Administration de Base de Données

ORACLE

Pr. Abderrahim EL QADI
Département Mathématique appliquée et Génie Informatique
ENSAM-Université Mohammed V de Rabat

INDIA 2

Année Universitaire : 2021/2022

Plan

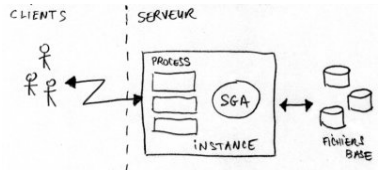
Références

- Oracle 10g Administration, Olivier Heurtel, 2005, 489p.
- Les bases de données Oracle8i, Développement, Administration et Optimisation, Roger Chapuis, Dunod, 2001, 382pp.
- Oracle10g sous Windows, Gilles Briard, Eyrolles, 895pp.
- Oracle11g Administration, Razvan Bizoï, Eyrolles, 2011, ISBN : 978-2-212-12898-7.

1. Introduction : SGBD Oracle

Une base de données Oracle est un objet riche et complexe composé physiquement :

- D'une INSTANCE (zone mémoire partagée + processus de fond)
- D'un certain nombre de FICHIERS PHYSIQUES, totalement indépendants du nombre de table.



– Instance Oracle :

- A chaque démarrage d'une base, une mémoire SGA (System Global Area) est allouée et un ensemble de BACKGROUND PROCESS est démarré.
- La combinaison SGA+BACKGROUND PROCESS est appelée : **« instance ».**
- Le nom de l'instance est dépendant d'une variable d'environnement ORA_SID (ou ORACLE_SID).
- Pour changer d'instance, il suffira de changer la valeur de cette variable.

```
C:> set oracle_sid=info
```

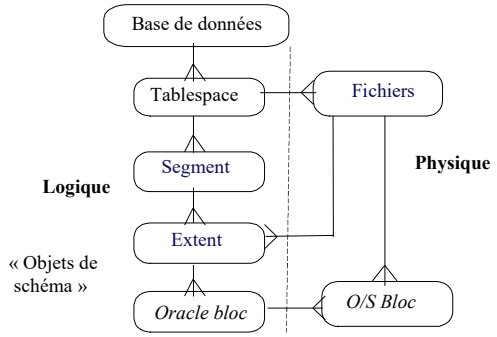

– Session utilisateur : est une connexion à la base de données par l'intermédiaire d'un PROCESS USER

Exemple : Quand un utilisateur se connecte à travers SQL*Plus, cet utilisateur doit fournir son nom et son mot de passe. Une session est alors ouverte.

```
SQL> Entrer le nom d'utilisateur : scott
SQL> Entrer le mot de passe: tiger
pour changer l'utilisateur:
SQL> connect to system/manager
```

2. Architecture d'une BD Oracle

2.1. Architecture de base



2.1.1. Niveau Physique

- Ce niveau constitue d'une série de fichiers qui contiennent l'ensemble des données de la base :

2.1.1.1 Les fichiers de données (data files) :

- Ils contiennent toutes les données de la base :

Exemple : données utilisateurs : informations clients (nom, prénom, adresse…)

Données systèmes : taille d'une table, nom et mot passe d'utilisateur…etc.

- Toutes les structures logiques et physiques y sont stockées : tables, index, …

2.1.1.2 Les fichiers de journalisation (redo log files) :

- Ces fichiers contiennent la trace de l'activité en terme de mise à jour sur la base
- deux groupes au moins avec chacun au moins un fichier sont obligatoires
- ils doivent être multiplexés pour plus de sécurité
- ne sont utiles qu'en cas de perte des fichiers de données ou d'arrêt anormal de la base
- Fonctionnent de façon cyclique

2.1.1.3 Les fichiers de contrôle (control files) :

- Contiennent la description physique de la base :
  - le nom de la base,
  - le nom et le chemin d'accès de chacun des fichiers,
  - la date et l'heure de création de la base,
- Ils sont obligatoires au moment du démarrage et de l'ouverture de la base.
- Il est recommandé d'en avoir plusieurs et de les localiser physiquement sur des disques différents.

2.1.1.4 Fichier INIT.ORA

- est le fichier de paramétrage de la base de données.
- Contient des paramètres de fonctionnement d'une instance et un paramètre identifiant le ou les fichiers **CONTROL**
- Sans ce fichier la base ne pourra pas démarrer.
- L'emplacement par défaut est ORACLE_HOME\dbs.

Quelques Paramètres du fichier INIT.ORA

- **CONTROL_FILES** : Noms des fichiers de contrôle.
- **DB_NAME** : Identifiant de la base de données de 5 caractères ou moins. (seul paramètre nécessaire à la création d'une base).

2.1.2. Niveau logique

- Ce niveau se compose des objets suivants :

Tablespaces ; segments ; extensions (extents) ; blocs ; objet de schéma (schema object).

Les objets de schéma comprennent : Les tables ; les vues ; les index ; les séquences ; les procédures stockées ; les fonctions ; les packages ; les déclencheurs (triggers).

2.2. Architectures du Serveur Oracle

- 3 types d'architecture globale de systèmes d'information basés sur Oracle :
  - Architecture locale : Tout est sur le même serveur matériel, programme client et serveur de données Oracle.
  - Architecture client/serveur : On a un programme client (un exécutable) sur le poste client, dit alors 'client lourd'. Le PC communique avec le serveur de données sur un serveur distant, via le réseau, et la couche Oracle Net.

  - Architecture 3 tiers : Pas de programme client. Un navigateur suffit sur le poste de travail (dit alors 'client léger'). Il dialogue avec le serveur de données distant via http.

2.2.1. Configuration Oracle Net

- Les fichiers de configuration coté client et/ou serveur se trouvent dans le répertoire ORACLE_HOME/network/admin
- Le programme client doit notamment pouvoir identifier le serveur de données Oracle sur le réseau. Il utilise pour ce faire une résolution de nom de serveur.

2.2.1.1 Les méthodes de résolution de nom

- 4 modes de nommage :
  - local : on utilise un fichier de configuration situé sur le poste client : TNSNAMES.ORA
  - via un annuaire : on utilise un serveur d'annuaire compatible LDAP : Oracle Internet Directory par exemple…
  - basé uniquement sur TCP/IP (exceptionnellement pas de couche spécifique Oracle Net nécessaire coté client) : On utilise une chaîne de connexion explicite, qui devra préciser le serveur (machine) cible, et éventuellement le port d'écoute du logiciel serveur Oracle Net, et un nom de service de données.

Exemple:

```
CONNECT user1/pwd@serveur1:1521/ma_base
```

le port 1521 est le port par défaut du listener Oracle Net.

  - externe : on utilise un service de nom externe, non Oracle mais compatible avec ce dernier (NIS, DCE)

2.2.2. Fichier de configuration client : TNSNAMES.ORA

Exemple d'une entrée 'ORCL_NET' qui pointe sur la BD 'ORCL' du serveur 'LocalHost'

2.3. Fichier de configuration serveur : LISTENER.ORA

3. Dictionnaire de données

– C'est un ensemble de tables et de vues contenant des informations sur la BD. Il fournit des informations sur :

- Structure physique et logique de la base de données
- Noms, descriptions, implantation des objets
- Contraintes d'intégrité
- Utilisateurs et privilèges.

– On peut accéder au dictionnaire que par l'intermédiaire de vues.

- Il existe 4 catégories de vues (reconnaissables par leur préfixe) :
  - **Vues_dynamiques : v$_nomvue :** donnent les informations systèmes de la base.

  Exemple : v$instance : donne le nom d'instance.

  - **USER_XXX** : décrit les objets appartenant à l'utilisateur connecté
  - **ALL_XXX** : décrit les objets accessibles à l'utilisateur connecté
  - **DBA_XXX** : décrit tous les objets (vues autorisées aux DBAs seulement...)

Chaque XXX est en général remplacé par un nom (en anglais) significatif.

- Un (méta) description du dictionnaire est donnée par la vue DICT.

Select * from DICT where table_name= 'DBA_USERS';

Exemples de vues:

- Tables qui contiennent un attribut Name

```
SELECT TABLE_NAME FROM USER_TAB_COLUMNS
WHERE COLUMN_NAME='NAME';
```

- Attributs de la table DEPT

```
SELECT COLUMN_NAME FROM USER_TAB_COLUMNS
WHERE TABLE_NAME='DEPT';
```


4. Gestion de la sécurité et des ressources

4.1. Contrôle des accès

L'accès à la base de données s'effectue par l'intermédiaire de la notion utilisateur.

4.1.1. Utilisateur

Chaque utilisateur, ou USER est définit par :

- Un nom d'utilisateur + mot de passe ;
- Liste de tablespaces ;
- Un ensemble de privilèges et rôles ;
- Un profil.

4.1.1.1 Création d'un Utilisateur

La création d'un utilisateur s'effectue par l'ordre Create user :

Exemple:

```
CREATE USER user1
IDENTIFIED BY info
DEFAULT TABLESPACE USERS
QUOTA 15m ON USERS
PASSWORD  EXPIRE
PROFILE default;
```


- Default tablespace : tablespace qui sera utilisé par défaut lors de la création d'une table.
- Quota [nombre/unlimited] on tablespace : limite l'espace qui pourra être alloué pour les tables de l'utilisateur.
- Profile nom_profile : affecter un profil particulier à l'utilisateur
- Password Expire : demande le changement de passe.

La vue **DBA_USERS** permet de consulter la liste des caractéristiques des utilisateurs.

4.1.1.2 Modification du mot de passe

```
ALTER USER user1
IDENTIFIED BY deptinfo
PASSWORD EXPIRE;
```

4.1.1.3 Modification du quota

```
ALTER USER user1
QUOTA 10M ON USERS;
```

4.1.1.4 Suppression d'un Utilisateur

```
DROP USER user1;
```

Suppression des Objets associés : Clause CASCADE :

```
DROP USER user1 CASCADE;
```


4.1.2. Privilège

Deux types de privilèges :

- les privilèges système autorisent l'accès à la BD.
- les privilèges objets sur les données (tables, vues, procédures, fonctions, séquences) qui donnent les autorisations d'accès aux données.

- La commande **GRANT** permet d'attribuer un privilège à un utilisateur ou à un groupe d'utilisateurs.
- La commande **REVOKE** permet de supprimer les privilèges.

4.1.2.1 Privilège système

Exemple : Privilèges SYSTEM

CREATE SESSION : connexion à la BD
CREATE TABLE, ALTER TABLE, DROP TABLE

a- Attribution d'un privilège SYSTEM :

```
GRANT privilège TO [utilisateur | rôle | PUBLIC][WITH ADMIN OPTION]
```

- WITH ADMIN OPTION : autorise celui qui a reçu le privilège ou le rôle à le transmettre à un autre utilisateur ou rôle.
- Public permet d'affecter le privilège ou le rôle à tous les utilisateurs.

Exemples:

```
GRANT CREATE SESSION, CREATE TABLE TO user1;
GRANT CREATE SESSION TO scott WITH ADMIN OPTION;
```

b- Suppression d'un privilège SYSTEM

```
REVOKE privilège FROM utilisateur| rôle | PUBLIC;
```

Exemples:

```
REVOKE CREATE TABLE FROM user1;
REVOKE CREATE SESSION FROM scott;
```


c- Les vues du dictionnaire

```
SELECT * FROM DBA_SYS_PRIVS ORDER BY grantee, privilege ;
```

GRANTEE	PRIVILEGE	ADM
CONNECT	ALTER SESSION	NO
CONNECT	CREATE CLUSTER	NO

4.1.2.2 Privilège objet

Un utilisateur qui crée un objet à tous les droits sur celui-ci, les autres utilisateurs (sauf DBA) n'ont aucun droit.

a- Création d'un privilège OBJET

```
GRANT [privilège | ALL] ON objet TO [utilisateur | rôle | PUBLIC]
[WITH GRANT OPTION];
```

Exemple :

```
GRANT UPDATE(ename, sal) ON emp TO user1
WITH GRANT OPTION;
```

b- Suppression d'un privilège

```
REVOKE [privilège | ALL] ON objet FROM [utilisateur | rôle | PUBLIC] ;
```


c- Visualisation des privilèges objets

DBA_TAB_PRIVS, DBA_COL_PRIVS, ALL_TAB_PRIVS, ...

- Principales Colonnes de vues ci-dessus

GRANTEE : utilisateur ayant reçu le privilège

OWNER : propriétaire de la table

TABLE_NAME : nom de la table

COLUMN_NAME : Nom de la colonne concerné

GRANTOR : Utilisateur ayant affecté le privilège

PRIVILEGE : privilège affecté

GRANT : privilège reçu

4.1.3. Rôle

C'est un groupe nommé de privilèges qui peut être accordé à un utilisateur.

4.1.3.1 Création

```
CREATE ROLE nom_de_rôle [NOT IDENTIFIED]
IDENTIFIED BY[mot_de_passe]
```

Exemples :

```
CREATE ROLE role1;

GRANT CREATE TABLE, CREATE SESSION TO role1;
GRANT SELECT ON emp TO role1;
```


4.1.3.2 Modification

```
ALTER ROLE nom_rôle [NOT IDENTIFIED]
IDENTIFIED BY [mot_de_passe]
```

4.1.3.3 Activation et désactivation des Rôles

- Désactiver un rôle enlève les privilèges associés aux utilisateurs.
- Activer un rôle affecte les privilèges associés aux utilisateurs.
- La commande SET ROLE permet d'activer et de désactiver les rôles
- Les rôles par défaut sont affectés à l'utilisateur à sa connexion.
- Un mot de passe peut être nécessaire


```
SET ROLE {rôle [IDENTIFIED BY mot_de_passe] | ALL |
EXCEPT rôle [,rôle] …};
```

Exemples :

```
SET ROLE  role1;
SET ROLE ALL EXCEPT role1;
```

4.1.3.4 Suppression d'un rôle pour un utilisateur

```
REVOKE rôle [,rôle] … FROM {utilisateur | rôle | PUBLIC};
```

Exemple:

```
REVOKE  role1 FROM PUBLIC;
```


4.1.3.5 Suppression d'un rôle

Exemple :

```
DROP ROLE role1;
```

4.1.3.6 Informations sur les rôles

• Les vues suivantes contiennent des informations sur les rôles : dba_roles, user_role_privs, dba_role_privs, role_role_privs, role_sys_privs, role_tab_privs, session_roles

Exemple 1 : liste de tous les rôles de la base

```
sql> SELECT * FROM sys.dba_roles ;
```

4.2. Profil

- C'est un objet géré par la BD qui définit l'ensemble de ressources mises à la disposition des utilisateurs par le SGBDR, et les paramètres de gestion des mots de passe.

4.2.1. Création

– **Syntaxe partie limite des ressources :**

```
CREATE PROFILE profile LIMIT
 [ SESSIONS_PER_USER { integer | UNLIMITED | DEFAULT} ]
  [ CONNECT_TIME { integer | UNLIMITED | DEFAULT } ]
```


- **Mots clés et paramètres :**

- **Session_per_user** : Nombre maximum de sessions par utilisateur
- **connect_time** : temps écoulé maximum (en minutes)
- **unlimited** : limite de la ressource illimitée
- **default** : prend la limite par défaut de la ressource

Exemple :

```
CREATE  PROFILE  etud  LIMIT
SESSIONS_PER_USER  1
CONNECT_TIME  480;
```


– **Syntaxe partie password :**

```
CREATE PROFILE profile LIMIT
 [FAILED_LOGIN_ATTEMPTS {expr | UNLIMITED | DEFAULT}]
 [PASSWORD_LIFE_TIME {expr | UNLIMITED | DEFAULT}]
  [PASSWORD_LOCK_TIME {expr | UNLIMITED | DEFAULT}]
  [PASSWORD_VERIFY_FUNCTION {function, NULL,
DEFAULT}]
```

– **Mots clés et paramètres :**

- **Failed_login_attempts**: nombre d'échecs avant le blocage du compte
- **password_life_time**: durée en jours avant l'expiration du mot de passe
- **password_lock_time**: durée en jours du verrouillage d'un compte
- **password_verify_function**: fonction de contrôle des mots de passes.

4.2.2. Attribution Profile à un utilisateur

```
CREATE USER user1
  IDENTIFIED BY info
  PROFILE etud;
```

```
Ou:    ALTER USER user1
       PROFILE etud;
```

La vue **DBA_PROFILE** permet de consulter la liste des caractéristiques des utilisateurs.

4.2.3. Modification

```
ALTER profile nom_prifile
  …Liste des instructions de création de profile
```

Exemple :

```
ALTER PROFILE default LIMIT
CPU_PER_SESSION 600
```

4.2.4. Suppression

```
DROP profile nom_profile  CASCADE;
```


5. Gestion de l'Espace physique d'une base

5.1. Tablespace

- Unité logique de stockage des données,
- Lié à un ou plusieurs fichiers,
- un fichier est associé à un et un seul Tablespace,
- un Tablespace peut être mis **OFFLINE**, sauf le Tablespace **SYSTEM** (crée en même temps que la BD).
- Un Tablespace regroupe des objets de schéma (tables, index, etc).

Exemples d'utilisation

- Limitation d'utilisation des espaces pour les utilisateurs (quota),
- disponibilité des informations,
- sauvegarde et restauration des données,
- distribution des informations sur différents disques.

5.1.1. Création

```
CREATE  TABLESPACE  nom_tablespace
DATAFILE  fichier [, fichier] …
DEFAULT  STORAGE ([INITIAL valeur]
      [NEXT valeur]
      [MINEXTENTS  nombre]
      [MAXEXTENTS  nombre]
      [PCTINCREASE nombre]) ;
```


Les paramètres suivants influent sur l'allocation de l'espace physique:

• INITIAL : taille de la 1ère extension du segment

• NEXT : taille 2ème extension du segment

• MINEXTENTS : nombre d'extensions allouées à la création du segment (par défaut : 1)

• MAXEXTENTS: nombre maximal d'extensions pouvant être allouées au segment (par défaut : dépend de la taille du bloc, min : 1)

• PCTINCREASE : pourcentage d'accroissement de la taille des extensions à partir de la 3ème (taille extension i+1=(taille extension i) × (1+p/100)) (par défaut : 50%) .

Exemple:

```
CREATE TABLESPACE app_data
DATAFILE 'app01.dbf' SIZE 1M,
         'app02.dbf' SIZE 1M
DEFAULT STORAGE (INITIAL 500K NEXT 500K MINEXTENTS 1
MAXEXTENTS 500
PCTINCREASE 0);
```


5.1.2. Ajout de fichier à un Tablespace

Exemple

```
ALTER TABLESPACE app_data
ADD DATAFILE 'app03.dbf' SIZE 2M;
```

5.1.3. Extension Automatique

Exemple :

```
ALTER TABLESPACE app_data
ADD DATAFILE 'app04.dbf' SIZE 2M
AUTOEXTEND ON NEXT 1M MAXSIZE 50M;
```


5.1.4. Modification de la taille d'un fichier

Exemple :

```
ALTER DATABASE DATAFILE 'app02.dbf' RESIZE 2M;
```

5.1.5. Modification des paramètres de Stockage

Exemple :

```
ALTER TABLESPACE app_data
DEFAULT STORAGE
(INITIAL 2M NEXT 2M MAXEXTENTS 1000);
```

5.1.6. Statut d'un Tablespace

• En mode **OFF LINE**, les données ne sont pas accessibles

• En mode **ON LINE**, les données sont accessibles

• Le tablespace SYSTEM ne peut pas être mis OFF LINE.

Exemple: ALTER TABLESPACE app_data OFFLINE;

5.1.7. READ-ONLY Tablespace

Exemple:

```
ALTER TABLESPACE app_data READ ONLY;
```

Les données du Tablespace ne sont accessibles qu'en consultation.

5.1.8. Suppression d'un Tablespace

La suppression d'un tablespace entraîne la disparition de tous les objets qu'il contient.

Exemple: DROP TABLESPACE app_data
INCLUDING CONTENTS;

L'ordre DROP ne supprime pas les fichiers du disque; il faut ensuite détruire ces fichiers par les commandes du système d'exploitation.

Avant de supprimer un tablespace, il faut le mettre hors service (OFFLINE).

5.2. Segment

Un segment est un ensemble d'espaces disque alloué à un seul objet.

Un segment ne peut pas s'étendre sur plusieurs tablespaces.

Dans une BD, il y'a plusieurs types de segments :

- Les segments de données (data) pour stocker les données contenues dans les tables.
- Les segments d'index pour stocker les informations d'index.

5.3. Extent

– Ensemble de blocs contigus

– Allouée lorsque le segment est

+ Créé

+ Étendu

+ Modifié

5.4. Bloc

Le bloc est l'unité logique de transfert d'informations entre disque et mémoire centrale. La taille du bloc peut soit être fixée par l'administrateur de la base (paramètre **DB_BLOCK_SIZE** du fichier INIT.ORA).

Un bloc est principalement divisé en trois sous-parties :

• En-tête de bloc : contient des informations sur le bloc : numéro de bloc, la date de création et le type de segment auquel il appartient.

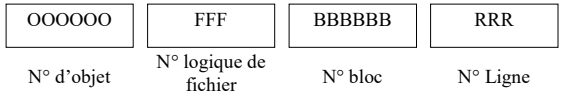
• Espace libre : est une zone vide pouvant être utilisée pour ajouter de nouvelles données ;

• Espace de lignes : contient les données sur les lignes de la table à laquelle le bloc appartient.

- Structure d'une ligne

Chaque enregistrement est identifié par un **ROWID**;

Format de ROWID



Package DBMS_ROWID : permet d'interpréter la valeur du ROWID.

Exemple:

```
SET SERVEROUTPUT ON
SELECT DBMS_ROWID.ROWID_OBJECT (rowid) "n°objet",
    DBMS_ROWID.ROWID_RELATIVE_FNO (rowid) "n°file",
    DBMS_ROWID.ROWID_BLOCK_NUMBER (rowid) "n°bloc",
    DBMS_ROWID.ROWID_ROW_NUMBER (rowid) "n°ligne"
FROM Employe;
/
```

n° objet	n°file	n°bloc	n°ligne
2430	4	2638	0
2430	4	2638	1

6. Gestion des tables

Au moment de sa création, une table est allouée dans un tablespace et est constituée :

- d'un segment de données.
- de zéro, un ou plusieurs segments Index

6.1. Création d'une Table

```
CREATE TABLE Etudiant (
        numero char(4)  primary key,
        nom VARCHAR2 (30),                    } Données
        codefiliere char(2)
    )
    TABLESPACE app_data;                      } Implémentation
```

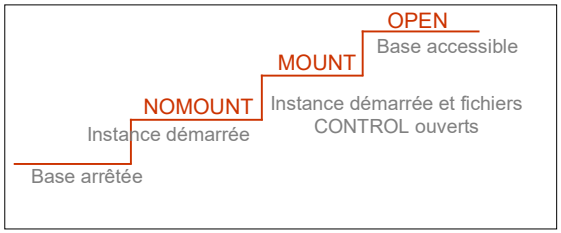

6.2. Modification d'Allocation d'une Table

Exemple:

```
ALTER TABLE Etudiant
Move tablespace USERS;
```


7. Gestion d'une base de données

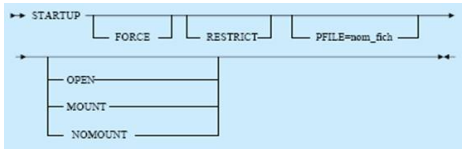
7.1. Les différents statuts d'une base



- Seul un utilisateur «privilégié» vis à vis de l'OS pourra modifier les statuts d'une base.

- NOMOUNT : l'instance seule est démarrée. On en peut déclencher que des pseudo connexions de type 'SYSDBA' ou 'SYSOPER'. Les opérations sont assez limitées. Surtout utile lors de la création de la base.
- MOUNT : l'instance est lancée, la base n'autorise pas de transactions, ni les connexions utilisateurs normales, mais des opérations d'exploitations portant généralement sur les fichiers.
- OPEN : c'est l'état classique, de la base disponible à tous les utilisateurs…

7.1.1. Démarrage de la base



– FORCE : permet de forcer un démarrage quel que soit l'état de la base, en forçant un arrêt brutal puis un startup.

– RESTRICT : démarre la base en accès restreint seulement pour les DBAs. Aucune autre connexion n'est acceptée.

– PFILE: permet de spécifier un fichier d'initialisation autre que le défaut.

NOMOUNT : démarrage de l'instance.

MOUNT : démarrage de l'instance et ouverture des fichiers CONTROL.

OPEN : démarrage complet de la base.

Exemples

- Démarrage complet :

```
SQLDBA> CONNECT
SQLDBA> STARTUP  BTEST
```

- Démarrage en étapes :

```
SQLDBA> CONNECT
SQLDBA> STARTUP NOMOUNT  BTEST
SQLDBA> ALTER DATABASE  BTEST  MOUNT
SQLDBA> ALTER DATABASE  BTEST OPEN
```


7.1.2. Arrêt d'une base

- Nécessite des privilèges au niveau du système d'exploitation.
- Se fait sous SQL*DBA avec la commande SHUTDOWN.
- Ne peut pas se faire en plusieurs étapes.

Exemples :

1. **Arrêt normal** : les nouvelles connexions ne sont pas permises,

```
SQLDBA > SHUTDOWN
Database closed
Database dismount
Oracle instance shut down
```


2. **Arrêt brutal** : arrêt de tous les process de la base (restauration d'instance au démarrage)

```
SQLDBA > SHUTDOWN ABORT
Oracle instance shut down
```

3. **Arrêt immédiate :** les utilisateurs sont déconnectés, les opérations en cours annulées (roolback)

```
SQLDBA > SHUTDOWN IMMEDIATE
```


7.2. Création d'une base

7.2.1. Création de l'instance

Etapes à suivre pour créer une base manuellement

Etape 1. Définir l'arborescence de la base

```
C:\app\...\dbtest\admin, …
```

Étape 2: Création d'une variable DOS ORACLE_SID

```
C:> SET ORACLE_SID=BTEST
```

Etape 3 : Création d'un Service Windows qui gère l' instance.

Nous créons ce service en utilisant ORADIM.EXE.

```
C:> oradim -new -sid BTEST -intpwd mypass -startmode manual
```


Etape 4 : Création du fichier de paramètres Init.ora

Ce fichier est un fichier binaire, appelé SPFILE. Ce fichier binaire est généré à partir d'un fichier ASCII INIT.ORA que nous devons créer.

Paramètres du fichier Init.ora :

CONTROL_FILES= liste des fichiers contrôles files
DB_NAME=BTEST

Etape 5 : Démarrage de l'instance

```
C:> sqlplus /nolog          -- pour ouvrir une session sql plus
SQL> connect / as sysdba    -- pour se connecter en tant qu'admin
```

– création du fichier des paramètres serveur :

```
SQL> Create spfile from pfile='C:\app\vaio\admin\Btest\pfile\\init.ora'
```

démarrage de l'instance :

```
SQL> startup nomount
```

• Résultat :

- la SGA est créée,
- les **BACKGROUND PROCESS** sont démarrés.

7.2.2. Création de la Base

La création d'une nouvelle base de données se fait par l'ordre CREATE DATABASE.

```
CREATE DATABASE nom_base
   [CONTROLFILE REUSE]
   [LOGFILE fichier, …]
   [MAXLOGFILES valeur]
   [MAXLOGMEMBERS valeur]
   [DATAFILES valeur]
   [MAXINSTANCES valeur] ;
```


Paramètres :

– nom_base : nom donné à la BD, composé de 8 caractères au maximum.

– CONTROLFILE : utilisé uniquement en cas de création d'une base suite à la suppression d'une base existante ; permet de réutiliser les fichiers de contrôle de la base précédente.

– MAXLOGMEMBERS valeur : définit le nombre maximal de copies pour un fichier de reprise

Exemple:

```
CREATE DATABASE B_TEST
   DATAFILE 'system.dbs' SIZE 5M
   LOGFILE 'log01.rdo‘ SIZE 200K, 'log02.rdo 'SIZE 200K ;
```

Résultat :

- un fichier **DATABASE** contenant le dictionnaire de données est créé,
- 2 fichiers **REDO LOG** sont créés,
- la base a le statut **OPEN**,
- 2 utilisateurs existent : **SYS/CHANGE_ON_INSTALL et SYSTEM/MANAGER**.

7.2.3. Création des tables système

– Création des vues du dictionnaire de données : Exécuter les scripts **SQL** fournis.

• Sous **SYS**:

- **CATALOG.SQL**: vues et synonymes publics
- **UTLMONTR.SQL**: synonymes publics pour les tables virtuelles

• Sous **SYSTEM**, et pour chaque administrateur de la base :

- **CATDBSYN.SQL**: synonymes sur les vues **DBA_***