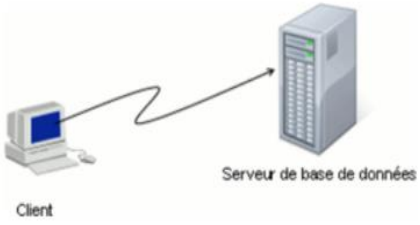
Architecture du serveur de base de données Oracle

Rôles d'un administrateur de BD :

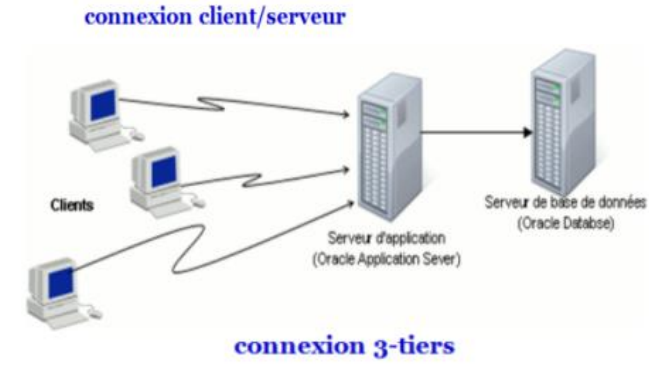
- Installer le serveur de BD, la créer, la démarrer, l'arrêter et manipuler les paramètres d'initialisation.
- Créer et configurer les espaces de stockage physiques et logiques : dimensionner l'espace de stockage physique et logique.
- Créer des utilisateurs et leur affecter des privilèges, des rôles et des profils.
- Auditer et sécuriser la BD.
- Limiter la consommation des ressources et assurer la continuité de fonctionnement et contrôler les indices de performance -> réparer en cas de problèmes.
- Assurer l'export et l'import des données interbase = gestion de flux de données.
- Assurer la sauvegarde et la récupération en cas de pannes.

Serveur de base de données Oracle :

Système de gestion de base de données qui fournit une approche intégrée, complète et ouverte de la gestion des informations. Il se compose d'une instance Oracle et des fichiers de base de données.

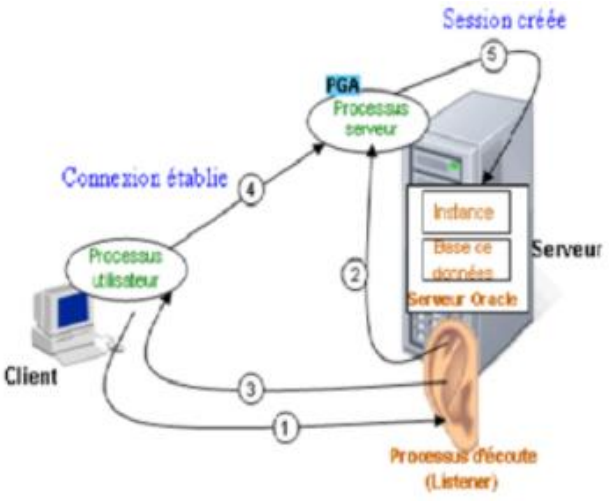


C'est la première architecture du connexion BD : les applications clients envoient les requêtes SQL et PL/SQL au serveur du BD



Présence d'un serveur d'application dont le rôle est d'alléger le travail sur serveur BD

Scénario de connexion à une BD Oracle :



1- Le client contacte le listener Oracle (processus d'écoute : permet d'écouter les requêtes de réseau) et choisit une instance (demande d'un nom de service).

2- Le listener démarre un processus dédié appelé processus serveur : un intermédiaire entre le processus utilisateur et Serveur Oracle -> à chaque processus utilisateur, un processus serveur qui lui est dédié

3- Le listener envoie un accusé de réception au client avec l'adresse du processus serveur dédié.

4- Le client établit une connexion avec le processus serveur dédié.

5- Le processus serveur se connecte à l'instance Oracle pour le compte du processus utilisateur (création d'une session utilisateur).

Architecture : un serveur Oracle se compose d'une instance Oracle (zones mémoires & processus d'arrière-plan) et d'une BD Oracle.

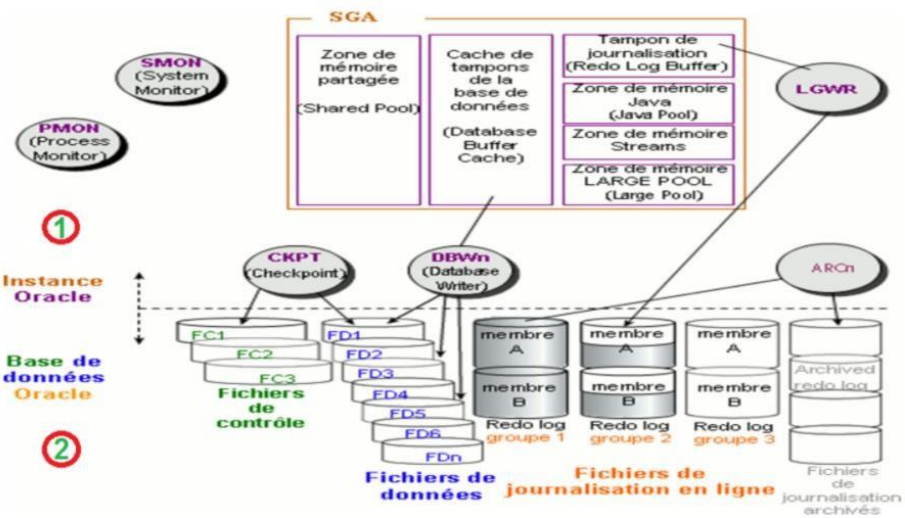
BD Oracle : BD physique :

->fichiers de contrôle : contient des infos sur la base (son nom, stockage, les tables...)

->fichiers de données : les instances des données des utilisateurs sont stockées le dictionnaire de données

->fichiers de journalisation : un pt de restauration de données dans la bd

Une instance Oracle est composée d'une SGA (System Global Area : accessible à tous les utilisateurs, allouée au démarrage et libérée à l'arrêt), structure de mémoire partagée par tous les processus serveurs et les processus en arrière-plan (utilisateur), et de plusieurs processus oracle en arrière-plan ayant chacun un rôle.



System Global Area (SGA) : (représente 2% de la taille du BD) constitué de

Zones de mémoire obligatoires :

- Zone de mémoire partagée (Shared Pool). (40% de la taille du SGA)
- Cache de tampons de la BD (DB buffer cache). (50% de la taille du SGA)
- Tampon de journalisation (Redo log buffer). (10% de la taille SGA)

Zones mémoire non obligatoires :

- Zone de mémoire LARGE POOL.
- Zone de mémoire Java (Java Pool).
- Zone de mémoire Streams (Streams pool).

Zone de mémoire partagée (Shared Pool) :

Elle est constituée de 2 structures mémoire qui assurent la performance :

- Cache du dictionnaire de données (row cache) :

Quand un utilisateur soumet une requête SQL, le processus serveur extrait des informations stockées dans les tables du dictionnaire (données du compte utilisateur, noms des fichiers de données, noms des segments de tables et index, emplacements d'extents, descriptions des tables et privilèges utilisateur,

etc.) et les place dedans pour des besoins de réutilisation.

Au cours des prochaines analyses, le processus serveur recherche les informations dans le cache du dictionnaire pour résoudre les noms d'objet et valider l'accès -> à chaque requête on récupère l'information du cache et pas du mémoire physique -> Améliorer temps du réponse.

- Cache "Library" :

Il conserve, à des fins de partage, des informations sur les commandes SQL (utiles pour les commandes dansés) et le code PL/SQL récents soumis par des utilisateurs de la BD → Optimiser les ressources : charger une instruction 1 seul fois pour multiple instructions.

Ex : SELECT * FROM EMPLOYEES (stocké dans cache library avec le plan d'execution de cette requête) -> structure de la table, emplacement des fichiers de données sont stockés dans row cache.

Cache de tampons de la BD (DB buffer cache) :

Géré par un algorithme LRU (List Recently Used : algorithme de remplacement qui remplace les lignes de cache en cherchant le traitement le plus anciennement référencé afin de ne pas surchager le cache de tampon), il conserve des copies des blocs de données extraits des fichiers.

Lorsqu'un client lance une requête de SELECT, le résultat va être stockée dans ce cache de tampon.

+ Gain de performances lors de l'obtention et de la mise à jour de données.

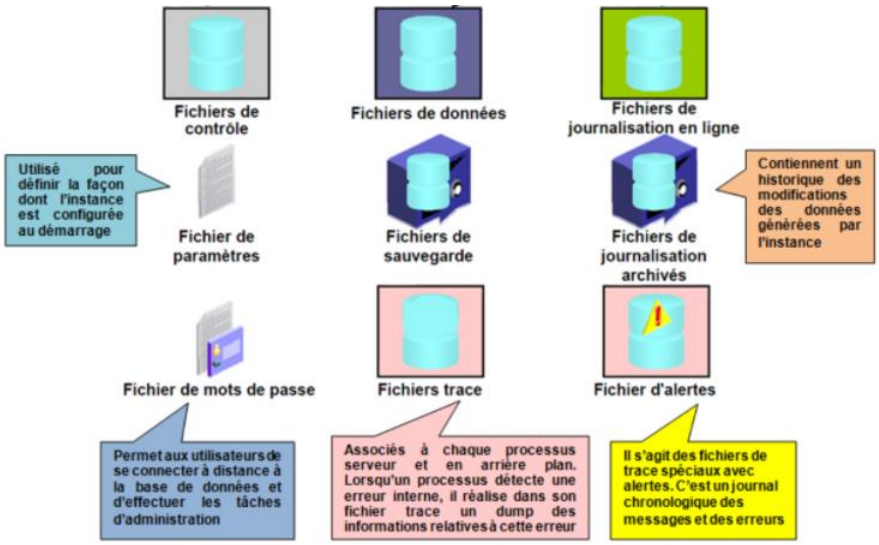
DB_BLOCK_SIZE : détermine la taille du bloc principal de ce cache de tampon.

Tampon de journalisation (Redo log buffer) :

Il récupère les données (point de restauration) et enregistre toutes les modifications apportées aux blocs de données de la BD (historique). Ces modifications constituent des entrées de journalisation qui vont être écrits dans des fichiers de journalisation.

LOG_BUFFER : définit la taille du tampon -> l'administrateur gère le tampon via cette taille.

NB : si on arrête l'instance oracle, les données dans SGA (cache) seront perdus.



Fichiers de contrôle / Fiche descriptive de serveur : contient des infos sur la base (son nom, emplacement de fichiers de données, les tables (description des objets et des données) ...)

Fichiers de données : les blocks des données, métadonnées, structure des tables, ensembles des enregistrements, index, vues, fonctions & procédures sont stockés la

Fichiers de journalisation : un pt de restauration de données dans la bd (historique de modifications).

Les processus en arrière-plan :

Rôle : exécuter des activités asynchrones d'écriture et de contrôle et d'écrire les données de cache dans la BD (structure physique) -> correspondre les SGA avec fichiers du BD

Processus DBWn (DB Writer) : écrit les blocs modifiés du cache tampon de la BD d'une façon permanente dans les fichiers de données dans les cas suivants :

- Point de reprise
- Evènement déclenché avec la commande COMMIT
- Seuil des tampons "dirty" atteint
- Aucune mémoire tampon disponible
- Temps imparti dépassé
- Demande de ping RAC
- Tablespace hors ligne ou en lecture seule
- DROP ou TRUNCATE sur une table
- BEGIN BACKUP sur un tablespace

Processus LGWR (Log Writer) : écrit les entrées de journalisation (qui se trouve dans le tampon de journalisation) sur le disque (fichiers de journalisation) dans les cas suivants :

- Evènement déclenché avec la commande COMMIT : Validation
- Un tiers du cache est occupé
- La journalisation atteint 1 Mo toutes les trois secondes avantque le processus DBWn ne procède à une opération d'écriture.

Processus SMON (System Monitor) : assure la récupération en cas de panne (réimplémente des modifications dans les fichiers de journalisation, ouvre BD pour l'accès utilisateurs et annule les transactions non validées ou non suivi par COMMIT), la fusion de l'espace libre et la libération des segments temporaires (pour améliorer temps de réponse).

Processus CKPT (Check Point) : informe le processus DBWn des points de reprise (restaurations) pour lancer la mise à jour, met à jour les entête de fichiers de données et de contrôle avec les informations sur le point de reprise.

PMON (Process Monitor) : assure le nettoyage des processus utilisateur en cas d'échec -> transaction défaillantes (session abandonnée) : annule la transaction, libère des verrous et d'autres ressources, redémarre les répartiteurs interrompus, etc.

Program Global Area (PGA):

C'est une structure de mémoire créée pour chaque utilisateur connecté (zone mémoire non partagée). Elle stocke des informations de contrôle spécifiques à la session de l'utilisateur (Ex : zones privées pour le traitement des curseurs, variables attachées (bind), informations sur la session, etc.)

Chaque processus serveur ou en arrière-plan dispose de sa propre mémoire PGA privée qui lui est exclusivement

réservée.

Lorsque le processus utilisateur se déconnecte, le processus serveur associé prend fin et la m mémoire PGA est libérée.

NB : PMON fait le même travail que SMON mais SMON nettoie zone mémoire partagée SGA et PMON nettoie zone mémoire non partagée (PGA)

Le dictionnaire de données :

Le dictionnaire de données Oracle (table systéme) est la description d'une BD (toutes les informations des objets du BD), créé et mis à jour en même temps qu'elle.

DICTIONARY/dict : accessible en lecture seulement et contient le nom et la description des tables et vues du dictionnaire.

```
DESCRIBE DICTIONARY
```

Name	Null?	Type
TABLE_NAME		VARCHAR2(30)
COMMENTS		VARCHAR2(4000)

TABLE_NAME est nom de la vue qui contient la description des données.

Il appartient à l'utilisateur SYS, mais les autres utilisateurs peuvent y accéder par le biais de vues prédéfinies :

Préfixe **USER** : vue de l'utilisateur connecté (ce que contient son schéma : liste des objets).

Préfixe **ALL** : vue étendue de l'utilisateur (tous les objets qu'il peut accéder) : les tables que j'ai créées et les tables que les autres ont crées mais qui permit de manipuler et visualiser.

Préfixe **DBA** : vue de l'administrateur de la BD (ce que contient le schéma de chaque user).

Préfixe **V$** : vue dynamique qui conient données de performances (taille mémoire, user connectés, etc.).

Exemples :

SELECT * FROM dictionary WHERE table_name = USER_OBJECTS ;

USER_OBJECTS : tous les objets dont vous êtes propriétaire.

```
SELECT object_name, object_type, created, status
FROM   user_objects
ORDER BY object_type;
```

OBJECT_NAME	OBJECT_TYPE	CREATED	STATUS
REG_ID_PK	INDEX	10-DEC-03	VALID
...			
DEPARTMENTS_SEQ	SEQUENCE	10-DEC-03	VALID
REGIONS	TABLE	10-DEC-03	VALID
LOCATIONS	TABLE	10-DEC-03	VALID
DEPARTMENTS	TABLE	10-DEC-03	VALID
JOB_HISTORY	TABLE	10-DEC-03	VALID
JOB_GRADES	TABLE	10-DEC-03	VALID
EMPLOYEES	TABLE	10-DEC-03	VALID
JOBS	TABLE	10-DEC-03	VALID
COUNTRIES	TABLE	10-DEC-03	VALID
EMP_DETAILS_VIEW	VIEW	10-DEC-03	VALID

NB : Vue ALL_OBJECTS permet d'afficher tous les objets auxquels vous avez accès

```
SELECT table_name
FROM   user_tables;
```

TABLE_NAME
JOB_GRADES
REGIONS
COUNTRIES
LOCATIONS
DEPARTMENTS

USER_TABLES : les informations relatives aux tables dont vous êtes propriétaire (nom, nom du tablespace, nom du cluster et nom IOT).

USER_TAB_COLUMNS : les informations relatives aux colonnes des tables dont vous êtes propriétaire**.**

```
SELECT column_name, data_type, data_length,
       data_precision, data_scale, nullable
FROM   user_tab_columns
WHERE  table_name =  EMPLOYEES ;
```

COLUMN_NAME	DATA_TYPE	DATA_LENGTH	DATA_PRECISION	DATA_SCALE	NUL
EMPLOYEE_ID	NUMBER	22	6	0	N
FIRST_NAME	VARCHAR2	20			Y
LAST_NAME	VARCHAR2	25			N
EMAIL	VARCHAR2	25			N

USER_CONSTRAINTS : définition de contraintes sur les tables (nom, type, nom du table….)

```
SELECT constraint_name, constraint_type,
       search_condition, r_constraint_name,
       delete_rule, status
FROM   user_constraints
WHERE  table name =  EMPLOYEES ;
```

CONSTRAINT_NAME	CON	SEARCH_CONDITION	R_CONSTRAINT_NAME	DELETE_RULE	STATUS
EMP_LAST_NAME_NN	C	"LAST_NAME" IS NOT NULL			ENABLED
EMP_EMAIL_NN	C	"EMAIL" IS NOT NULL			ENABLED
EMP_HIRE_DATE_NN	C	"HIRE_DATE" IS NOT NULL			ENABLED

USER_CONS_COLUMNS : colonnes dont vous êtes propriétaire & définies dans des contraintes.

```
SELECT constraint_name, column_name
FROM   user_cons_columns
WHERE  table_name =  EMPLOYEES ;
```

CONSTRAINT_NAME	COLUMN_NAME
EMP_EMAIL_UK	EMAIL
EMP_SALARY_MIN	SALARY
EMP_JOB_NN	JOB_ID
EMP_HIRE_DATE_NN	HIRE_DATE

USER_VIEWS:

1
```
DESCRIBE user_views
```

Name	Null?	Type
VIEW_NAME	NOT NULL	VARCHAR2(30)
TEXT_LENGTH		NUMBER
TEXT		LONG

2
```
SELECT DISTINCT view_name FROM user_views;
```

VIEW_NAME
EMP_DETAILS_VIEW

3
```
SELECT text FROM user_views
WHERE view_name =  EMP_DETAILS_VIEW ;
```

TEXT
SELECT e.employee_id, e.job_id, e.manager_id, e.department_id, d.locat ion_id, l.country_id, e.first_name, e.last_name, e.salary, e.commissio n_pct, d.department_name, j.job_title, l.city, l.state_province, c.cou ntry_name, r.region_name FROM employees e, departments d, jobs j, loca tions l, countries c, regions r WHERE e.department_id = d.department_id AN D d.location_id = l.location_id AND l.country_id = c.country_id AND c.region _id = r.region_id AND j.job_id = e.job_id WITH READ ONLY

USER_SEQUENCE:

```
SELECT   sequence_name, min_value, max_value,
         increment_by, last_number
FROM     user_sequences;
```

SEQUENCE_NAME	MIN_VALUE	MAX_VALUE	INCREMENT_BY	LAST_NUMBER
LOCATIONS_SEQ	1	9900	100	3300
DEPARTMENTS_SEQ	1	9990	10	280
EMPLOYEES_SEQ	1	1.0000E+27	1	207

La colonne **LAST_NUMBER** affiche le prochain numéro de séquence disponible.

USER_SYNONYMS :

```
SELECT *
FROM   user_synonyms;
```

SYNONYM_NAME	TABLE_OWNER	TABLE_NAME	DB_LINK
EMP	ORA1	EMPLOYEES	

Les vues V$:

Ce sont des vues dynamiques qui enregistrent l'activité en cours de la BD. Elles sont constamment mises à jour lorsque la BD est active et accessibles par un DBA.

Les informations sont lues à partir de la mémoire et du fichier de contrôle. Exemples :

V$session : affiche les sessions en cours.

V$logfile : affiche la liste des fichiers journaux.

V$log : affiche le statut des groupes de fichiers journaux.

Démarrage de la BD Oracle :

Fichiers de paramètres d'initialisation :

Une BD oracle n'est disponible à l'utilisateur que lorsque le DBA a démarré une instance et ouvert la BD.

Pour démarrer une instance, le serveur Oracle doit lire le fichier de paramètres d'initialisation qui contient les paramètres d'initialisation pour allouer la SGA et démarrer les processus d'arrière-plan.

Il existe deux types de fichiers de paramètres d'initialisation :

Fichier de Paramètres statiques PFILE : (nommé initSID.ORA)

Ils sont des fichiers textes qui peuvent être modifiés manuellement à l'aide d'un éditeur que dans le fichier de paramètres.

Un redémarrage de l'instance est nécessaire pour que les modifications prennent effet.

Ce fichier doit être stocké côté client

Fichier de Paramètres dynamiques (persistant) SPFILE : (nommé spfileSID.ORA)

C'est un Fichier binaire qui va contenir les paramètres

Ce fichier doit être stocké côté serveur.

Ne peut être modifié (pas manuellement) que par le serveur du BD (à travers les commandes SQL)

NB :

On peut créer un fichier pfile via un fichier spfile : create pfile [='nom_pfile'] From Spfile [='nom_spfile'];

On peut créer un fichier spfile à travers un fichier pfile : create spfile [='nom_spfile'] From pfile [='nom_pfile'] ;

(nom_pfile / nom_spfile : spécifier nom ou chemin)

Paramètres d'initialisation :

Les paramètres d'initialisation permettent de spécifier le nom de la BD, l'emplacement des fichiers de contrôle, le répertoire de destination des fichiers de données (datafiles), la destination des fichiers de journalisation (redo log files), définir des limites par rapport des ressources SGA, valeurs affectées aux structures mémoire SGA (paramètres qui déterminent la taille du bloc SGA : DB_BLOCK_SIZE…)…

Re les paramètres d'initialisation sont manipulés par un DBA (Admin BD)

Paramètre	Description
CONTROL_FILES	Un ou plusieurs noms de fichier de contrôle
DB_FILES	Nombre maximal de fichiers de base de données
PROCESSES	Nombre maximal de processus utilisateur du système d'exploitation pouvant se connecter simultanément
DB_BLOCK_SIZE	Taille de bloc de base de données standard utilisée par tous les tablespaces
DB_CACHE_SIZE	Taille du cache de tampons de bloc standard
SGA_TARGET	Taille totale de tous les composants SGA
MEMORY_TARGET	Mémoire utilisable à l'échelle du système Oracle
PGA_AGGREGATE_TARGET	Quantité de mémoire PGA allouée à tous les processus
SHARED_POOL_SIZE	Taille de la zone de mémoire partagée (en octets)
UNDO_MANAGEMENT	Mode de gestion du volume d'annulation à utiliser

Il existe deux types de paramètres d'initialisation :

Paramètres statiques :

Ils ne peuvent être modifiés que dans le fichier de paramètres uniquement à l'aide des commandes **ALTER**

SYSTEM avec l'option **SCOPE='SPFILE'** et stockés dans les fichiers de paramètres statiques.

Un redémarrage de l'instance est nécessaire pour que les modifications prennent effet (ajouter le paramètre **SCOPE='SPFILE'** dans la requête**)**

ALTER SYSTEM SET parameter=value COMMENT='. . . ' [SCOPE = MEMORY| SPFILE | BOTH] ;

COMMENT : donner un commentaire (Ex : pourquoi on veut changer cette valeur là ?)

SCOPE : Spécifier à quel moment le changement doit prendre effet (MEMORY : changement prend effet immédiatement, dure jusqu' à la base soit arrêtée et en mémoire (instance courante), SPFILE, BOTH)

BOTH est la valeur par défaut si la BD a été démarré en utilisant un SPFILE

MEMORY est la valeur par défaut et la seule portée si la BD a été démarré en utilisant un PFILE

Paramètres dynamiques :

Ils peuvent être modifiées tant que la BD est en ligne (en cours de l'instance) et sont valides pour la durée de la session ou dans les limites établies par le paramètre **SCOPE** (où on va mettre cette modification).

Les modifications se font en deux niveaux : niveau session **ALTER SESSION** et niveau système **ALTER SYSTEM**.
ALTER SESSION SET parameter=value ;

On peut visualiser les infos % aux paramètres d'initialisation via vue V$PARAMETER

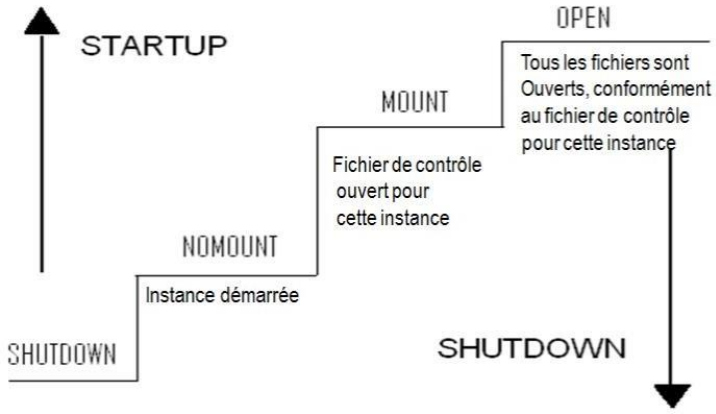
```
SQL> desc V$PARAMETER
 Nom                                       NULL ?   Type
 ----------------------------------------- -------- ----------------
 NUM                                                NUMBER
 NAME                                               VARCHAR2(80)
 TYPE                                               NUMBER
 VALUE                                              VARCHAR2(512)
 DISPLAY_VALUE                                      VARCHAR2(512)
 ISDEFAULT                                          VARCHAR2(9)
 ISSES_MODIFIABLE                                   VARCHAR2(5)
 ISSYS_MODIFIABLE                                   VARCHAR2(9)
 ISINSTANCE_MODIFIABLE                              VARCHAR2(5)
 ISMODIFIED                                         VARCHAR2(10)
 ISADJUSTED                                         VARCHAR2(5)
 ISDEPRECATED                                       VARCHAR2(5)
 DESCRIPTION                                        VARCHAR2(255)
 UPDATE_COMMENT                                     VARCHAR2(255)
 HASH                                               NUMBER
```

- NAME : nom du paramètre
- TYPE : type du paramètre
- VALUE : la valeur
- ISDEFAULT : FALSE : s'il est initialisé par le fichier de paramètres TRUE :sinon (valeur par défaut)
- Isxxx_MODIFIABLE : indique si le paramètre peut être modifié par session (ISSES_MODIFIABLE) ou par instance (ISSYS_MODIFIABLE)

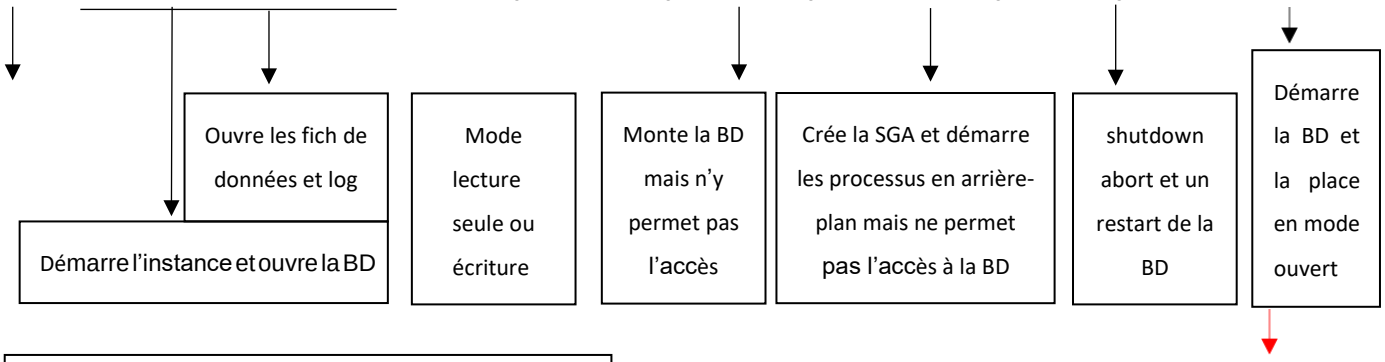
Démarrage de la BD :

Lors de démarrage le serveur Oracle va lire les paramètres d'initialisation & allouer les zones mémoire SGA.

Différents modes de démarrages sont possibles :



STARTUP [OPEN [READ {ONLY | WRITE }] | MOUNT | NOMOUNT | FORCE | RESTRICT]



Annulation : **ALTER SYSTEM DISABLE RESTRICTED SESSION**

FORCE est utilisé lorsque je ne peux pas démarrer en MOUNT ou en NOMOUNT (en mode arrêt brusque)
RESTRICT : accès seulement aux utilisateurs qui ont les privilèges RESTRICTED SESSION (pour effectuer des opérations de tuning, import/export,… sans que personne ne se connecte à BD)

La commande ALTER :

Lorsque l'instance est démarré, ALTER permet de faire passer la BD du statut NOMOUNT à MOUNT ou du statut MOUNT à OPEN :

- ALTER DATABASE { NOMOUNT | MOUNT }
- ALTER DATABASE { MOUNT | OPEN }

Pour que les données ne soient pas modifiées, on peut ouvrir la base en mode read-only :

- ALTER DATABASE OPEN [READ WRITE| READ ONLY]

Arrêt de la BD :

Fermer la BD :

- Quand la BD se ferme, Oracle écrit les changements de la BD buffer cache et les entrées du buffer redo log dans les data files et redo log files (mapper les entrées du cache vers les fichiers lors de COMMIT).
- Les control files restent ouverts.

Démonter la BD d'une instance :

- Fermeture des control files.

Arrêt de l'instance :

- Libération de la SGA.
- Arrêt des background processes.
- Fermeture des fichiers trace et ALERT (crées avec le démarrage de l'instance : enregistrer l'historique des messages d'erreur de démarrage).

SHUTDOWN [NORMAL | TRANSACTIONNEL | IMMEDIATE | ABORT]

Arrêt NORMAL (par défaut) :

- Le serveur attend la déconnexion de tous les utilisateurs avant de terminer l'arrêt (temps d'attente peut être important).
- Oracle ferme et démonte la BD avant d'arrêter l'instance.
- Pas de restauration de l'instance lors du démarrage suivant (car toutes les informations modifiées encore présentes dans la SGA sont écrites dans les data files et redo log files).

Arrêt TRANSACTIONNEL :

- Le client ne sera plus connecté dès la fin de la transaction en cours -> ne pas autoriser des nouvelles connexions.
- Pas de restauration de l'instance lors du démarrage suivant.

Arrêt IMMEDIATE : **arrêter d'une façon immédiate**

- Le serveur n'attend pas la déconnexion des utilisateurs avant de terminer l'arrêt.
- Les ordres SQL en cours ne seront pas traités.
- Oracle ferme et démonte la base avant d'arrêter l'instance.
- Pas de restauration de l'instance lors du démarrage suivant.

Arrêt ABORT : **forcer l'arrêt de l'instance**

- Le serveur oracle n'attend pas la déconnexion des utilisateurs avant de terminer l'arrêt.
- Les transactions non commitées ne seront pas effectuées (rollback).
- L'instance sera fermée sans la fermeture des fichiers.
- Restauration de l'instance lors du démarrage suivant.

TP : (Vérifier la correction du TP)

Liste des utilisateurs créés sur le serveur : **SELECT USERNAME,CREATED FROM DBA_USERS;**

Liste des utilisateurs connectés sur votre instance courante**. SELECT USERNAME FROM V$SESSION WHERE TYPE='USER';**

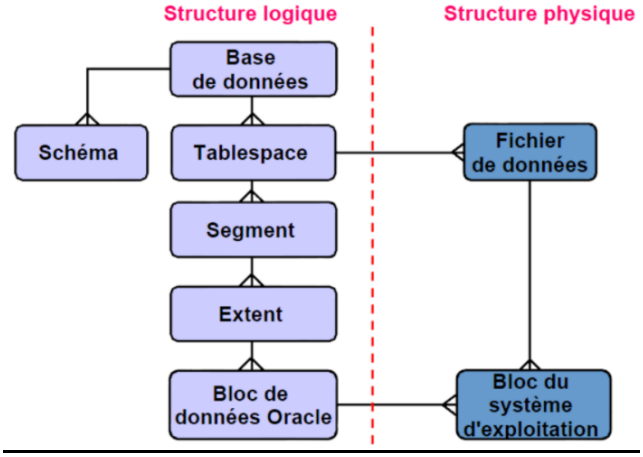
Taille totale de la SGA : **SELECT SUM(VALUE) FROM V$SGA;**

Nombre total des tables créés dans le serveur : **select count(*) from dba_tables; --ou all_tables**

Nombre total de table créés par l'utilisateur HR : **select count(*) from all_tables where owner='HR';**

Les structures de stockage Oracle

Structures de stockage logiques et physiques :



Structure Logique	Structure Physique
Les données sont stockées par oracle dans des tablespaces : divisés en unités de stockage logiques, représenté par 1/plusieurs fichiers de données,	Les données sont stockées dans des fichiers de données : ne peuvent appartenir qu'à un seul tablespace et à une seule base de données
Segments	Tables

-< : contient

Tablespaces : Une BD est définie avec au moins un tablespace (SYSTEM tablespace) contenant le dictionnaire de données, n'appartenant qu'à une seule BD.

NB : Oracle recommande de ne pas stocker d'autres objets outre le dictionnaire de données dans le SYSTEM tablespace.

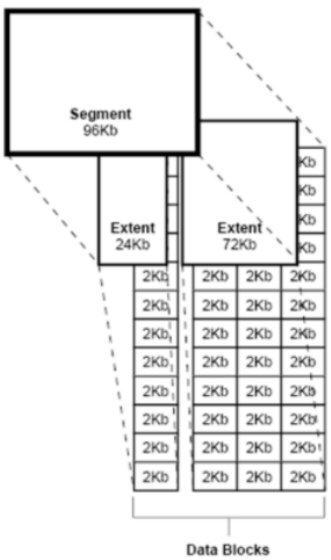
Un tablespace ne peut jamais être désactivé et peut être :

- Actif (online) : ses données sont accessibles aux utilisateurs.
- Désactivé (offline) : ses données ne sont plus accessibles aux utilisateurs.

Définir et paramétrer différemment plusieurs tablespaces permet au DBA de :

- Organiser la base : assigner des quotas de ressources aux utilisateurs et contrôler la disponibilité des données en mettant hors service ou en lecture seule certains tablespaces.
- Améliorer la performance en répartissant les zones de stockage sur plusieurs disques.

Extension et segments :

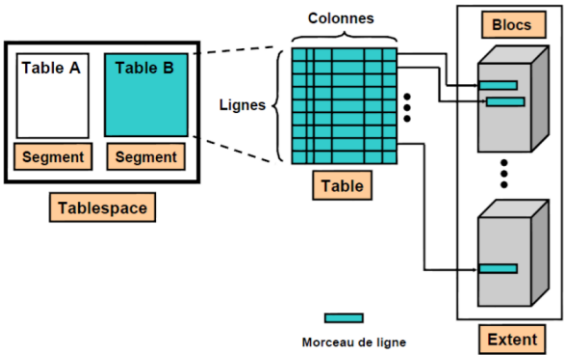


Extension (ou extents) : C'est une suite contiguë de blocs (au sens de l'emplacement sur le disque). Une extension est affectée à un type de données (ex : enregistrements d'une table) et le nombre de blocs dans celle-ci est fixé par le DBA.

Segments : C'est un ensemble d'extensions. Chaque segment est dédié au stockage d'un type particulier d'informations (tables, index, etc.)

une extension initiale est allouée lors de la création d'un segment et de nouvelles extensions sont allouées dynamiquement si besoin

Manipuler les tablespaces :



Gestion de l'espace dans les tablespaces :

Tablespace géré localement (recommandé) :

- Les extents libres sont gérés dans le tablespace.
- Un bitmap est utilisé pour enregistrer les extents libres.
- Chaque bit correspond à un bloc.
- La valeur du bit indique si le bloc est libre ou utilisé.

Tablespace géré au moyen du dictionnaire :

- Les extents libres sont gérés par le dictionnaire de données.
- Les tables appropriées sont mises à jour lorsque des extents sont alloués ou libérés.

Créer un TABLESPACE :

```
CREATE {BIGFILE|SMALLFILE} TABLESPACE nom_tablespace
[DATAFILE ['nom_fichier'] [SIZE integer {K|M|G|T}]
[AUTOEXTEND
      {OFF| ON [NEXT integer {K|M|G|T}]
      [MAXSIZE {UNLIMITED | integer {K|M|G|T}] } ] [,...]
]
[BLOCKSIZE integer [K]]
[{LOGGING|NOLOGGING}] [{ONLINE|OFFLINE}]
[Extent_mangement_clause]
[DEFAULT storage_clause] ;
```

Étendre l'espace de stockage d'un datafile avec une limite -> Extension automatique
OFF : max atteint -> pas d'ajout
ON : lorsqu'il atteint le maximum il va calculer selon la taille donnée jusqu'à atteindre la MAX_SIZE

Journalisation ou pas des modifications.

Exemple :

```
CREATE TABLESPACE TBL01
DATAFILE 'C :\ORACLE\PRODUCT\10.2.0\ORADATA\ORCL\FDAUTO1.dbf'
SIZE 10M
AUTOEXTEND ON NEXT 2M MAXSIZE 20M ;
```

FDAUTO1.dbf : fichier de données

Créer plusieurs datafiles dans le même tablespace :

CREATE TABLESPACE TBL01
DATAFILE 'fd01tbl01.dbf' SIZE 6M,
'fd02tbl01.dbf' SIZE 4M;

Modifier un TABLESPACE :

Modifier la taille d'un tablespace :

- Ajouter de l'espace à un tablespace existant en lui ajoutant des fichiers de données : **alter tablespace TBL01 add datafile 'C :\ORACLE\ORADATA\BDSCO2\TBL01fd03.dbf'**
-
- **SIZE 30M AUTOEXTEND ON NEXT 5M MAXSIZE 100M ;**

NB : Pour le chemin, Oracle stocke toujours sous son repo.

- Modifier taille de fichier existant :
  Alter database datafile 'C:\ORACLE\ORADATA\BDSCO2\TBL01fd03.dbf' resize 50 M ;

Read only : arrête toutes les écritures dans le tablespace. Les transactions en cours peuvent se terminer, mais aucune nouvelle instruction LMD ou autre activité d'écriture n'est autorisée à démarrer sur le tablespace :
alter tablespace TBL01 read only ;

Read write : permet le lancement d'instructions LMD et d'autres activités d'écriture sur les objets du tablespace : **Alter tablespace TBL01 read write ;**
Online : rendre le tablespace accessible par tous les utilisateurs (disponible) : **Alter tablespace tbl01 online ;**
Offline : rendre le tablespace non accessible (indisponible) : **Alter tablespace tbl01 offline ;**

Supprimer un TABLESPACE :

Supprimer logique : **drop tablespace nom_tablespace**

Supprimer logique et physique : **drop tablespace nom_tablespace and datafiles**

Supprimer les objets stockés dans le tablespace : **drop tablespace nom_tablespace including contents**

Suppression des contraintes d'intégrité : **drop tablespace nom_tablespace including contents cascade constraints**

Pour supprimer le tablespace dans le cas où il existe déjà **: DROP tablespace TBL01 including contents and datafiles;**

Consulter les informations relatives aux tablespaces :

Ces informations peuvent être consultée via l'interrogation de ces vues :

- Informations relatives aux tablespaces :
  - DBA_TABLESPACES , V$TABLESPACE
- Informations relatives aux fichiers de données :
  - DBA_DATA_FILES , V$DATAFILE
- Informations relatives aux fichiers temporaires :
  - DBA_TEMP_FILES , V$TEMPFILE
- Le dictionnaire de données contient des vues qui décrivent l'organisation de la base et l'utilisation de l'espace :
  - DBA_EXTENTS : liste des extensions
  - DBA_SEGMENTS : liste des segments
  - DBA_FREE_SPACE : espace disponible restant dans les tablespaces
  - DBA_TABLESPACES : liste des tablespaces
  - DBA_DATA_FILES : liste des fichiers de données
  - DBA_TEMP_FILES : liste des fichiers temporaires des tablespaces
- Le paramètre de taille du block par defaut :
  - DB_BLOCK_SIZE

NB :

- La taille totale est calculée par la somme du data files .
- La taille occupée est calculée par la somme des segment

Veuillez consulter le TP et la correction.

Administrer la sécurité utilisateur

Compte utilisateur de la BD :

Pour pouvoir accéder aux données stockées, on doit se connecter via un compte d'utilisateur aura certains priviléges une certaine visibilité de la base de données

A chaque création d'un compte utilisateur de BD il faut définir un nom unique (login) et une méthode d'authentification (password). Les éléments pouvant être attribués par défaut sont : un tablespace par défaut, quotas sur un ou plusieurs tablespaces, un tablespace temporaire, un profil utilisateur, un statut de compte (verrouillé ou expiré), etc.

Une fois crée, il ne peut pas être modifié.

Pour consulter les informations des utilisateurs, on utilise la vue DBA_USERS.

Compte prédéfinis SYS et SYSTEM : (par défaut pour tous les utilisateurs Oracle)

L'Utilisateur SYS	L'Utilisateur SYSTEM
Super Administrateur DBA	Administrateur DBA
Propriétaire des tables et des vues du dictionnaire de données	Peut seulement consulter les objets au niveau du serveur
Peut faire les opérations de démarrage et d'arrêt du BD, ainsi que pour certaines commandes de maintenance	Pas de manipulation

NB : HR est un utilisateur par défaut.

Authentification des utilisateurs :

Oracle fournit plusieurs méthodes d'authentification :

*Operating System (authentification par le système d'exploitation) :

Mode d'authentification qui nécessite :

- 1 seule authentification
- Le privilège SYSDBA ou SYSOPER (permet de récupérer les mots de passe des utilisateurs depuis un fichier de mots de passe externe (external password file) qui peut être lu par l'instance (même si la BD n'est pas ouverte).

CONNECT / AS [SYSOPER | SYSDBA] ;

NB : external password file peut être utilisé lorsqu'on a oublié le mot de passe.

*Password File Authentication :

Mode d'authentification qui nécessite le privilège SYSDBA ou SYSOPER.

CONNECT username / password AS [SYSOPER | SYSDBA] ;

*Password (authentification par la BD Oracle) :

C'est le mode le plus courant et le mode par défaut -> l'utilisateur se connecte par le mot de passe stockés dans la base de données -> BD doit être ouverte.

Associer à chaque utilisateur créer un mot de passe qu'il devra saisir lors de chaque connexion.

CREATE USER <name> IDENTIFIED BY <password> ;

Externe :

Oracle délègue la tâche d'authentification à un service externe.

Si l'option de sécurité avancée est autorisée -> le service externe peut être un serveur Kerberos, un serveur RADIUS, ou (dans l'environnement Windows), le service d'authentification natif de Windows.

Sinon l'authentification se fait par le système d'exploitation : créer les utilisateurs ayant les mêmes noms d'utilisateur du système d'exploitation préfixées avec la valeur du paramètre OS AUTHENT PREFIX (sous windows on doit ajouter le domaine windows aussi au préfixé).

CREATE USER <name> IDENTIFIED EXTERNALLY ;

```
CREATE USER nom IDENTIFIED BY mot_de_passe
[ DEFAULT TABLESPACE nom_tablespace ]
[ TEMPORARY TABLESPACE nom_tablespace ]
[ QUOTA { valeur [K|M] | UNLIMITED } ON nom_tablespace [,...] ]
[ PROFILE nom_profil ]
[ PASSWORD EXPIRE ]
[ ACCOUNT { LOCK | UNLOCK } ];
```

Nom : unique, pas de caractères spéciaux, ne dépasse pas 30 caractères,

NB :

- Les autres attributs vont prendre des valeurs par défaut
- Tablespace prise par défaut et celle de la BD
- **Nom et mot_de_passe sont obligatoires**

QUOTA : permet à l'utilisateur de créer ses objets selon un stockage prédéfini.

TEMPORARY TABLESPACE : tablespace temporaire qui ne permet pas à l'utilisateur de créer des objets réservés au système pour pouvoir stocker des objets temporaires de la session

PASSWORD EXPIRE : définir un utilisateur avec un mot de passe expiré dès la première connexion il lui demande de renouveler son mot de passe -> **ALTER USER nom IDENTIFIED BY password ;**

ACCOUNT LOCK : utilisateur verrouillé -> modifier l'état : **ALTER USER name IDENTIFIED BY password UNLOCK ;**

Global :

Ce mode d'authentification permet d'identifier les utilisateurs via Oracle Internet Directory.

CREATE USER <name> IDENTIFIED GLOBALLY ;

Privilèges :

Plusieurs utilisateurs peuvent accéder aux objets de la BD -> l'autorisation peut être contrôlée sur ces utilisateurs.

C'est le droit d'exécuter un type particulier d'instruction SQL ou d'accéder à l'objet d'un autre utilisateur.

- Attribution de privilège : GRANT
  - exemples :
    - GRANT CREATE SESSION TO HR
    - GRANT SELECT, UPDATE(SALARY) ON EMPLOYEES TO SCOTT ;

- Déclination de privilège : REVOKE
  - exemples :
    - REVOKE CREATE SESSION FROM HR
    - REVOKE SELECT, UPDATE(SALARY) ON EMPLOYEES FROM SCOTT ;

GRANT CREATE SESSION TO HR : donner l'autorisation de se connecter à HR -> si on ne donne pas ce privilège : Error user lacks CREATE SESSION privilege.

GRANT UPDATE(SALARY) : donner le privilége de modifier seulement la colonne salary

Il existe deux types de privilèges utilisateur :

Privilèges système : privilèges d'accès aux objets

Autorisent un utilisateur à effectuer certaines opérations de la BD (Ex : tablespace, sessions).

Permettent création suppression modification des structures de données (CREATE/ALTER/DROP)

Ces privilèges peuvent être accordé à un administrateur ou un utilisateur à la permission d'administrer ces objets (Un administrateur donne les droits d'administration (accorder des privilèges) à cet utilisateur)

Ex de privilèges qui ont été créés et stockés par la BD Oracle :

- CREATE CLUSTER (segments de données dans le chemin propriétaire)

- CREATE/ALTER/DROP ANY CLUSTER (segment de données dans n'importe quel chemin)
- ALTER DATABASE (modification de la base de données)
- CREATE/ALTER/DROP ANY INDEX ….
- GRANT ANY PRIVILEGE (affecter n'importe quel privilège système et non objet)

PROCEDURE	
CREATE PROCEDURE	Création des procédures et fonctions stockées et gérer les packages dans son propre schéma.
CREATE ANY PROCEDURE	Création des procédures et fonctions stockées et gérer les packages dans n'importe quel schéma (nécessite aussi les privilèges ALTER ANY TABLE, BACKUP ANY TABLE, DROP ANY TABLE, SELECT ANY TABLE, INSERT ANY TABLE, UPDATE ANY TABLE, DELETE ANY TABLE, or GRANT ANY TABLE)
ALTER ANY PROCEDURE	Recompile n'importe quelle procédure ou fonction et modifie n'importe quel package dans n'importe quel schéma.
DROP ANY PROCEDURE	supprime n'importe quelle procédure ou fonction et supprime n'importe quel package dans n'importe quel schéma
EXECUTE ANY PROCEDURE	Exécute n'importe quelle procedure ou fonction (stand-alone ou packaged), or reference any public package variable in any schema.
ROLE	
CREATE ROLE	Création d'un rôle
ALTER ANY ROLE	Modification de n'importe quel rôle dans une base de données
DROP ANY ROLE	Suppression de n'importe quel rôle dans une base de données
GRANT ANY ROLE	Attribution de n'importe quel rôle dans la base de données

SESSION	
CREATE SESSION	Connexion à la base de données
ALTER SESSION	Modification état de connexion
RESTRICTED SESSION	Connexion quand la base est ouverte en mode STARTUP RESTRICT
SYNONYME	
CREATE SYNONYM	Création des synonymes dans son propre schéma.
CREATE ANY SYNONYM	Création des synonymes dans n'importe quel schéma
DROP ANY SYNONYM	Supprimer les synonymes dans n'importe quel schéma
TABLE	
CREATE TABLE	Création des tables dans son propre schéma. Permet aussi d'ajouter le privilège d'ajout des index sur les tables crées. l'attribution doit indiquer une valeur limite des espaces disques logiques sinon le privilège UNLIMITED TABLESAPACE doit être attribué.
CREATE ANY TABLE	Création des tables dans n'importe quel schéma.
BACKUP ANY TABLE	Exécute une opération d'export incrémental sur les tables d'un schéma

NB : il y a d'autres exemples dans le cours

Privilèges objet :

Permettent à un utilisateur d'effectuer une action particulière (manipulation) sur le contenu d'un objet spécifique (Ex : table, vue, séquence, procédure… CRUD : SELECT, INSERT, UPDATE, DELETE).

Peuvent être accordés par le propriétaire d'un objet, par l'administrateur, ou par un intermédiaire auquel la permission d'accorder des privilèges sur l'objet a été attribuée par le propriétaire de l'objet.

Privilège Objet	SQL
ALTER	ALTER object (table ou sequence)
DELETE	DELETE FROM object (table ou vue)
EXECUTE	EXECUTE object (procedure ou fonction). Références to public package variables
INDEX	CREATE Index ON object (Table)
INSERT	INSERT on object (Table ou vue)
REFERENCES	CREATE or ALTER TABLE <<définition Clé étrangère>> on Object (table) REFERENCES…..
SELECT	SELECT ….. FROM ….. (vue, table, sequence)
UPDATE	UPDATE object (table ou vue)

EXECUTE pour fonctions/procédures.

Options d'administration :

On peut donner le privilège à un utilisateur pour déléguer l'administration des privilèges à un autre utilisateur.

- Déléguer un privilège système avec WITH ADMIN OPTION
  - exemple :
    - GRANT CREATE SESSION TO HR WITH ADMIN OPION ;
    - connect hr/hr
    - GRANT CREATE SESSION TO SCOTT ;

NB : un REVOKE du privilège système CREATE SESSION pour l'utilisateur HR ne sera pas transmis à SCOTT. SCOTT gardera le privilège CREATE SESSION.

- Déléguer un privilège objet avec WITH GRANT OPTION
  - exemple :
    - GRANT SELECT ON EMPLOYEES to STOCK WITH GRANT OPTION ;
    - connect STOCK/STOCK
    - GRANT SELECT ON EMPLOYEES to USER1 WITH GRANT OPTION ;

NB : un REVOKE du privilège objet SELECT pour l'utilisateur STOCK sera transmis au utilisateur USER1. USER1 n'aura plus le privilège SELECT ON EMPLOYEES.

Rôles :

Avantages des rôles :

Simplifier la gestion des privilèges -> regrouper un ensemble de privilèges : Accorder des privilèges à un rôle, puis accorder ce rôle à chaque utilisateur.

Ex : Affecter à 5 utilisateurs les mêmes privilèges : au lieu d'affecter à chaqu'un ces privilèges -> regrouper ses privilèges dans un seul rôle (objet) et affecter ce rôle à ces 5 utilisateurs

Gestion dynamique des privilèges : modification des privilèges d'un rôle = tous les utilisateurs auxquels ce rôle est accordé bénéficient automatiquement et immédiatement des privilèges modifies.

Disponibilité sélective des privilèges : retirer temporairement les privilèges -> désactiver le rôle et le réactiver lorsque je veux redonner ces privilèges. => manipuler tout un rôle au lieu de manipuler privilèges par privilèges.

Rôles prédéfinis :

Lorsque je donne un rôle, je lui affecte les privilèges prédéfinis de ce rôle (Ex : affecter le rôle CONNECT -> affecter privilèges CREATE SESSIONS,CREATE TABLE,….

Rôle	Privilèges
CONNECT	CREATE SESSION, CREATE TABLE, CREATE VIEW, CREATE SYNONYM, CREATE SEQUENCE, CREATE DATABASE LINK, CREATE CLUSTER, ALTER SESSION
RESOURCE	CREATE TABLE, CREATE PROCEDURE, CREATE SEQUENCE, CREATE TRIGGER, CREATE TYPE, CREATE CLUSTER, CREATE INDEXTYPE, CREATE OPERATOR
SCHEDULER_ADMIN	CREATE ANY JOB, CREATE JOB, EXECUTE ANY CLASS, EXECUTE ANY PROGRAM, MANAGE SCHEDULER
DBA	La plupart des privilèges système et plusieurs autres rôles. Ce rôle ne doit pas être accordé aux utilisateurs qui ne sont pas administrateurs.
SELECT_CATALOG_ROLE	Pas de privilèges système, mais plus de 1600 privilèges objet sur le dictionnaire de données.

Créer un rôle :

Création d'un rôle qu'on va lui nommé SUPERDBA avec mot de passe et intégrant le rôle DBA et des privilèges système :

CREATE ROLE"SUPERDBA"IDENTIFIEDBY"sup"; le mdp sera demandé à l'activation du rôle -> n'est pas obligatoire.

```
GRANT ALTER ANY INDEXTYPE TO "SUPERDBA"
GRANT ALTER ANY PROCEDURE TO "SUPERDBA"
GRANT ALTER ANY TABLE TO "SUPERDBA"
```

ANY : tous les chemins.
Une fois ce rôle a été créé on lui affecte les privilèges en utilisant GRANT

GRANT "DBA"TO "SUPERDBA"WITH ADMIN OPTION ; => on a affecté le rôle prédéfini DBA au rôle qu'on a crée SUPERDBA -> SUPERDBA peut affecter les privilèges système aux autres utilisateurs

Affectation du rôle SUPERDBA à l'utilisateur STOCK : **GRANT "SUPERDBA" TO STOCK WITH ADMIN OPTION ;**

Profils :

Un seul profil est affecté à un utilisateur à un instant donné (un utilisateur ne peut avoir qu'un seul profil très bien) : Ce sont les ensembles de limites qu'un utilisateur ne doit pas les dépasser ->

Limites % mots de passe (option toujours activée)

FAILED_LOGIN_ATTEMPTS	Nombre d'échecs de connexion avant le verrouillage du compte
PASSWORD_LOCK_TIME	Nombre de jours pendant lesquels le compte est verrouillé après le nombre déterminé d'échecs de connexion

FAILED_LOGIN_ATTEMPTS : Si atteint -> compte sera verrouillé (Ex : FAILED_LOGIN_ATTEMPTS = 3 : je tape mdp erroné 3 fois -> compte verrouillé et **ne peut être déverrouillé qu'avec l'administrateur**)

PASSWORD_LOCK_TIME : même l'administrateur ne peut pas déverrouiller le compte pendant ce temps-là (EX : PASSWORD_LOCK_TIME = 3 -> compte verrouillé pendant 3 jours)

PASSWORD_LIFE_TIME	Durée de vie du mot de passe, en jours, avant expiration
PASSWORD_GRACE_TIME	Période de grâce, en jours, permettant le changement de mot de passe après la première connexion réussie suite à l'expiration du mot de passe

Une fois **PASSWORD_LIFE_TIME** fini -> l'utilisateur doit changer son mot de passe sinon compte expiré.

Ex : PASSWORD_LIFE_TIME = 4 & PASSWORD_GRACE_TIME = 2 -> après 4 jour mon mot de passe expiré et j'aurais 2 jours pour changer mon mot de passe

PASSWORD_REUSE_TIME	Nombre de jours pendant lesquels le mot de passe ne peut pas être réutilisé
PASSWORD_REUSE_MAX	Nombre de fois le mot de passe actuel peut être utilisé

PASSWORD_REUSE_TIME : nombre de jours de délai entre 2 utilisations du même mot de passe (EX : PASSWORD_REUSE_TIME = 2 -> je ne peux changer un mdp qu'après 2 jours / PASSWORD_REUSE_MAX = 2 -> je peux redéfinir ce mot de passe que 2 fois)

PASSWORD_VERIFY_FUNCTION	Fonction PL/SQL qui effectue une vérification de complexité avant l'affectation d'un mot de passe

Fonction stocké qui doit être créé au préalable et qui permet de définir critères personnalisés (longueur mdp, mdp contient lettres…) et affectées aux variables **PASSWORD_VERIFY_FUNCTION**

Limites % la consommation des ressources (taille en cpu, temps de connexion, nombre de sessions ouvertes, etc) : option activée si la valeur de paramètres d'initialisation RESOURCE LIMIT est à TRUE.

Sa valeur par défaut est FALSE.

SESSIONS_PER_USER	Nombre de sessions maximal qu'un utilisateur peut l'ouvrir simultanément -> SESSIONS_PER_USER= 2 : je peux ouvrir que 2 instances de fenêtres SQL avec même user.
CPU_PER_SESSION	Définir temps de processeur maximum (en centisecondes) qu'une session peut l'utiliser
CPU_PER_CALL	Si je lance une requête et la dépassé le CPU_PER_CALL(en CentiSeconde) elle va être stoppée automatiquement
CONNECT_TIME	Temps en secondes de la durée maximale de la connexion pour une session EX : CONNECT_TIME = 30*60 : si je connecte à 21h -> 21h30 il me déconnecte automatiquement
IDLE_TIME	Temps en secondes de la durée d'inactivité maximale (Ex : IDLE_TIME = 30*60 : si dernière requête a été lancé à 21h et je ne lance pas une requête après 30 min - session stoppé.

Création d'un profil :

```
CREATE PROFILE <profile_name> LIMIT
[Limit_ressourse> value]
[Limit_passowrd> value] ;
value :={integer | UNLIMITED | DEFAULT} ;
value :={function | null | DEFAULT} pour le paramètre PASSWORD_VERIFY_FUNCTION
```

On peut utiliser autant d'options qu'on veut

Pour avoir chiffre en jour en dévise par 1440

Vérifiez l'attribution du profil en utilisant les vues du dictionnaire de données : **select profile from dba_users where username='TP3';**

Exemple

```
CREATE PROFILE developer_profile LIMIT
SESSIONS_PER_USER 2
CPU_PER_SESSION 10000
CONNECT_TIME 480
IDLE_TIME 60
FAILED_LOGIN_ATTEMPTS 3
PASSWORD_LIFE_TIME 90
PASSWORD_REUSE_TIME 2
PASSWORD_VERIFY_FUNCTION my_function ;
```

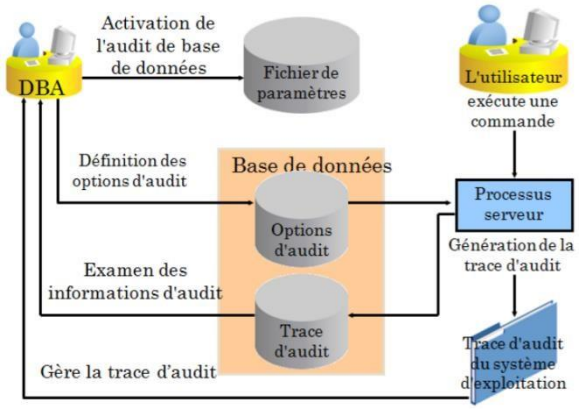
Sécurité et audit des bases de données oracle

Audit de la Base de Données :

Audit = surveillance / contrôler.
Les outils d'audit intégrés d'Oracle sont les suivants :

- Oracle Standard Auditing appelé Audit de la BD.
- Auditing par Trigger appelé aussi Audit basé sur les valeurs ou sur les données.
- Fine Grained Auditing n'est pas disponible sur toutes les versions (audit détaillé).

Audit standard de la BD :



Deux moyens pour activer un audit :

- Activer pendant l'instance en cours
- Activer en façon permanente -> écrire cette activation au niveau de fichier de paramètres spfile.

L'audit se fait sur l'utilisation des privilèges, notamment l'accès aux objets. Il permet d'identifier les commandes exécutées sur une table, mais ne peut pas fournir des informations sur les modifications faites.

Les preuves d'audit sont un ensemble fixe de données.

L'audit est activé via le paramètre AUDIT TRAIL.

NONE : désactive les enregistrements.

OS : active l'audit, les enregistrements sont dans un fichier défini par **AUDIT_FILE_DEST**.

DB : active l'audit,les enregistrements sont dans la table BD défini par **SYS.aud$**. Pour changer la valeur du paramètre AUDIT_TRAIL :

- Si c'est un pfile, alors il faut d'éditer le contenu du fichier.
- Si c'est un spfile, alors il faut exécuter **ALTER system set audit trail=DB scope=spfile ;**

L'audit via l'**OS** est préféré lorsqu'on veut auditer plusieurs BDs et dans une même destination.

L'audit via la BD permet de visualiser les résultats par des requêtes simples contre les vues du dictionnaire relatives à l'audit. Il faut aussi auditer les actions de la table SYS.AUD$ par **audit all on SYS.AUD$ by access**.

Niveaux d'audit :

Oracle permet un Auditing standard sur 4 niveaux :

- Auditing de commandes (LDD).
- Audit de privilèges (systèmes).
- Audit d'objets de schéma (privilèges objets).
- Audit de connexion à la BD.

BY SESSION : n'insérer par session qu'un enregistrement par objet de BD.

BY ACCESS : insérer un enregistrement dans la trace à chaque action soumise.

WHENEVER : les audits ne doivent être exécutés que lorsque l'exécution de commandes SQL est terminée, réussie ou non.

Audit de privilèges ou de commandes

```
AUDIT {commande | privilège_système}
      [ , {commande | privilège_système} ] ...
      [BY user [ , user ] ... ]
      [BY {SESSION | ACCESS} ]
      [WHENEVER [NOT] SUCCESSFUL]
```

Audit d'objets

```
AUDIT commande [ , commande ] ...
```

Vues d'audit :

DBA_AUDIT_TRAIL : Toutes les entrées de la trace d'audit.

DBA_AUDIT_OBJECT : Enregistrements des objets de schémas.

DBA_AUDIT_SESSION : Toutes les entrées de connexion et de déconnexion.

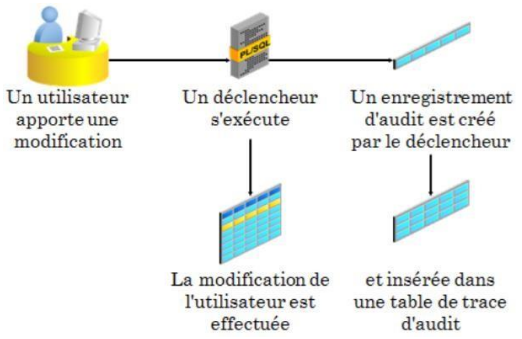
DBA_AUDIT_STATEMENT : Enregistrements d'audit des instructions.

Audit basé sur les données (avec les triggers) :

L'audit se fait sur les données modifiées par les instructions LMD. Les preuves d'audit sont définies par l'administrateur.

DML trigger : permet d'enregistrer les valeurs de toutes les modifications de la BD. (Tâche impossible pour l'audit intégré d'Oracle).

System trigger : auditing de toute action de création, de suppression ou de connexion à la BD. (Inutile puisqu'elles seront auditées en utilisant l'audit intégré d'Oracle).




```
CREATE [OR REPLACE] TRIGGER nom_trigger
{BEFORE|AFTER|INSTEAD OF}
{INSERT|UPDATE [ OF nom_colonne [, nom_colonneN]] |DELETE}
OR {INSERT|UPDATE [OF nom_colonne [, nom_colonneN ]] |DELETE}
REFERENCING {[OLD [AS] ancien] | [NEW [AS] nouveau]}
ON nom_table
[FOR EACH ROW]
[WHEN] (condition)
DECLARE
/* déclaration */
BEGIN
/* traitement */
[EXCEPTION]
END ;
```

Audit détaillé « Fine Grained Auditing » (FGA) :

L'audit se fait sur les instructions SQL. Les preuves d'audit sont un ensemble fixe de données incluant l'instruction SQL. Il permet de définir des conditions fines pour l'auditing :

- Surveille l'accès aux données en fonction du contenu.
- Peut être lié à une table ou à une vue.
- Peut exécuter une procédure PL/SQL.
- Est administré via le package DBMS FGA.

Une stratégie d'audit détaillé est constituée de 2 parties :

- Les critères d'audit : définit la condition à vérifier pour que l'instruction soit auditée.
- L'action d'audit : le nom d'une procédure à exécuter lorsque la condition est vérifiée. Elle est créée via la procédure **ADD_POLICY** du package **DBMS_FGA**.

DBMS FGA inclut les sous-programmes suivants :

- **ADD_POLICY** : crée une stratégie d'audit à l'aide du prédicat fourni en tant que condition d'audit.

- **DROP_POLICY** : supprime une stratégie d'audit.

- **ENABLE_POLICY** : active une stratégie d'audit.
- **DISABLE_POLICY** : désactive une stratégie d'audit. Vues d'audit :

DBA_FGA_AUDIT_TRAIL : Tous les événements d'audit détaillé.

ALL_AUDIT_POLICIES : Toutes les stratégies d'audit détaillé pour les objets auxquels l'utilisateur actuel peut accéder.

DBA_AUDIT_POLICIES : Toutes les stratégies d'audit détaillé dans la BD.

USER_AUDIT_POLICIES : Toutes les stratégies d'audit détaillé pour les objets du schéma de l'utilisateur actuel.

Règles d'audit détaillé :

- Pour auditer toutes les instructions, il faut utiliser une condition NULL.
- Une erreur est générée lors de l'ajout d'une stratégie existante.
- La table ou la vue auditée doit déjà exister lorsqu'il y a création de la stratégie.
- Si la colonne d'audit n'existe pas dans la table, aucune ligne n'est auditée.

Déplacement des données

Il existe plusieurs méthodes pour charger des données dans les tables d'une BD Oracle :

- Utilitaire d'export / import ORACLE DATA PUMP.
- SQL*Loader.

Oracle Data Pump :

C'est un utilitaire (côté serveur) de déplacement de données et de métadonnées de masse de manière très rapide entre des BD Oracle (deux utilitaires d'import et d'export des données).

Pour lancer Data Pump, il faut créer un Directory Oracle où il y aura stockage des exports. Data Pump peut être appelé via :

- Enterprise Manager (EM) Database Control.
- Les binaires EXPDP et IMPDP situés dans le dossier bin d'ORACLE HOME.
- Le package SYS.DBMS DATAPUMP.

Il existe 4 modes d'export/import avec Data Pump :

- Export / Import COMPLET : opération demandée par le paramètre FULL.
- Export / Import de SCHEMA (metadata) : permet l'imp/exp d'un ou plusieurs schémas de la BD (mode par défaut).
- Export / Import de TABLE : mode commandé par le paramètre TABLES qui permet de sélectionner des tables à exporter à partir d'un schéma.
- Export / Import de TABLESPACE : permet d'exporter les tables d'au moins un espace disque logique (Seules les tables seront exportées, et non les espaces disque logiques).

3 types de fichiers sont générés avec Data Pump :

Fichiers SQL : contiennent les instructions LDD de la création des objets de la BD. Fichiers DUMP : contiennent les données exportées.

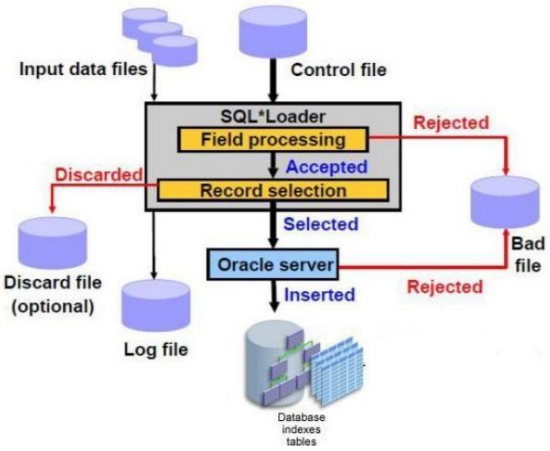
Fichiers LOG : contiennent le journal d'historique de l'exécution du job (export ou import).

EXPDP/ IMPDP :

Privilèges nécessaires :

- EXP FULL DATABASE et IMP FULL DATABASE : permet d'exporter l'intégralité d'une BD, un Tablespace, un schéma autre que le sien ou une table située dans un autre schéma.
- Tous les utilisateurs qui possèdent le rôle DBA ont la possibilité d'effectuer un import/ export d'une BD.

SQL*Loader :



C'est un utilitaire qui charge les données de fichiers externes dans des tables d'une BD Oracle.

Il dispose d'un puissant moteur d'analyse (parse) des données, qui ne limite que très peu le format des données du fichier.

Log File :

C'est un fichier qui enregistre les activités SQL Loader durant un chargement de données :

- Les noms des fichiers CONTROL FILE, BAD FILE, DISCARD FILE, Input Data File.
- Les champs et types de données qui ont été chargées.
- Messages d'erreurs sur les enregistrements non chargés.
- Le nombre d'enregistrements lus dans le fichier de données ou rejetés en raison d'erreurs ou de critères de sélection.
- Le temps de chargement. Bad File :

SQL Loader enregistre les erreurs (de lecture ou de chargement d'un enregistrement) dans un fichier BAD FILE : enregistrement non conforme au format dans le fichier de contrôle, violations de contraintes d'intégrité, tablespace plein, etc.

Discard File :

Les enregistrements qui ne répondent pas aux conditions de sélection spécifiées dans le Control File sont rejetés et écrits dans le fichier DISCARD FILE.

Control File :

Le fichier de contrôle indique à SQL*Loader :

- L'emplacement des fichiers Bad File, Discard File et Log File.
- Les détails de configuration : Gestion de la mémoire Rejet des enregistrements.
- L'emplacement, la structure, types, longueurs, précisions des données à charger.
- Les noms de tables à charger.
- La correspondance entre les champs des données et les colonnes des tables de la BD.
- Les critères de sélection des enregistrements à insérer dans les tables de la BD.
- Formatage des enregistrements de données : si le format est délimité, etc.

Exemple :

- Soit le fichier « Emp.dat » contenant :
  10001,"Scott Tiger", 1000, 40
  10002,"Frank Naude", 500, 20
- Pour charger le fichier emp.dat dans la table « emp » du schéma HR, le control file doit contenir les informations suivantes :

```
load data
infile 'c :\Emp.dat'
into table emp     – { INSERT | REPLACE | TRUNCATE | APPEND }
fields terminated by "," optionally enclosed by '"'
( empno, empname, sal, deptno )
```

Pour charger les données on doit exécuter la commande suivante :
SQL> host
C :\> Sqlldr user/password control=<control_file>.ctl
log=<log_file>.log bad=<bad_file>.bad discard=<discard_file>.dsc