
  Extent management local uniform size 128k;
```

Exemple 3

Create temporary tablespace temp

TEMFILE ‘d:\oracle\oradata\iges5\temp01.dbf’ size 100M

autoextend on next 10M maxsize 1000M

extent management local uniform size 4M;

Tablespaces temporaires

- Tablespaces temporaires
  - ❖ Utilisé pour les opérations de tri
  - ❖ Ils ne peuvent pas contenir d’objets permanents
  - ❖ La gestion local des exents est recommandée
  - ❖ Exemple :

    CREATE TEMPORARY TABLESPACE temp

    TEMPFILE ‘/DISK2/temp01.dbf’ SIZE 500M

    Extent management local uniform size 4M;

Les Undo Tablespaces

- Les tablespaces de undo (tablespaces d’annulation) sont une nouveauté de Oracle 9i.
  - ❖ Ils sont utilisés uniquement pour stocker les segments de undo (annulation)
  - ❖ Il ne peut contenir aucun autre objet
  - ❖ Il ne peut être utilisé qu’avec la clause datafile

    CREATE UNDO TABLESPACE undo1

    DATAFILE ‘u01/oradate/undo101.dbf’ SIZE 40M;
- Les informations d’utilisation des segments de UNDO sont stockées dans la vue V$UNDOSTAT.

Augmenter la capacité de stockage

- Augumenter la taille d’un tablespace
  - ❖ Ajouter des fichiers de données au tablespace

    ALTER TABLESPACE data

    ADD DATAFILE

    ‘d:\oracle\oradata\iges5\data03.dbf' SIZE 200M;
  - ❖ Automatiquement avec l’activation de AUTOEXTEND

ALTER TABLESPACE data

ADD DATAFILE

‘d:\oracle\oradata\iges5\data02.dbf' SIZE 200M

AUTOEXTEND ON NEXT 10M MAXSIZE 500M;

- ❖ Redimensionner manuellement un fichier de données

  ALTER DATABASE

  DATAFILE ‘d:\oracle\oradata\iges5\data02.dbf' RESIZE 60M;

Statut OFFLINE

- Les données d'un tablespace offline ne sont plus accessibles
  - ❖ L’administrateur peut sauvegarder cette partie des données sans arrêter le fonctionnement du reste de la base
  - ❖ Ne peuvent jamais être mis offline
    - o le tablespace SYSTEM
    - o tout tablespace avec des rollbacks segments actifs
- Exemple :

  ALTER TABLESPACE data OFFLINE;

  Pour rendre de nouveau le tablespace online :

  (ALTER TABLESPACE data ONLINE;)

Déplacement de fichiers de données

- Fermer la base de données (Shutdown)
- Utiliser les commandes de système d’exploitation pour déplacer le fichier de données
- Monter la base (Mount)
- Exécuter la commande ALTER TABLESPACE RENAME :

  ALTER TABLESPACE data

  RENAME DATAFILE

  ‘d:\oracle\oradata\iges5\data02.dbf ' TO

  'd:\data\DISK5\data02.dbf‘;

- Ouvrir la base

Lecture seule

- Empêche toute opération d'écriture sur les fichiers de données
- Permet de faciliter l'administration d'un sous-ensemble stable des données de l'application
    - ❖ données peuvent résider sur CDROM
    - ❖ Exemple :

      ALTER TABLESPACE data

      READ ONLY;
    - ❖ Pour rendre de nouveau accessible en modification de données :

      ALTER TABLESPACE data

      READ WRITE;

Suppression d'un tablespace

- Un tablespace ne peut être supprimé:
    - ❖ S'il s'agit du tablespace SYSTEM
    - ❖ S'il possède des segments actifs
- Syntaxe :

DROP TABLESPACE nom

[INCLUDING CONTENTS

[CASCADE CONSTRAINTS]] ;

- ❖ INCLUDING CONTENTS : supprime tous les objets (segments) du tablespace
- ❖ CASCADE CONSTRAINTS : supprime les contraintes d'intégrité référentielle des tables définies dans un autre tablespace qui référencent une table du tablespace à supprimer

- Un tablespace contenant encore des données ne peut être supprimé sans l'option INCLUDING CONTENTS
- Ne supprime que les pointeurs de fichiers du fichier de contrôle => les fichiers de données existent toujours

- Pour supprimer les fichiers de données associés à un tablespace il faut ajouter AND DATAFILES
- **Exemple**

  DROP TABLESPACE data

  INCLUDING CONTENTS AND DATAFILES

  CASCADE CONSTRAINTS ;

Informations sur les tablespaces

- Requêtes sur la vue DBA_TABLESPACES
  - ❖ TABLESPACE_NAME
  - ❖ NEXT_EXTENT
  - ❖ MAX_EXTENTS
  - ❖ PCT_INCREASE
  - ❖ MIN_EXTLEN
  - ❖ STATUS
  - ❖ CONTENTS
- Informations sur les fichiers de données v$datafile

Requêtes sur la vue DBA_DATA_FILES

- ❖ FILE_NAME
- ❖ TABLESPACE_NAME
- ❖ BYTES
- ❖ AUTOEXTENSIBLE
- ❖ MAXBYTES
- ❖ INCREMENT_BY

- Information sur les fichiers temporaires :
  - ❖ Dba_temp_files
  - ❖ V$tempfile
- Informations sur les fichiers de données et les tablespaces à partir du fichier de contrôle

- ❖ Jointure entre V$DATAFILE et V$TABLESPACE sur l'attribut TS#

Quelques conseils

- Séparez les données utilisateurs des données du dictionnaire
- Séparez les données utilisateurs suivant les applications qui les utilisent => notion de charge
    - ❖ Afin de diminuer la contention E/S, placer les fichiers des tablespaces différents sur des disques distincts
- Créer tablespace UNDO

Chapitre 4

Gestion de l’instance

IV. Gestion d'une instance Oracle

Objectifs de chapitre :

- Créer et gérer des fichiers de paramètres d'initialisation.
- Démarrer et arrêter une instance.
- Modifier le mode de la Base de Données.
- Restreindre les connexions.
- Surveiller et utiliser des fichiers de diagnostic.
- Les utilisateurs administrateurs de la base de données sont responsables de la gestion et de l'administration du serveur Oracle. Des privilèges particuliers sont requis pour permettre de faire ces tâches.
- Deux comptes utilisateur DBA sont créés automatiquement et ont le rôle DBA :
- **Le compte SYS : Créé lors de l'installation d'Oracle avec tous les privilèges systèmes.**
    - ❖ Mot de passe par défaut : change_on_install, à changer après l'installation.
- **Le compte SYSTEM : Créé lors de l'installation d'Oracle avec tous les privilèges systèmes.**
    - ❖ Mot de passe par défaut : manager à changer après l'installation.

VI.1 Fichiers de paramètres d'initialisation

- Pour démarrer une instance, le serveur oracle doit lire le fichier de paramètres d'initialisation.
    - ❖ Connect / as sysdba
    - ❖ Startup
- Existe deux types de fichiers de paramètres d'initialisation:
    - ❖ Fichier de paramètre statique, PFILE, généralement nommé initSID.ora
    - ❖ Fichier de paramètres persistant, SPFILE, généralement nommé spfileSID.ora
- Une instance peut présenter plusieurs fichiers de paramètres d'initialisation
    - ❖ Pour optimiser les performances dans certaines situations.

Contenu des fichiers de paramètres d'initialisation

- Les paramètres les plus intéressants sont :

  instance_name=IGES5

  db_name=IGES5

  control_files=("D:\oracle\oradata\GI4\CONTROL01.CTL", "D:\oracle\oradata\GI4\CONTROL02.CTL", "D:\oracle\oradata\GI4\CONTROL03.CTL")

  db_block_size=8192

  db_cache_size=175112192

  java_pool_size=20971520

  shared_pool_size=57671680

- Ces paramètres sont traditionnellement stockés dans le fichiers init.ora associé à l'instance(appelé pfile : parameter file).

  $ORACLE_home\admin**DB_NAME**\pfile\init.ora

Fichiers PFILE initSID.ora

- Il s'agit d'un fichier texte
- Il peut être modifié à l'aide d'un éditeur du Système d'exploitation
- Toute modification est apportée manuellement
- Les modifications sont apportées au démarrage suivant

Créer un fichier PFILE initSID.ora

- Créer ce fichier à partir d'un exemple de fichier init.ora
  - ❖ Oracle universel installer installe un exemple de fichier
  - ❖ Copiez l'exemple à l'aide des commande appropriée du SE
  - ❖ Identifiez-le de façon unique à l'aide d'un SID de base de données
    - o Cp init.ora $oracle_home/dbs/initdba01.ora
- Modifiez le fichier InitSID.ora
  - ❖ Editez les paramètres
  - ❖ Affectez des valeurs qui répondent aux besoins de la BDD

Volatilité des paramètres d'initialisation & spfile

- Si un paramètre était modifié via la commande alter uniquement, au prochain redémarrage de l'instance les changements sont perdus à moins d'avoir modifié manuellement init.ora
    - ❖ Alter system set undo_tablesapce='UNDO2'
    - ❖ Solution : Fichiers de paramètres serveur : spfile Format binaire avec paramètres d'initilaisation non volatils.
- $ORACLE_HOME\database\spfile_SID.ora

Fichiers SPFILE SpfileSID.ora

- Il s'agit d'un fichier binaire
- Sa mise à jour est effectuée par le serveur oracle
- Il réside toujours côté serveur
- Il permet de rendre les modifications persistantes après l'arrêt et le redémarrage

Créer un fichier SPFILE

- Créez un fichier SPFILE à partir d'un fichier PFILE

  SQL> CREATE SPFILE ='$oracle_home/dbs/SPFILEdb01.ora'

  from PFILE ='$oracle_home/admin/DB_NAME/pfile/initDB01.ora'
- Il peut être exécuté avant ou après le démarrage de l'instance (Privilège **SYSDBA**)

Modifier des paramètres du fichier SPFLE

- Utilisez la commande ALTER SYSTEM pour apporter des modifications aux valeurs de paramètres :
    - ❖ Alter system set undo_tablesapce='UNDO2'
    - ❖ Indiquez si ces modifications sont temporaires ou persistantes :
        - o Alter system set undo_tablespace='UNDO2'

          scope=BOTH
        - o Scope=memory|spfile|both
            - ▪ Memory : modifier la valeur du paramètre uniquement dans l'instance en cours
            - ▪ Spfile: modifier la valeur du paramètre uniquement dans l'instance en cours

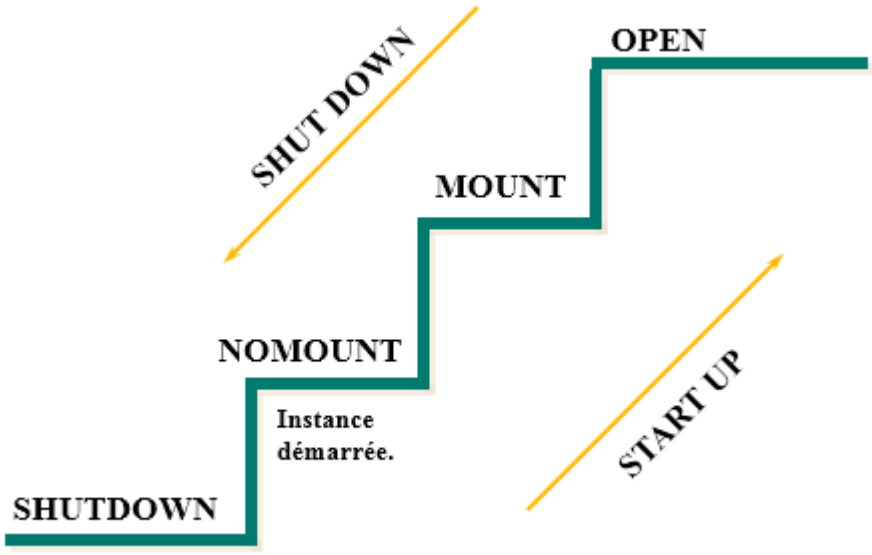
- Both :modifier la valeur du paramètre dans l'instance en cours et le spfile

Fonctionnement de la commande STARTUP

- Ordre de priorités:
  - spfileSID.ora
  - SPFILE par défaut
  - initSID.ora
  - PFILE par défaut
- Vous pouvez modifier ces priorités si vous indiquez:
  - Startup pfile= $oracle_home/admin/DB_NAME/pfile/initDB01.ora

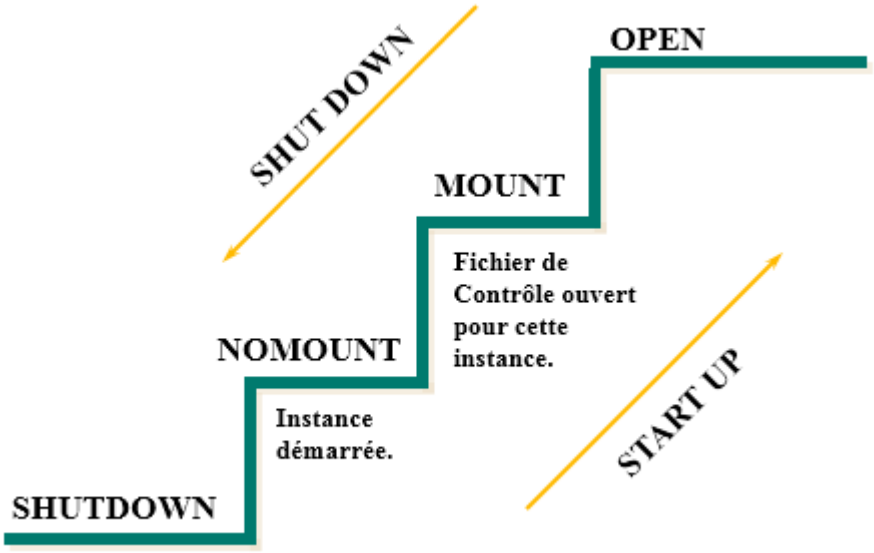
VI.2 Démarrer une base de données

Mode NOMOUNT



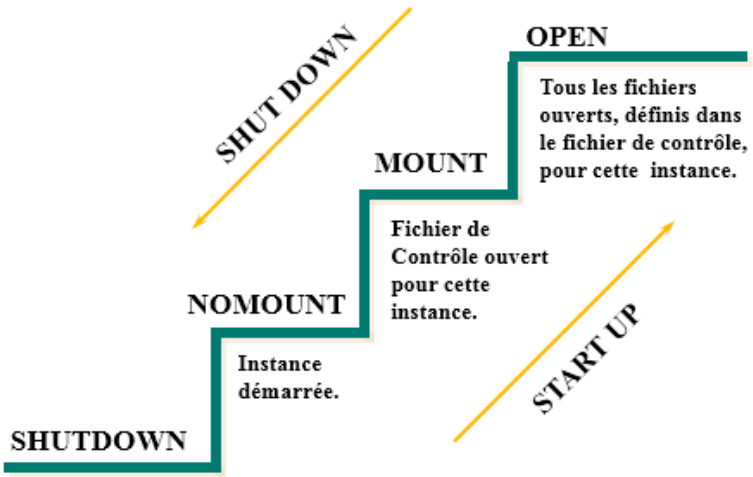
- Le démarrage d'une instance en mode NOMOUNT ne s'effectue qu'à la création de la BDD ou la récréation de fichiers de contrôle
- Le démarrage en mode NOMOUNT comprend les tâches suivantes :
  - Lecture du fichier d'initialisation dans l'ordre de priorité (spfile/pfile)
  - Allocation des zones mémoires
  - démarrage des processus d'arrière-plan
  - L'ouverture du fichier alertSID.log et des fichiers de trace

Mode MOUNT



- Démarrage de l'instance et lecture des fichiers de contrôle
- Localiser les fichiers de données sans les ouvrir (infos dans les fichiers de contrôle)
- Monter la base de données pour effectuer des opérations de maintenance :
  - ❖ Renommer des fichiers de données
  - ❖ Activer ou désactiver des options d'archivage de fichiers de journalisation
  - ❖ Effectuer une récupération complète de la base de données

Mode OPEN



- Ouvrir la base de données en mode de fonctionnement normale
- Les utilisateurs sont autorisés peuvent se connecter à la BDD et effectuer les opérations standard sur la base de données.
- L'ouverture de la BDD comprend les tâches suivantes :
  - ❖ Ouverture les fichiers de données online
  - ❖ Ouverture des fichiers de journalisation en ligne
- Au cours de cette dernière étape, le serveur oracle vérifie que tous les fichiers de données et de journalisation en ligne peuvent être ouverts et contrôle la cohérence de la BDD.
- Si un fichier de données ou de journalisation en ligne manque, le serveur oracle renvoie une erreur.

La commande STARTUP

- Démarrez l'instance et ouvrez la BDD
- Vous pouvez modifier ces priorités si vous indiquez :
  - ❖ Startup pfile= $oracle_home/admin/DB_NAME/pfile/initDB01.ora

    Startup [FORCE] [RESTRICT] [PFILE=name]

    [OPEN [RECOVER] [DATABASE]

    |MOUNT

    NOMONT]

  Force : interrompt l'instance en cours, puis exécute un démarrage normal

  Restrict : n'autorise l'accès à la BDD qu'aux utilisateurs disposant du

  privilège RESTRICTED SESSION

  Recover: lance la procédure de la restauration au démarrage physique la BDD
- Dépannage :
  - ❖ Si vous rencontrez des erreurs à l'exécution de la commande startup. Vous devez d'abort lancer la commande shutdown AVANT DE REEXECUTER LA COMMANDE STARTUP
- Remarque:
  - ❖ Startup et Shutdown sont des commande de **SQL*PLUS** et non **SQL**

Commande ALTER DATABASE

- Remplacez le status NOMOUNT de la BDD par le status MOUNT:

  ALTER DATABASE name **MOUNT**;

- Du status MONT à OPEN

  ALTER DATABASE name **OPEN**;

- Ouvrez la base de données en lecture seule:

  ALTER DATABASE name **READ ONLY**;

- Mode lecture et écriture

  ALTER DATABASE name **READ WRITE**;

Ouvrir une BDD en mode d'accès restreint

- Utilisez la commande STARTUP pour restreindre l'accès à une BDD
  - ❖ STARTUP RESTRICT
- Utilisez la commande ALTER SYSTEM pour placer une instance en mode d'accès restreint :
  - ❖ ALTER SYSTEM ENABLED RESTRICTED SESSION;
- Utilisez la commande ALTER SYSTEM pour désactiverle mode d'accès restreint :
  - ❖ ALTER SYSTEM DISABLE RESTRICTED SESSION;
- Pour mettre fin à une session
  - ❖ ALTER SYSTEM KILL SESSION 'integer1,integer2'
    - Ou integer1= SID de la Vue v$session et integer2 le SERIAL# de la même Vue.
- À l'exécution de ALTER SYSTEM KILL SESSION, le processus PMON effectue les tâches suivantes :
  - ❖ Annulation de la transaction en cours de l'utilisation
  - ❖ Libration de tous les verrous de table ou de ligne
  - ❖ Libération de toutes les ressources réservées par l'utilisateur

Ouvrir une BDD en mode lecture seule

- Ouvrir une BDD en mode lecture seule
  - ❖ STARTUP MOUNT

ALTER DATABASE OPEN READ ONLY

- Une BDD en lecture seule permet :
  - ❖ D'exécuter des interrogations
  - ❖ D'exécuter des tris sur disque à l'aide de tablespaces gérés localement,
  - ❖ De mettre des fichiers de données hors ligne et en ligne, mais pas des tablespaces,
  - ❖ De récupérer des fichiers de données et des tablespaces hors lignes ;

VI.3 Arrêt de la base de données

- Arrêtez la BDD pour :
  - ❖ effectuer la sauvegarde hors ligne de toutes les structures physiques via le système d'exploitation
  - ❖ Appliquer les modifications aux paramètres d'initialisation statique
  - ❖ Mode d'arrêt :
    - ○ ABORT (A)
    - ○ IMMEDIATE (I)
    - ○ TRANSACTIONAL (T)
    - ○ NORMAL (N) (mode par défaut)

Mode d'arrêt	A	I	T	N
Permet de nouvelles connexions	Non	Non	Non	Non
Attend la fin des sessions en cours	Non	Non	Non	Oui
Attend la fin des transactions en cours	Non	Non	Oui	Oui
Applique un point de reprise et ferme les fichiers	Non	Oui	Oui	Oui

Arrêt en mode Normal, Transactional ou immediate

- Phases d'arrêt :
    1. Le cache de tampon de la BDD est écrit dans les fichiers de données,
    2. Les modifications non validées sont annulées,
    3. Les ressources sont libérées(base de données cohérente (base « propre »)
    1. Aucune récupération d'instance

Arrêt en mode Normal

- Arrêt en mode Normal
    - ❖ Aucune nouvelle connexion ne peut être établie
    - ❖ Le serveur oracle attend la déconnexion préalable de tous les utilisateurs
    - ❖ Les tampons de journalisation et de données sont écrits sur disque
    - ❖ Oracle ferme et démonte la base de données
    - ❖ Les processus d'arrière-plan prennent fin et la zone SGA est supprimée de la mémoire
    - ❖ La récupération de l'instance n'est pas nécessaire lors du redémarrage

Arrêt en mode Transactional

- Arrêt en mode transactionnel (évite aux client de perdre leurs travaux en cours)
    - ❖ Aucune nouvelle connexion ne peut être établie
    - ❖ Le client est déconnecté lorsqu'il termine la transaction en cours
    - ❖ La fin de toutes les transactions entraîne l'arrêt immédiat de la BDD
    - ❖ La récupération de l'instance n'est pas nécessaire lors du redémarrage
- Arrêt en mode immediate
    - ❖ Aucune nouvelle connexion ne peut être établie
    - ❖ Les instructions SQL en cours de traitement par oracle ne sont pas terminées
    - ❖ Le serveur oracle n'attend pas la déconnexion des utilisateurs de la BDD
    - ❖ Oracle annule les transactions actives et déconnecte tous les utilisateurs

- ❖ Oracle ferme et démonte la BDD après il arrête l'instance
- ❖ La récupération de l'instance n'est pas nécessaire lors du redémarrage

Arrêt en mode ABORT

- Si les arrêt en modes normale et immediate échouent, vous pouvez abandonner l'instance de la BDD en cours avec le mode ABORT:
    1. Les instructions SQL en cours de traitement par oracle ne sont immédiatement interrompues
    2. Le serveur oracle n'attend pas la déconnexion des utilisateur de la BDD
    3. Les tampons de la BDD et de journalisation ne sont pas écrit dans les fichiers de données,
    4. La base de données n'est pas fermée ni démontée
    5. Une récupération d'instance est nécessaire au démarrage (c'est automatique)
- Déconseiller de sauvegarder une base de données incohérente.

Chapitre 5

Les objets d'une base de données oracle

V. Les Objets d'une Base de données Oracle

V.1 Tables

Les Types de données

Type de données	Description
VARCHAR2 (taille)	Texte de longueur variable. La longueur minimale est de 1 caractère et maximale 4000 caractères. La taille devra obligatoirement être définie.
CHAR (taille)	Texte de longueur fixe. La taille minimum de 1 caractère et pouvant aller jusqu'à 2000 caractères.
NUMBER (p,e)	Nombre ayant une précision pouvant aller de 1 à 38 chiffres et ayant une échelle pouvant aller de –84 à 127. (L'échelle est en fait le nombre de chiffre à afficher après la virgule.)
DATE	Date notée sous la forme DD-MON-YY.
LONG	Texte de longueur variable pouvant stocker jusqu'à 2 gigas. On ne pourra mettre qu'une colonne de type LONG par table.
RAW (taille)	Equivalent à VARCHAR2, mais il permet de stocker des données binaires qui ne sont pas interprétées par Oracle. La taille maximum est de 2000.
LONGRAW	Equivalent à LONG mais pour des données de type binaire non interprétées par Oracle.
CLOB	Permet de stocker un pointeur vers un fichier de données composé de caractère et pouvant contenir jusqu'à 4 gigas.
BLOB	Permet de stocker un pointeur vers un fichier composé de données binaire et pouvant contenir jusqu'à 4 gigas.
BFILE	Permet de stocker les données binaires d'un fichier externe pouvant contenir jusqu'à 4 gigas.

TABLE 3.1 Precision, Scale, and Rounding

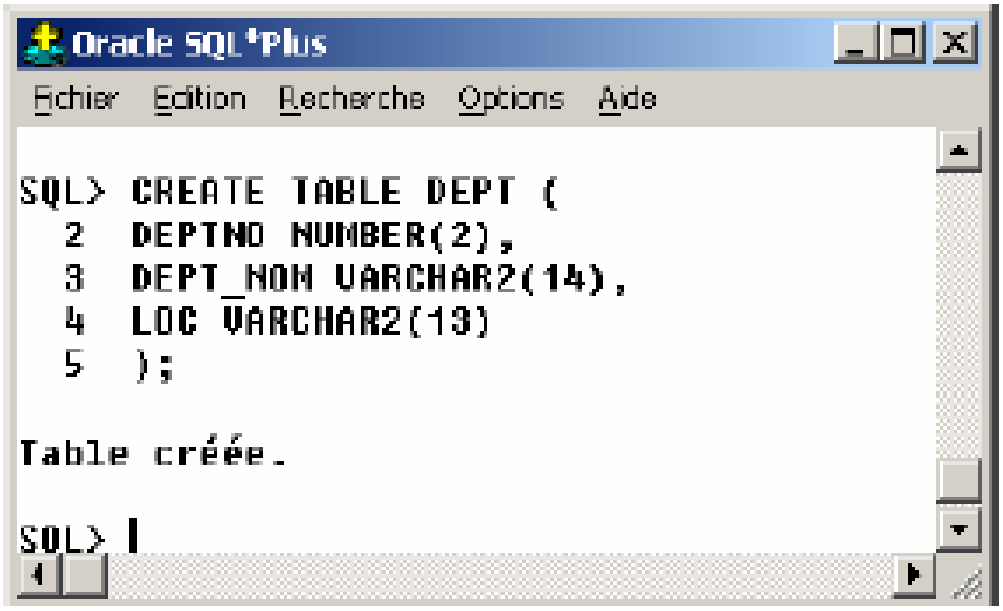
Specification	Actual Value	Stored Value
NUMBER(11,4)	12345.6789	12345.6789
NUMBER(11,2)	12345.6789	12345.68
NUMBER(11,-2)	12345.6789	12300
NUMBER(5,2)	12345.6789	Error – Precision is too small
NUMBER(5,2)	123456	Error – Precision is too small

Tables

- Est une unité de stockage élémentaire, composée de lignes et de colonnes
- Le nom d'une table dans une BDD Oracle
  - ❖ Doit commencer par une lettre
  - ❖ Ne doit pas dépasser les 30 caractères
  - ❖ Ne peut contenir que les caractères A à Z, a à z, 0 à 9, _, $, et #
  - ❖ Ne doit pas porter le nom d'un autre objet appartenant au même utilisateur
  - ❖ Ne doit pas être un mot réservé
- Pour créer une table, vous devez avoir le privilège CREATE TABLE

Syntaxe

```
SQL> CREATE TABLE [Schema.]nom_table
        (colonne1 datatype [Default expr] [Contraintes_Colonne] [,
                  colonne2 datatype [Default expr] [Contraintes_Colonne],
                 … ]
        [Contraintes_Table]
);
```



CREATE TABLE EMP2

(EMPNO NUMBER(2),

ENAME VARCHAR2(50),

LOCATION VARCHAR2(50))

TABLESPACE DATA1;

- Attention
  - ❖ Pour accéder à une table d'un autre utilisateur, il faut précéder le nom de la table par le nom du propriétaire
    - Ex: SCOTT.EMP
  - ❖ La création d'une table fait appel à un ordre du LDD

    ⇒sa validation est automatique
- L'option DEFAULT
  - ❖ Permet de définir la valeur par défaut
    - Ex:
      - DATE_PRET DATE DEFAULT SYSDATE
      - DEPARTEMENT VARCHAR2(25) DEFAULT 'Commercial'

Création des tables en utilisant une sous-requête

- Il est possible de créer une table et insérer des lignes en utilisant la requête **CREATE TABLE et** l'option ***AS sous-requête.***

  CREATE TABLE *table*

  [(colonne, colonne …)]

  AS *sous-requête ;*

- Exemple

  SQL> CREATE TABLE copy_emp AS SELECT * FROM emp;

- Explication : On créée une nouvelle table *copy_emp qui possède la même structure et les mêmes* données que la table *emp.*

- Exemple

  SQL> CREATE TABLE copy_emp

  NOLOGGING

  TABLESPACE DATA2

  AS SELECT * FROM emp

  WHERE EMPNO<20;

- ATTENTION
  - ❖ Le nombre de colonnes spécifiées doit correspondre au nombre des colonnes de la requête
  - ❖ La définition d'une colonne ne peut contenir que son nom et sa valeur par défaut
  - ❖ Si aucune colonne n'est spécifiée, la table aura les mêmes colonnes que celles de la requête
  - ❖ Les contraintes ne sont pas prises en compte

Modification des tables

- L'ordre ALTER TABLE
  - ❖ Permet de modifier la structure d'une table (Ajout d'une colonne, modification, définition d'une valeur par défaut, supprimer une colonne, etc.)
  - ❖ La modification d'une valeur par défaut ne s'appliquer qu'aux insertions ultérieures dans la table

- Apres avoir créé une table, il est parfois nécessaire de modifier sa structure. Pour modifier une table il faudra employer la commande : **ALTER TABLE n*om_de_la_table***

  ALTER TABLE *tablename*

  ADD|MODIFY|DROP *(column datatype [DEFAULT expr]);*

  ALTER TABLE EMPLOYEE

  ADD TELEPHONE VARCHAR2(10);

  ALTER TABLE EMPLOYEE

  ADD (fax VARCHAR2(10)

  email VARCHAR2(50));

  ALTER TABLE EMPLOYEE

  MODIFY TELEPHONE VARCHAR2(20);

  ALTER TABLE EMPLOYEE

  DROP COLUMN TELEPHONE

  CASCADE CONSTRAINTS;

- L'ordre RENAME
  - Permet de modifier le nom d'une table, d'une vue, d'une séquence, et d'un synonyme
- Syntaxe

  SQL> RENAME nom_objet TO nouveau_Nom;

  Ou

  SQL> ALTER TABLE nom_table RENAME

  TO nouveau_Nom;

  - Vous devez être propriétaire de l'objet

Supprimer une table

- L'ordre DROP TABLE
  - ❖ Syntaxe

    SQL> DROP TABLE [schema.]nom_table

    CASCADE CONSTRAINTS;

    - o **La clé étrangère doit être supprimée avant la table parent (cascade constraints)**

- L'ordre DROP TABLE

  SQL> DROP TABLE emp;

- Pour supprimer une table, l'utilisateur Doit être le créateur
- Il possède le privilège DROP ANY TABLE

Vider une table

- L'ordre TRUNCATE TABLE
  - ❖ Permet de vider une table et libérer l'espace de stockage utilisé par la table
- Syntaxe

  SQL> TRUNCATE TABLE nom_table;

- Cet ordre est irréversible
- Vous pouvez utiliser l'ordre DELETE pour supprimer les lignes d'une table
  - ❖ Mais il ne libère pas l'espace de stockage

L'ordre COMMENT ON

- ❖ Permet d'ajouter des commentaires à une table

- Syntaxe

  SQL> COMMENT ON [TABLE nom_table| COLUMN

  nom_table.nom_colonne]IS 'commentaire';

  COMMENT ON TABLE EMP IS

  'Employes';

COMMENT ON COLUMN EMP.TEL IS

'Numero de Telephone';

- ❖ Pour afficher les commentaire, on peut utiliser
  - ALL_COL_COMMENTS
  - USER_COL_COMMENTS
  - ALL_TAB_COMMENTS
  - USER_TAB_COMMENTS

- Intégrité de données
  - ❖ Garantit que les données d'une base respectent certaines règles
    - pour empêcher l'utilisateur d'entrer des données invalides dans la base
  - ❖ Empêchent la suppression d'une table lorsqu'il existe des dépendances
- Toutes les contraintes sont stockées dans le dictionnaire de données (dans la table USER_CONSTRAINTS).
  - ❖ Sont nommées sous Oracle
    - Automatiquement sous le format SYS_Cn
    - Manuellement par l'utilisateur
      - Lors de la création (CREATE TABLE)
      - Après la création (ALTER TABLE)
  - ❖ Donner des noms explicites à vos contraintes pour faciliter le référencement.
- Les contraintes peuvent être définies soit au moment de la création de la table soit après la création.
- Types de contraintes valides

NOT NULL	**Spécifie que cette colonne ne doit pas contenir** une valeur **NULL**
UNIQUE	**Spécifie une colonne (ou une combinaison de colonnes) dont les valeurs doivent être uniques**
PRIMARY KEY	**Identifie la clé primaire**
FOREIGN KEY	**Identifie la clé étrangère**
CHECK	**Spécifie une condition à laquelle doit répondre chaque ligne de la table**

- **La contrainte NOT NULL**

  CREATE TABLE employee (

  employee_id NUMBER(6),

  nom VARCHAR2(25) NOT NULL, ➔NOMME PAR LE SYSTEME

  sal NUMBER(8,2),

  hiredate DATE CONSTRAINT emp_hiredate_nn NOT NULL,

  ➔NOMME PAR L'utilisateur …) ;

Les Contraintes

- PRIMARY KEY
  - ❖ Définit :

    Au niveau table

```
SQL> CREATE TABLE EMP (
          empno NUMBER(4),
          ename VARCHAR2(10),
          …
          deptno NUMBER(7,2) NOT NULL,
          CONSTRAINT emp_emmno_pk PRIMARY KEY (EMPNO)
);
```

    Au niveau colonne

```
SQL> CREATE TABLE EMP (
          empno NUMBER(4) PRIMARY KEY,
          ename VARCHAR2(10),
          …
          deptno NUMBER(7,2) NOT NULL,
          );
```

- FREIGN KEY
  - ❖ Définit au niveau table ou colonne
  - ❖ Associé à
    - REFERENCES pour identifier la table et la colonne de la table maître
    - ON DELETE CASCADE pour autoriser la suppression d'une ligne dans la table maître et des lignes dépendante dans la table détail
  - ❖ Exemple :

    Au niveau Table

```
SQL> CREATE TABLE EMP (
          empno NUMBER(4) PRIMARY KEY,
          ename VARCHAR2(10),
          …
          deptno NUMBER(7,2) NOT NULL,
          CONSTRAINT emp_deptno_KT FOREIGN KEY (deptno) REFERENCES DEPT (deptno));
```

Au niveau Colonne

```
SQL> CREATE TABLE EMP (
          empno NUMBER(4) PRIMARY KEY,
          ename VARCHAR2(10),
          …
          deptno NUMBER(7,2) NOT NULL REFERENCES DEPT (deptno));
```

- CHECK

```
SQL> CREATE TABLE EMP (
          empno NUMBER(4) PRIMARY KEY,
          ename VARCHAR2(10),
          …
          deptno NUMBER(7,2) NOT NULL,
          CONSTRAINT emp_deptno_KT CHECK (deptno BETWEEN 10 AND 99), …));
```

- UNIQUE
  Définit au niveau table et colonne
  Au niveau Table

```
SQL> CREATE TABLE EMP (
          empno NUMBER(4) PRIMARY KEY,
          ename VARCHAR2(10),
          …
          deptno NUMBER(7,2) NOT NULL,
          CONSTRAINT emp_nom_KT UNIQUE (ename)
);
```

Au niveau Colonne

```
SQL> CREATE TABLE EMP (
          empno NUMBER(4) PRIMARY KEY,
          ename VARCHAR2(10) UNIQUE,
          …
          deptno NUMBER(7,2) NOT NULL,
);
```

- Ajout d'une contrainte
  ALTER TABLE table
  ADD [CONSTRAINT constriant] type (column);

- Pour ajouter la contrainte NOT NULL utilisez la clause MODIFY
  ALTER TABLE table
  MODIFY column NOT NULL;

- Suppression d'une contrainte
  ALTER TABLE EMP
  DROP CONSTRANT cn_nom
- Désactivation/Activation d'une contrainte

```
SQL> ALTER TABLE nom_table
[DISABLE | ENABLE] CONSTRAINT nom_contrainte [CASCADE]
);
```

✓ On peut utilizer les clauses ENABLE et DISABLE dans CREATE TABLE et ALTER TABLE

- Désactivation de contraintes

  ALTER TABLE Emp

  DISABLE CONSTRAINT pk_emp CASCADE;

  ALTER TABLE Emp

  DISABLE CONSTRAINT cn_nom;

- Activation de contraintes

  ALTER TABLE Emp

  ENABLE CONSTRAINT pk_emp;

- Comment afficher les contraintes sur une table

```
SQL> SELECT constraint_name, constraint_type, search_condition
          FROM user_constraints
          WHERE table_name = 'EMP'
);
```

Constraint_Name	Constraint-type	Search_Condition
SYS_C003042	**C** (Signifie CHECK)	EMPNO is not Null
SYS_C003044	**U** (Signifie Unique)	
TEST_TT	**P** (Signifie Primary Key)	

- Comment afficher les colonnes associées aux contraintes dans une table

```
SQL> SELECT constraint_name, column_name
          FROM user_cons_column
          WHERE table_name = 'EMP'
);
```

Constraint_Name	Column Name
TEST_TT	EMPNO
SYS_C003042	DEPTNO

Données

L'ordre INSERT

- Permet d'ajouter de nouveaux enregistrements dans une table
- Syntaxe

```
SQL> INSERT INTO nom_table [(colonne_1 [, colonne_2…)]
          VALUES (Valeur_colonne_1 [, valeur_colonne_2…]
);
```

- Placer les valeurs de types caractère et date entre simple quotes
- Exemple
  - ❖ En précisant les colonnes concernées

```
SQL> INSERT INTO Emp (empno, ename, sal)
          VALUES (5555, 'BARTH', 2500);
```

- Sans précision

```
SQL> INSERT INTO Emp
        VALUES (5555, 'BARTH', NULL, NULL, SYSDATE, 2500, NULL, NULL);
```

- Avec des variables saisies par l'utilisateur

```
SQL> INSERT INTO Emp (empno, ename, sal)
        VALUES (&Num_Salarie, '&Nom_salarie', &salaire);
```

- A partir d'une autre table

```
SQL> INSERT INTO Emp (empno, ename, sal)
            SELECT empno, ename, sal FROM SCOTT.EMP WHERE job = 'manager';
```

L'ordre UPDATE

- Permet de modifier les enregistrements Existants
  - A partir des valeurs

```
SQL> UPDATE Nom_table
        SET Colonne_1 = valeur_1 [, Colonne_2 = …]
        [WHERE condition] ;
```

  - A partir d'une requête

```
SQL> UPDATE Nom_table
        SET (Colonne_1, Colonne_2, ) =
        (SELECT colonne_11, Colonne_22, …
         FROM nom_table_2
         WHERE condition)
     [WHERE condition] ;
```

- Exemple
  - Modifier le poste et le n° de département de l'employé 7698 à l'identique de l'employé 7499

```
SQL> UPDATE Emp
        SET (job, deptno) =
        (SELECT job, deptno
         FROM Emp
         WHERE empno = 7499)
     WHERE empno = 7698 ;
```

L'ordre DELETE

- Permet de supprimer les enregistrements d'une table

```
SQL> DELETE [FROM] nom_table
     [WHERE condition] ;
```

✓ Attention, si vous omettez la cause WHERE, toutes les lignes sont supprimées

V.2 Vues

- Les vues sont des tables virtuelles qui permettent de définir des filtres sur des tables réelles.
  - ❖ Une Vue est requête pouvant être manipulée comme une table
  - ❖ Contrairement à une table, une vue stocke aucune donnée, seulement une instruction SQL
- Elles permettent de limiter l'accès à certaines colonnes d'une table pour un groupe d'utilisateurs et le type d'accès
- Elles permettent de simplifier l'accès aux données

❖ Syntaxe

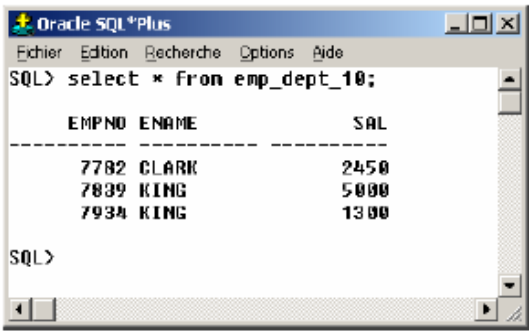
```
SQL> CREATE [OR REPLACE] [FORCE|NOFORCE] VIEW Nom_Vue
         [(alias [, alias]…)]
AS requête
[WITH READ ONLY]
```

REPLACE	Recrée la vue
FORCE	Crée la vue même si les tables n'existent pas
WITH READ ONLY	Garantit qu'aucune opération LMD ne peut être exécutée dans la vue

Exemple

- Sans alias

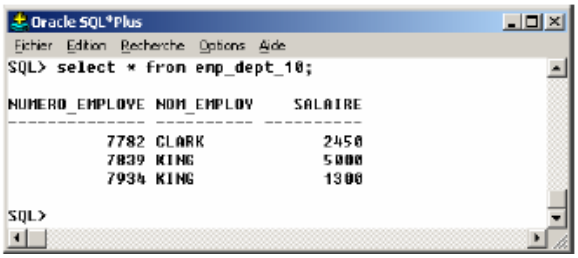
```
SQL> CREATE OR REPLACE VIEW
EMP_DEPT_10
        AS SELECT empno, ename, sal
        FROM Emp
        WHERE deptno = 10;
```



- Avec des Alias

```
SQL> CREATE VIEW EMP_DEPT_10
        AS SELECT empno Numero_Employe, ename Nom_Employe, sal Salaire
        FROM Emp
        WHERE deptno = 10;
```

```
SQL> CREATE VIEW EMP_DEPT_10 (Numero_Employe, Nom_Employe, Salaire)  AS
SELECT empno, ename, sal
        FROM Emp
        WHERE deptno = 10;
```



Règles d'exécution des ordres LMD

Vous ne pouvez pas :

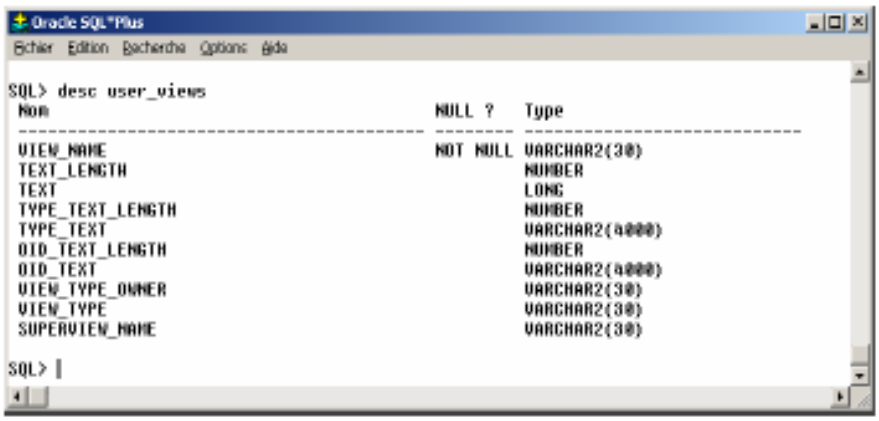
- Supprimer un enregistrement si la vue contient
  - Des fonctions de groupe
  - Une clause GROUP BY
  - Le mot clé DISTINCT
- Modifier les données si la vue contient
  - Des fonctions de groupe
  - Une clause GROUP BY
  - Le mot clé DISTINCT

- Des colonnes définies par des expressions

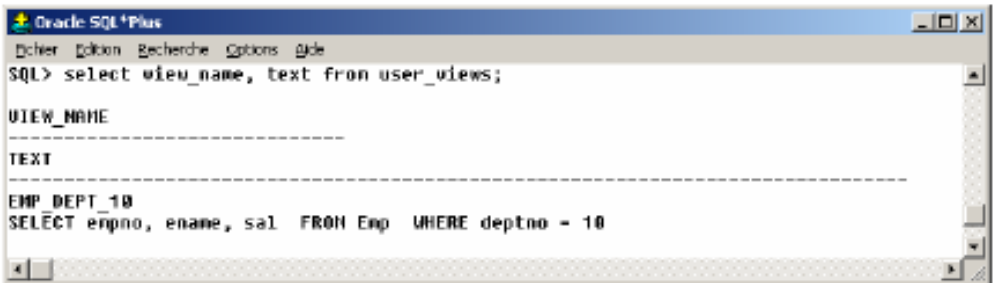
❖ Ajouter de données si la vue contient
- Des fonctions de groupe
- Une clause GROUP BY
- Le mot clé DISTINCT
- Des colonnes définies par des expressions
- Les tables de base contiennent des colonnes NOT NULL non sélectionnées par la vue
    - Sans valeur par default

La table USER_VIEWS

- Permet d'afficher le nom et la définition des vues de l'utilisateur



- L'ordre SELECT est stocké dans une colonne de type LONG



L'ordre DROP VIEW

- Permet de supprimer une vue
- N'entraine pas la perte des données
- Exemple

```
SQL> DROP VIEW EMP_DEPT_10;
Vue supprimée.
```

V.3 Séquences

- Une séquence est un objet créé par l'utilisateur.
- Elle sert à créer des valeurs pour les clés primaires, qui sont incrémentées ou décrémentées par le serveur Oracle.
- Noter que la séquence est stockée et générée indépendamment de la table, et une séquence peut être utilisée pour plusieurs tables.

Syntaxe

[CREATE | ALTER] SEQUENCE ***[schéma].Nom_séquence***

[INCREMENT BY **entier**] –> définit l'intervalle d'incrémentation ou de –

[START WITH **entier**] –> Première valeur de la séquence

[MAXVALUE **entier**] –>Valeur maximale (par défaut 1027 ou -1)

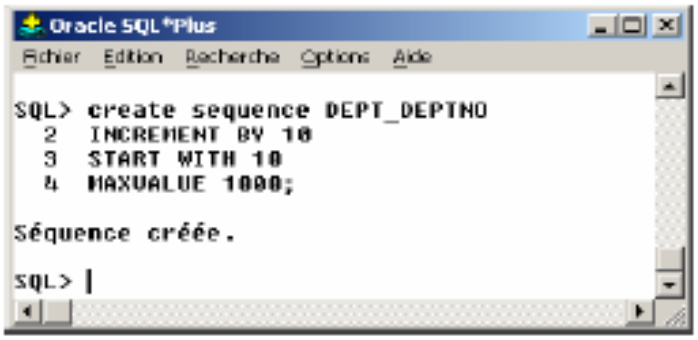
[MINVALUE **entier**] –> Valeur minimale (par défaut 1 ou -1026)

[CYCLE | NOCYCLE] –> Revenir à la valeur initiale

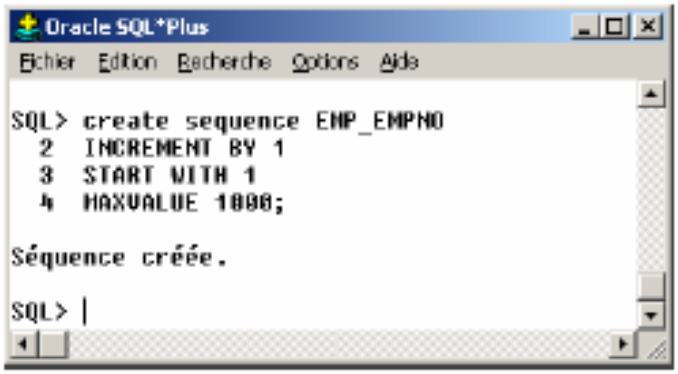
[CACHE **entier** | NOCACHE] –> allocation de séquences en mémoire (par défaut 20)

Exemple

Créer un compteur DEPT_DENO pour la clé primaire de la table DEPT

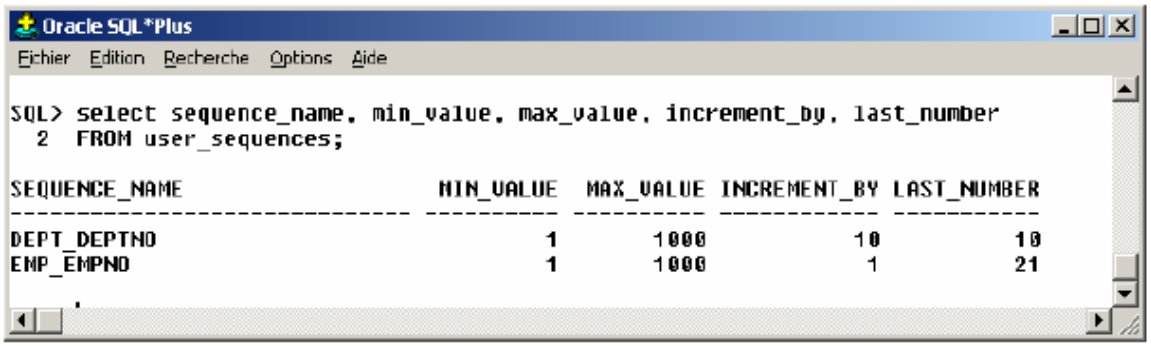


Créer un compteur pour la clé primaire de la table EMP



LA table USER_QEQUENCES

- Contient les valeurs des séquences définies par l'utilisateur



- NEXTVAL
  - ❖ Donne la prochaine valeur de la séquence
  - ❖ Retourne une valeur unique même si elle est utilisée par plusieurs utilisateurs
- CURRVAL
  - ❖ Renvoie la valeur courante de la séquence utilisée pour chacun des utilisateurs

NEXTVAL

```
SQL> INSERT INTO emp (empno, ename, sal)
        VALUES (emp_empno.NEXTVAL, 'rich', 2000);
```

CURRVAL

```
SQL> SELECT emp_empno.CURRVAL FROM dual;
```

- Pour utiliser la séquence d'un autre utilisateur, il faut préciser le nom du schéma (schéma.nom_séquence)

```
SQL> SELECT scott.emp_empno.CURRVAL FROM dual;
```

- Il faut avoir
  - ❖ Les privilèges objet SELECT
  - ❖ Les privilèges Système SELECT ANY SEQUENCE

Règles de modification

- Etre propriétaire ou voir le privilège Objet ALTER
- ALTER SEQUENCE ne modifie que les numéros de séquence à venir
- Vous devez supprimer la séquence puis la recréer si vous voulez modifier le premier numéro (START WITH)

L'ordre DROP SEQUENCE

- Permet de supprimer une séquence
- Syntaxe

```
SQL> DROP SEQUENCE Nom_séquence;
```

V.4 Synonymes

- Un synonyme est un nom alternatif pour désigner un objet de la base de données.
  - ❖ C'est aussi un objet de la base de données.
- Si un utilisateur veut accéder à un objet appartenant à un autre schéma, il faut le faire préfixer par le nom de schéma (SCOTT.emp)

Définition

- Permet de renommer un objet dans le schéma (Table, Vue, séquence, etc.)
- Elimine la contrainte de désigner l'objet avec son schéma

Syntaxe

```
SQL> CREATE [PUBLIC] SYNONYM nom_synonyme
       FOR objet;
```

Public donne l'accès à tous les utilisateurs

Exemple

```
SQL> CREATE PUBLIC SYNONYM ma_sequence
       FOR EMP_EMPNO;
```

Suppression

```
SQL> DROP SYNONYM nom_synonyme;
```

V.5 Index

- Un index est un objet qui peut augmenter la vitesse de récupération des lignes en utilisant les pointeurs.
- Les index peuvent être créés automatiquement par le serveur Oracle ou manuellement par l'utilisateur.
- Ils sont indépendants donc lorsque vous supprimez ou modifiez un index les tables ne sont pas affectées.
- Lors de la création des clés primaires ou des clés étrangères, les index sont créés automatiquement et ils possèdent les mêmes noms que les contraintes.
- **Quand créer un index :**
  - La colonne contient une large plage de valeurs.
  - La colonne contient plusieurs valeurs nulles.
  - Une ou plusieurs colonnes sont fréquemment utilisées dans la clause **WHERE** ou pour les conditions de jointure.
  - La table est grande et la plupart des requêtes recherchent moins de 2-4% des lignes.
- **Quand ne pas créer des index:**
  - La table est petite.
  - Les colonnes ne sont pas souvent utilisées.
  - Les requêtes recherchent plus de 2-4% des lignes.
  - La table est mise à jour fréquemment.
  - Les colonnes indexées sont référencées comme une partie de l'expression.
- Pour créer un index, il faut utiliser la requête **CREATE INDEX**

  CREATE INDEX ***nomindex***

  ON ***table (column1, column2 …)***

[TABLESPACE INDX];

- Exemple :

  SQL> CREATE INDEX emp_ename_idx

  ON emp(ename);

- Pour vérifier l'existence d'un index, vous pouvez interroger la table USER_INDEXES
- **Supprimer un index**
  - ❖ Il n'est pas possible de modifier un index, il faut le supprimer ou recréer.

    DROP INDEX *nomindex;*

    DROP INDEX emp_ename_idx;

Références Bibliographiques

Ian ABRAMSON, Michael ABBEY, Michael COREY, Oracle 10g: Notions Fondamentales, Oracle Press.

Olivier Heurtel, Oracle 10g Installation du serveur sous Windows, Linux, Oracle, Eni Editions.