Chapitre III
Administration du Système de Gestion de Base de Données Oracle

III.1. Architecture physique

III.1.1/ **Schéma général** : Une base de données Oracle est constituée de plusieurs éléments :

- Des processus chargés en mémoire sur le serveur
- Des fichiers physiques stockés sur le serveur
- D'un espace mémoire sur le serveur appelé *SGA* (*System Global Area*)

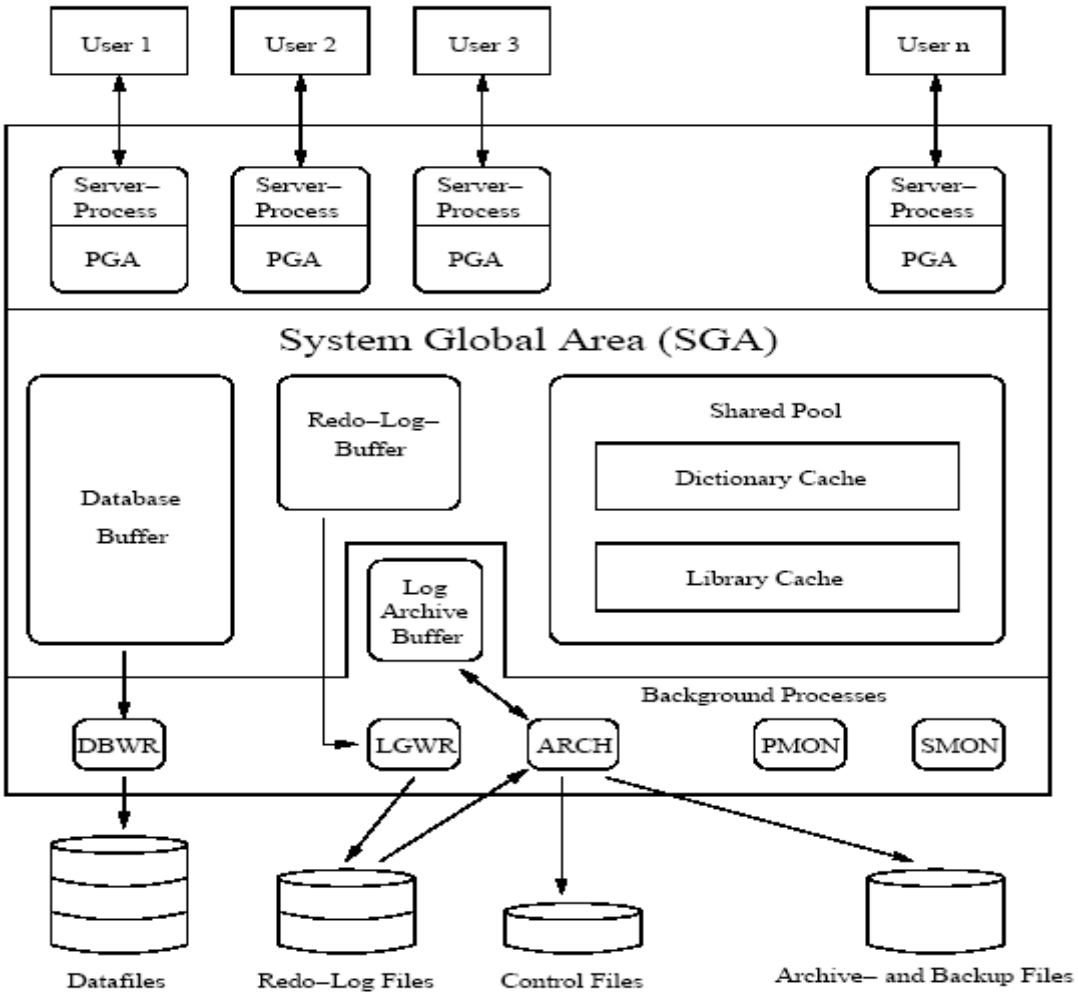


Figure 1. Architecture générale d'une instance Oracle

On appelle ***instance*** *Oracle* les **processus** et la **SGA** d'une base de données Oracle.

III.1.2/ Les fichiers physiques d'une base Oracle

Les fichiers physiques d'une base Oracle permettent de stocker de manière persistante les données manipulées par Oracle. On distingue deux types de fichiers :

- Les fichiers servant à stocker les informations de la base. Tous ces fichiers sont des fichiers binaires, ce qui signifie qu'ils sont inexploitables avec un éditeur de texte.
- Les fichiers destinés à la configuration et au fonctionnement de la base Oracle

Les fichiers d'une base de données Oracle sont les suivants :

- **Les fichiers de données**
- **Les fichiers Redo Log**
- **Les fichiers de contrôle**

Une **base de données Oracle** nécessite au minimum :
- un fichier de données **(**dont l'extension est .dbf**),**
- deux fichiers redo Log **(**dont l'extension est .rdo ou .log**),**
- et un fichier de contrôle **(**dont l'extension est .ctl**).**

III.1. 3/ **Les fichiers de données**

Ces fichiers contiennent l'ensemble des données de la base (les tables, les vues, les procédures stockées, ...). Il sont codés dans un format propriétaire. Seule les requêtes SQL permettant un accès implicite à ces fichiers.

Les fichiers de données contiennent des informations de deux types :

- Le **dictionnaire** de données et de travail
- Les **données** des utilisateurs

La **lecture** de ces fichiers de données est faire à l'aide des **processus utilisateurs** tandis que l'**écriture** est assuré par le **processus DBWR** (*Database Writer*).

III.1.4/ **Les fichiers Redo-log**

Les fichiers Redo-log contiennent l'**historique** des modifications apportées à la base de données Oracle. Ces fichiers de **journalisation enregistrent** les modifications successives de la base de données afin de pouvoir restaurer la base de données en cas de défaillance d'un disque dur. Ainsi le cas échéant, la base de données Oracle est à même de simuler l'ensemble des commandes n'ayant pas été sauvegardées pour rétablir le contenu de la base de données.

Au même titre que les fichiers de données, les fichiers Redo-log sont dans un format **propriétaire** Oracle et l'écriture dans ces fichiers est assurée par le **processus LGWR** (*Log Writer*).

Oracle propose également un **mode archivage** permettant la sauvegarde du fichier Redo-log avant sa réutilisation pour restaurer la base. Si ce mode n'a pas été activé, le contenu du fichier Redo Log est **supprimé après utilisation**.

Enfin ces fichiers peuvent être **multiplexés** afin de fournir un maximum de sécurité.

III.1.5/ Les fichiers de contrôle

Ces fichiers permettent de stocker les informations sur l'état de la base de données (emplacement des fichiers, dates de création, ...). Ils sont créés lors de la création de la base.

Ces fichiers permettent, lors de l'initialisation de la base, de savoir si la base de données a été arrêtée correctement, ainsi que de connaître l'emplacement des fichiers de données et des fichiers Redo Log. Les fichiers de contrôle sont eux-même repérés par le **fichier d'initialisation**.

Le fichier de contrôle contient les informations suivantes :

- Nom de la base de données
- Date et heure de création de la base
- L'emplacement des fichiers journaux (Redo-Log)

III.1.6/ Le fichier d'initialisation

Ce fichier est un fichier au **format texte** contenant l'ensemble des paramètres de démarrage de la base (il est généralement nommé **`init`**`SID`**`.ora`**, où `SID` représente le nom donné à l'instance). Son existence n'est toutefois pas majeure car il peut être facilement **reconstruit**.

Un fichier d'initialisation par défaut est créé lors de la **création d'une base**. Celui-ci est largement documenté et des exemples de valeurs sont donnés pour chaque paramètre. Toutefois parmi ces paramètres, **seul un nombre limité d'entre-eux est réellement utile**.

Exemple (`init.ora`) :

```
db_name = DB01
db_files = 20
control_files = /home/oracle/ORADBA/DB01/DATABASE/ctl1db01.ora
db_block_buffers = 200
db_block_size = 2048
```

III.1.7/ Les Autres fichiers de configuration

Le fichier listener.ora (déjà vu en chapitre 1) qui décrit les différentes bases accessibles sur le serveur courant. Il est généralement dans `ORACLE_HOME/network/admin`

Exemple

```
LISTENER =
  (DESCRIPTION_LIST =
    (DESCRIPTION =
      (ADDRESS_LIST =
        (ADDRESS = (PROTOCOL = IPC)(KEY = EXTPROC0))
      )
      (ADDRESS_LIST =
        (ADDRESS = (PROTOCOL = TCP)(HOST = LIH-NAKECH)(PORT = 1521))
      )
    )
  )
SID_LIST_LISTENER =
  (SID_LIST =
    (SID_DESC =
      (SID_NAME = PLSExtProc)
      (ORACLE_HOME = /oracle/ora92)
      (PROGRAM = extproc)
    )
    (SID_DESC =
      (GLOBAL_DBNAME = ora1)
      (ORACLE_HOME = /oracle/ora92)
      (SID_NAME = ora1)
    )
  )
```

Nom de la machine serveur

Nom de la base serveur

Le fichier TNSNAMES.ORA (déjà vu en chapitre 1) qui decrit les diferentes BD accéssibles (résolution de noms). Comme le fichier précédent, **TNSNAMES.ORA** est dans `ORACLE_HOME/network/admin`

Exemple

```
PP =
  (DESCRIPTION =
    (ADDRESS_LIST =
      (ADDRESS = (PROTOCOL = TCP)(HOST = 192.168.0.86)(PORT = 1521))
    )
    (CONNECT_DATA =
      (SERVICE_NAME = ora1)
    )
  )
```

Ip du serveur à distace

Nom d'accès à distance

Nom de la base distante

III.1.8/ Les principaux processus

On distingue généralement deux types de processus :

- **les processus utilisateurs** : Un processus utilisateur est créé pour chaque programme exécuté par un utilisateur (par exemple Oracle Forms ou Server Manager) afin de fournir l'environnement nécessaire à l'exécution de celui-ci.

- **les processus systèmes** : Les 4 principaux processus systèmes sont :

Le processus DBWR
Le processus Database Writer (DBWR) a pour but de transférer les blocs de données modifiés (appelés *dirty blocks*) de la *System Global Area* vers les fichiers de la base de données, afin de sauvegarder de manière permanente les données de la base. Ainsi, lorsqu'un ordre SQL modifie la base de données (c'est-à-dire lorsqu'une requête SQL *DELETE, INSERT ou UPDATE* est reçue), les blocs de données affectés sont modifiés dans le fichier de données associé.

Le processus LGWR
Le rôle du processus LGWR (*Log Writer*) est de mettre à jour les fichiers journaux (Redo Log) dans la SGA et sur le disque. Ainsi ce processus est chargé d'écrire le contenu du cache Redo Log de la SGA dans le fichier Redo Log à chaque fois qu'un ordre COMMIT est réceptionné.

Le processus SMON
Le processus SMON (*System Monitor*) est chargé de vérifier la cohérence du système et de la rétablir suite à un incident au démarrage de la base suivant. Ainsi, si la base n'a pas été stoppée correctement, le processus analyse les informations stockées dans les rollback segments (les rollback segments sont les zones de stockage des opérations n'ayant pas encore été validées) puis annule toutes les informations en attente mais pour lesquelles aucune validation n'a été enregistrées (appelées *deadlocks*).

Ainsi SMON a un rôle de libération des ressources utilisées inutilement par le système.

D'autre part SMON surveille les espaces libres des fichiers de la base de données et les réorganise si nécessaire afin de les défragmenter.

Le processus PMON
Le processus PMON (*Process Monitor*) a pour but de récupérer les ressources associées à des défaillances de processus utilisateurs. Ainsi il supprime les processus en erreur, il annule les transactions n'ayant pas été validées (par exemple si un client est déconnecté brutalement lors de la transaction); il libère les verrous, et libère les ressources utilisées inutilement dans la SGA.

III.1.9/ **L'utilisation de la mémoire par Oracle.**

Oracle fait un usage poussé de la mémoire physique (RAM) du serveur afin de fournir les meilleures performances possibles :

- accélérer l'accès aux données de la base régulièrement accédées
- mettre les processus en mémoire
- optimiser la communication entre les processus et la base de données

Structure de la mémoire utilisée par ORACLE

- **La zone SGA** (*System Global Area*) assurant le partage des données des différents utilisateurs, c'est-à-dire qu'il s'agit de la zone contenant les structures de données

accessibles par tous les processus.

- **La zone PGA** (*Program Global Area*) permettant le fonctionnement des divers processus (afin de stocker toutes les données ne nécessitant pas d'être partagées).

La SGA (appelée aussi *mémoire réservée*) est composée de plusieurs constituants :

- La *Shared Pool* (ou *zone partagée*) contenant des informations pouvant être réutilisées par les différents utilisateurs, c'est-à-dire les requêtes SQL, les curseurs, ... D'une manière générale, cette zone sert à mémoriser et traiter les requêtes SQL provenant des divers utilisateurs.

- Le Db block buffer (Database Buffer Cache ou cache des blocs de données) est un espace mémoire contenant toutes les données transitant de ou vers la base de données : blocs de données, blocs d'index et blocs contenant les ROLLBACK SEGMENTS.

- Le Redo Log buffer (ou cache Redo-log) contient les blocs de données (appelés *Redo Entries*) à modifier et les modifications à effectuer sur ces données, avant que l'ensemble de ces mises à jour de la base ne soient archivées dans les fichiers Redo-log

L'ensemble des tailles des caches peut être modifié (augmentée ou diminuée) grâce aux paramètres du fichier d'initialisation (**`init.ora`**) dont un exemple est déjà vu plus haut (paragraphe1.6).

III.2 Structure d’une base de données oracle

- Structure logique : Accessible à l’utilisateur

- Structure physique : Paramétrable par des fichiers de configuration

III.2.1 Structure logique d'une base de données oracle

Il existe plusieurs niveaux de structures logiques (**accessibles à l’utilisateur par requête SQL**) allant du **schema object** (la structure la plus importante) au **datablock** (la plus petite structure).

Les schema objects

Par Schema objet on entend un moyen d'accès à la BD. On y trouve notamment les tables, mais aussi les vues, les index, les clusters, les liens, les synonymes, les procédures PL/SQL et les packages PL/SQL.

Reprenons un à un ces schema objects.

Les tables : on ne reviendra pas sur ce concept. Les tables sont des Schema Objects. En effet elles permettent directement d'accéder aux données .

Les vues : ces éléments qui permettent de donner accès à un sous-ensemble d'une table (par exemple, car on peut créer des vues sur n'importe quel schema object : pourquoi pas une vue sur une vue) ou de plusieurs tables (jointes), sont des schema objects.

Nous verrons plus tard que ces vues peuvent être utilisées pour cloisonner le champ d'action d'un utilisateur (au lieu de lui donner accès à une table complète, on ne lui concède que le sous-ensemble requis).

Les index : ces éléments sont donc aussi des schema objects. En quelques mots, on peut dire qu'un index, similairement à l'index d'un ouvrage, permet à une instance du serveur d'accéder plus rapidement à des éléments. Nous reparlerons de cela plus en avant dans ce cours.

Les clusters : ces schema objects permettent aussi un accès plus rapide aux données. L'astuce consiste à supprimer des données doubles et donc à avoir moins de données à charger à partir des disques.

Les liens : ces schema objects permettent d'accéder des données sur une DB distante.

Les synonymes : ils consistent en un nom de remplacement sur un autre schema object.

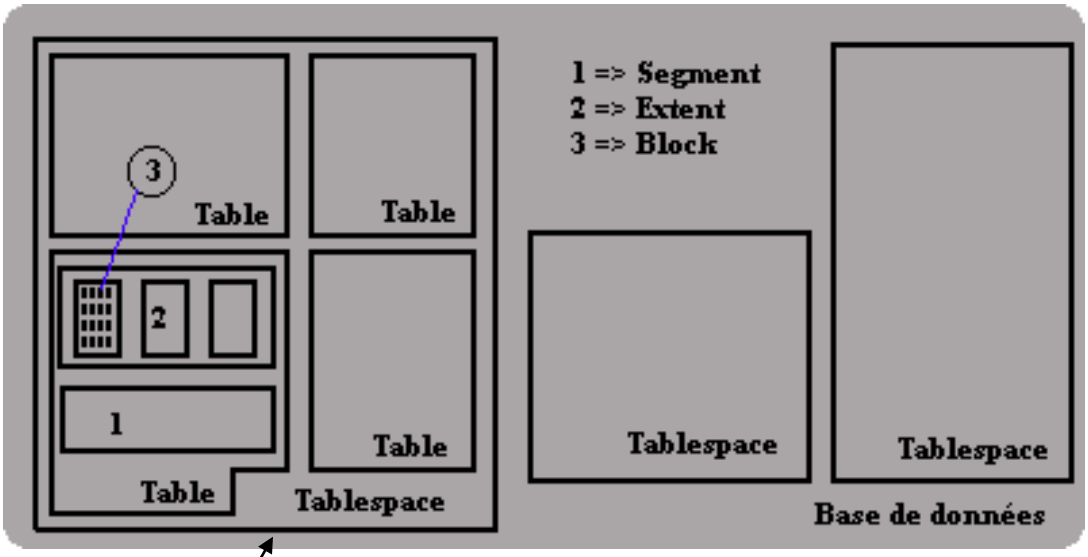
Les procédures et les packages : une procédure est un ensemble d'ordres PL/SQL permettant de réaliser une action sur des données.

Un package est un ensemble de procédures. Pour qu'elles puissent être utilisées, ces unités de stockage ont besoin d'être stockées sur la BD et comme elles permettent la manipulation des données, ce sont des schema objects.

L'ensemble de tous les schema objects pour un utilisateur est appelé **user's schema**.

III.2.2/ Structure physique d’une base de données oracle

Le schéma suivant replace, les unes en rapport aux autres, les diverses unités logiques existantes.



Les différentes structures logiques de la base de données.

Ce tablespace correspond à un ou plusieurs fichiers sur disque

III.2.3/ Les tablespaces

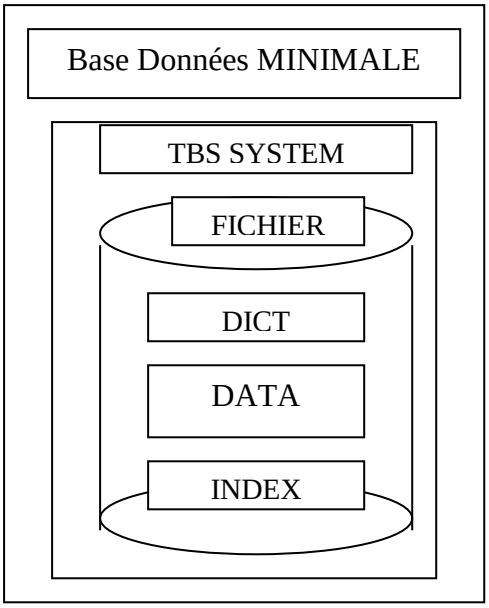
Notion de tablespaces

Cette unité logique rentre dans la constitution de la BD. En effet, une base peut être décomposée en tablespaces : partitions logiques contenant un ou plusieurs fichiers. Un fichier appartient à 1 et 1 seul tablespace.

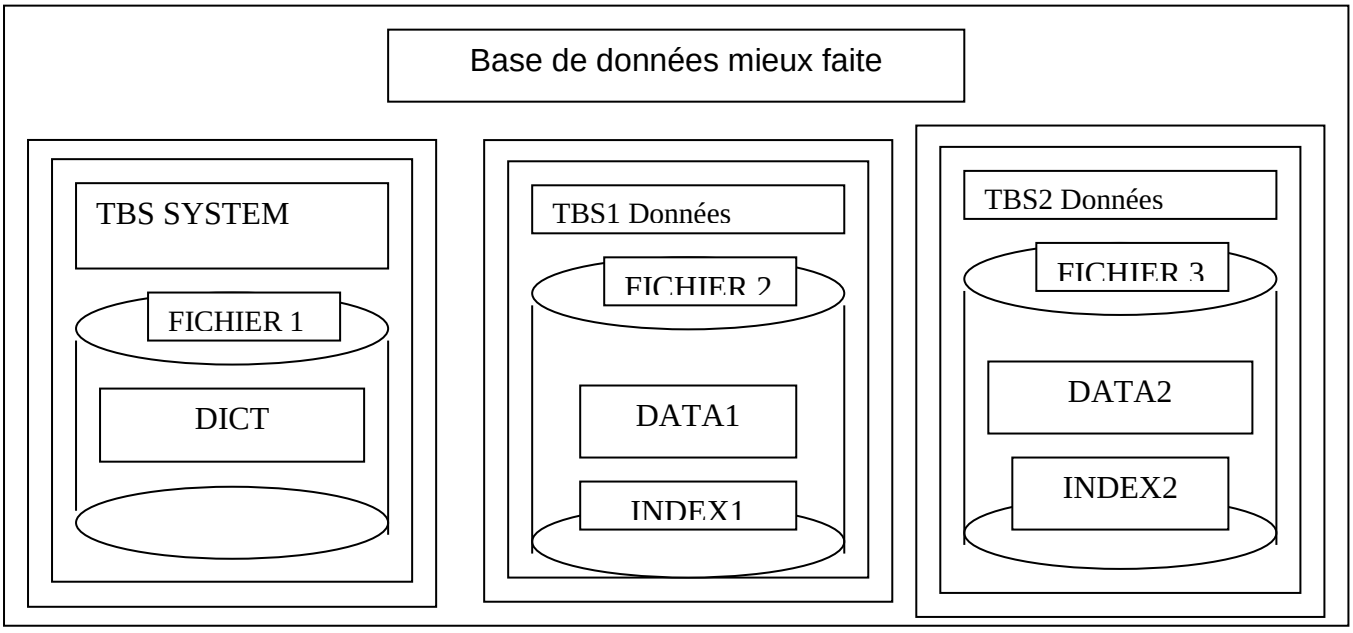
Un tablespace peut s'étendre soit par ajout (on-line) d'un fichier, soit par auto-extension DU fichier du tablespace.

Par défaut il existe toujours un tablespace baptisé SYSTEM qui contient le dictionnaire de données et le rollback segment SYSTEM.
Quand vous allez créer une nouvelle table, celle-ci sera contenue soit dans un **tablespace** existant (SYSTEM par défaut) soit dans un nouveau que vous créerez.

On peut également stocker les datas et les index dans ce même tablespace, et obtenir ainsi une base minimale peu structurée, peu performante et peu sécurisée :



Au contraire on peut répartir les données, les index, mais aussi les images avant (rollback segments) sur un nombre maximum de disques. On y gagnera en performance, en souplesse, et en sécurité :



Les Ordres SQL associés aux tablespaces :

```
CREATE TABLESPACE ...

DROP TABLESPACE...

ALTER TABLESPACE...
```

```
CREATE TABLE   emp  (empno NUMBER(5) PRIMARY KEY, ... )
TABLESPACE     tbs1
STORAGE
( INITIAL      50K
     NEXT      50K
     MAXEXTENTS     10
     PCTINCREASE    25) ;
```

ON LINE/.OFFLINE de un tablespace

par défaut un tablespace à la création est ON LINE (et donc accessible), il peut être mis OFFLINE (et les fichiers qu'il continet par conséquent) pour en interdire l'accès ou pour certaines opérations de maintenance

Description des tablespaces de la base courante dans les vues Dba_tablespaces et Dba_data_files du dictionnaire.

Les fichiers du tablespace

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

lors de la suppression du tablespace (DROP TABLESPACE...) les fichiers correspondant ne sont PAS SUPPRIMES par Oracle !! il faut le faire manuellement au niveau du système Unix ou Windows (rm, del...)

Quelques exemples SQL pour les tablespoaces et les fichiers

- creation d'un tablespace nommé RBS contenant un fic de 10MO et des EXTENTS de 1MO :

```
CREATE TABLESPACE RBS DATAFILE 'E:\orant\database\TEST\Rbs1TEST.ora'
SIZE 10M DEFAULT STORAGE ( INITIAL 1024K NEXT 1024K PCTINCREASE 0);
```

- Modification le tablespace toto, on ajout d'un ficheir auto exyensible jusqu'a 100 MO :

```
ALTER TABLESPACE toto OFFLINE;

ALTER TABLESPACE toto ADD DATAFILE 'E:\orant\database\TEST\TEST.ora'
SIZE 10M AUTOEXTEND ON NEXT 5M MAXSIZE 100M;
```

Extension du tablespace

Pour augmenter la taille d'un tablespace, il y a 2 solutions :

- Ajouter un fichier au tablespace, qui sera chainé au premier :
  `ALTER TABLESPACE toto ADD DATAFILE...`

- mettre le fichier du tablespace en AUTO extension
  ```
  ALTER DATABASE DATAFILE toto.dbf AUTOEXTEND ON)
  ```

Une table, peut "s'étaler" sur plusieurs fichiers. Ainsi le fait qu'une table sature un tablespace n'est pas bloquant il suffit d'augmenter la taille du tablespace.

Les différents types de tablespaces spéciaux

Tablespaces en lecture seule (READ ONLY tablespaces)

Ces tablespaces sont utilisés (on s'en serait douté) en lecture seule. Ils permettent de stocker des données statiques (ou variant très peu souvent, éventuellement sur des CDROMS, et ne rentrent pas en ligne de compte dans les sauvegardes / restaurations.

Pour modifier les données d'un Tablespace READ ONLY il est évidemment obligatoire de modifier préalablement son statut.

```
ALTER TABLESPACE toto READ ONLY;
ALTER TABLESPACE toto READ WRITE;
```

Tablespaces temporaires (temporary tablespaces)

On peut (et doit) créer un tablespace temporarire par défaut autre que SYSTEM, où seront stockées toutes les données temporaires (utilisées lors des tris, création d'index, jointures, etc). Ils sont définis lors de la création de la base :

```
CREATE DATABASE ma_base...
        DEFAULT TEMPORARY TABLESPACE mon_temp;
```

ou a posteriori :

```
ALTER DATABASE DEFAULT TEMPORARY TABLESPACE tempts2;
```

En plus de ce tablespace temporaire par défaut, chaque utilisateur peut se voir assigner un tablespace temporaire particulier

```
CREATE TEMPORARY TABLESPACE mon_temp TEMPFILE
'/oracle/data/temp01.dbf' SIZE 20M
EXTENT MANAGEMENT LOCAL UNIFORM SIZE 10M

CREATE USER toto IDENTIFIED BY tutu
DEFAULT TABLESPACE data
QUOTA 100M ON data
TEMPORARY TABLESPACE temp_ts
```

III.3. Le data dictionary (ou les tables système)

III.3.1/ Présentation générale

Le dictionnaire de données Oracle représente le coeur de la base de données. Il s'agit d'un ensemble de tables systèmes contenant les informations relatives à la structure de la base de données :

- Utilisateurs de la base, droits d'accès, rôles, profils
- Noms et caractéristiques des objets contenus dans la base :
  tables, vues, index, clusters, triggers, packages, contraintes d'intégrité, synonymes, procédures, fonctions, packages, …
- Ressources physiques allouées à la base (tablespaces, fichiers physiques)

Le dictionnaire est créé au moment de la création de la base et est mises à jour lors de l'utilisation de commandes DDL(data defintion language) sql. Si vous créez une nouvelle table, les données seront automatiquement mises à jour dans la BD. Vous ne pouvez en aucun cas modifier ce dictionnaire.

On ne peut donc qu'accéder le dictionnaire que pour des consultations (SELECT). Afin d'en simplifier cette consultation, des vues sont définies sur les tables du dictionnaire. Il est déconseillé d'accéder ce dictionnaire directement par les tables.

Il appartient à l'utilisateur *SYS*, mais l'utilisateur *SYSTEM*, c'est-à-dire l'administrateur de la base, possède des droits de lecture sur des vues du dictionnaire. Enfin le dictionnaire de données est conservé dans le tablespace *SYSTEM*.

III.3.2/ Les différentes vues

De nombreuses vues permettent à des utilisateurs d'accéder à certaines parties du dictionnaire de données. Les vues fournissent à l'administrateur de la base le meilleur moyen pour obtenir les caractéristiques techniques de celle-ci.

Les vues du dictionnaire de données sont classées par famille et nommées en fonction de l'appartenance à une de ces familles.
Voici la liste de ces familles de vues :

- Les vues **USER** (dont le nom commence par *USER_xxx*) donnent des informations sur tous les objets logiques dont l'utilisateur connecté est propriétaire (tables, index, vues, procédures, ...)
- Les vues **ALL** (dont le nom commence par *ALL_xxx*) fournissent des informations sur les objets pour lesquels l'utilisateur a un droit d'accès, c'est-à-dire les objets de la base créés par l'utilisateur ainsi que tous les objets accessibles par cet utilisateur.
- Les vues **DBA** (dont le nom commence par *DBA_xxx*). Ces vues sont réservées à l'administrateur de la base (DBA), afin de lui fournir des informations sensibles sur tous les objets de la base de données.

chaque XXX est en général remplacé par un nom (en anglais) significatif. Ainsi USER_TABLES est la vue de toutes MES tables, DBA_SYNONYMS est la vue de TOUS les synonymes du système.

- Les vues **V$** (dont le nom commence par *V$_*) sont des vues dynamiques permettant d'avoir des informations sur l'état courant de l'instance de la base de données de son démarrage à son arrêt. Elles permettent par exemple de connaître les fichiers physiques actuellement utilisés par la base (logs, rollback segments, ...).

V$VERSION	Nom du composant logiciel et no de version du noyau Oracle
V$DATABASE	Infos générales sur la base (nom, control files, reset logs, checkpoints, etc)
V$SESSION	Infos sur les sessions actuellement connectées
v$LOCK	Les verrous actifs et en attente
v$SGA	Taille et composition de la mémoire partagée
V$SQL	Ordres SQL en cache
V$SYSSTAT	Statistiques système

Remarque : Les vues commençant par 'ALL_' ne décrivent pas TOUS les objets mais les objets ACCESSIBLES à l'utilisateur courant. Ainsi ALL_TABLES ne donne pas la liste de toutes les tables de la base, c'est DBA_TABLES !

3.3/ Exemples : d'interrogation du data dictionary

SQL> `SELECT * FROM all_db_links;` Demande l'ensemble des liens utilisables sur la BD.

SQL> `SELECT * FROM V$DBFILE ;` Demande tous les datafiles de la BD.

```
SQL> select * from V$DBFILE;
     /ORACLE/DATABASE/USR1ORCL.ORA
     /ORACLE/DATABASE/RBS1ORCL.ORA
     /ORACLE/DATABASE/TMP1ORCL.ORA
     /ORACLE/DATABASE/SYS1ORCL.ORA
SQL>
```

Etant donné que le nombre de champs d'une table système est par fois très grand, il recommander de consulter la structure (les colonnes) d'une table du dictionnaire avant de l'interroger. Il suffit de faire un DESC. Par exemple :

SQL > `DESCRIBE v$session`

SQL > `DESCRIBE DBA_USERS`

```
          USERNAME                       NOT NULL  VARCHAR2(30)
          USER_ID                        NOT NULL  NUMBER
          PASSWORD                                 VARCHAR2(30)
          ACCOUNT_STATUS                 NOT NULL  VARCHAR2(32)
          LOCK_DATE                                DATE
          EXPIRY_DATE                              DATE
          DEFAULT_TABLESPACE             NOT NULL  VARCHAR2(30)
```

```
TEMPORARY_TABLESPACE          NOT NULL  VARCHAR2(30)
CREATED                       NOT NULL  DATE
PROFILE                       NOT NULL  VARCHAR2(30)
INITIAL_RSRC_CONSUMER_GROUP             VARCHAR2(30)
EXTERNAL_NAME                           VARCHAR2(4000)
```

SQL >

On obtient le meme resultat en regardant dans la vue DICT comment est décrite la VUE DBA_USERS :

SQL> `select * from DICT  where table_name= 'DBA_USERS'`

Annexe : Les tables systèmes d'oracle

Liste des vues statiques du dictionnaire Oracle pour l'utilisateur ou le développeur

Nom de la vue	synonyme	Contenu
DICTIONARY	DICT	Toutes les vues du dictionnaire ,pour le développeur ou le DBA : Nom de la vue, description
USER_TABLES	TABS	mes tables : nom, tablespace, stockage, statistiques, cluster éventuel
USER_TAB_COLUMNS	COLS	Colonnes de mes tables : Nom colonne, type, longueur, obligatoire
USER_VIEWS	-	Mes vues : Nom, texte de l'ordre SQL associé, type
USER_INDEXES	IND	Mes indexs : Nom, table indexée, unicité, stockage, statistiques
USER_IND_COLUMNS	-	Nom index, nom table, nom colonne, position et longueur
USER_CLUSTERS	CLU	Mes clusters ; Nom, stockage, statistiques
USER_OBJECTS	OBJ	Mes objets : tables, vues, indexes, clusters, synonymes, procédures, fonction, package, sequence
USER_SEQUENCES	SEQ	Mes séquences : Valeur min, max, increment, cycle, cache
USER_SYNONYMS	SYN	Mes synonymes : Nom du synonyme, de l atable, propriétaire et db link éventuel
USER_USERS	-	Caractéristiques générales du user : Nom, tablespace par defaut, tablespace temporaire
USER_CONSTRAINTS	-	De mes contraintes : Nom, type, table d'accueil, statut
USER_DB_LINKS	-	De mes database links (liens base distantes) : Nom, user distant, mot de passe, serveur distant, date de creation
USER_TAB_PRIVS	-	Des privilèges donnés ou reçus : Bénéficiaire, propriétaire, créateur
USER_EXTENTS	-	Caractéristiques de stockage de mes objets : Nom du segment, de la partition, du tablespace, taille en octets et en blocs
USER_TS_QUOTAS	-	Quota d'écriture autorisé sur les tablespace : Nom du tablespace, taille max en octets et en blocs

liste des vues statiques DBA les plus utilisées

TABLE_NAME	COMMENTS
DBA_CATALOG	All database Tables, Views, Synonyms, Sequences
DBA_CLUSTERS	Description of all clusters in the database
DBA_COLL_TYPES	Description of all named collection types in the database
DBA_COL_COMMENTS	Comments on columns of all tables and views

DBA_COL_PRIVS	All grants on columns in the database
DBA_CONSTRAINTS	Constraint definitions on all tables
DBA_CONS_COLUMNS	Information about accessible columns in constraint definitions
DBA_CONS_OBJ_COLUMNS	List of types an object column or attribute is constrained to in all tables in the database
DBA_DATA_FILES	Information about database data files
DBA_DB_LINKS	All database links in the database
DBA_DEPENDENCIES	Dependencies to and from objects
DBA_DIMENSIONS	Description of the dimension objects accessible to the DBA
DBA_DMT_FREE_SPACE	Free extents in all dictionary managed tablespaces
DBA_DMT_USED_EXTENTS	All extents in the dictionary managed tablespaces
DBA_ERRORS	Current errors on all stored objects in the database
DBA_EXTENTS	Extents comprising all segments in the database
DBA_INDEXES	Description for all indexes in the database
DBA_PARTIAL_DROP_TABS	All tables with partially dropped columns in the database
DBA_PENDING_TRANSACTIONS	information about unresolved global transactions
DBA_PROCEDURES	Description of all procedures
DBA_PROFILES	Display all profiles and their limits
DBA_ROLES	All Roles which exist in the database
DBA_ROLE_PRIVS	Roles granted to users and roles
DBA_ROLLBACK_SEGS	Description of rollback segments
DBA_RULES	Rules in the databse
DBA_SEGMENTS	Storage allocated for all database segments
DBA_SEQUENCES	Description of all SEQUENCEs in the database
DBA_SNAPSHOTS	All snapshots in the database
DBA_SYNONYMS	All synonyms in the database
DBA_SYS_PRIVS	System privileges granted to users and roles
DBA_TABLES	Description of all relational tables in the database
DBA_TABLESPACES	Description of all tablespaces
DBA_TAB_COLS	Columns of user's tables, views and clusters
DBA_TAB_COLUMNS	Columns of user's tables, views and clusters
DBA_TAB_COL_STATISTICS	Columns of user's tables, views and clusters
DBA_TAB_COMMENTS	Comments on all tables and views in the database
DBA_TAB_HISTOGRAMS	Histograms on columns of all tables
DBA_TAB_MODIFICATIONS	Information regarding modifications to tables
DBA_TAB_PRIVS	All grants on objects in the database
DBA_TRIGGERS	All triggers in the database
DBA_TRIGGER_COLS	Column usage in all triggers
	Tablespace quotas for all users

DBA_TS_QUOTAS	
DBA_TYPES	Description of all types in the database
DBA_UNDO_EXTENTS	Extents comprising all segments in the system managed undo tablespaces
DBA_UNUSED_COL_TABS	All tables with unused columns in the database
DBA_UPDATABLE_COLUMNS	Description of dba updatable columns
DBA_USERS	Information about all users of the database
DBA_VIEWS	Description of all views in the database

liste des vues dynamiques (permettant d'avoir des informations sur l'état courant de l'instance de la base de données)

V$VERSION	Nom du composant logiciel et no de version du noyau Oracle
V$DATABASE	Infos générales sur la base (nom, control files, reset logs, checkpoints, etc)
V$SESSION	Infos sur les sessions actuellement connectées
v$LOCK	Les verrous actifs et en attente
v$SGA	Taille et composition de la mémoire partagée
V$SQL	Ordres SQL en cache
V$SYSSTAT	Statistiques système

Remarque : Pour obtenir une liste complète des vues statiques DBA utiliser la requete suivante :

```
select * from dict  where table_name like 'DBA%'
```

III.4.2/ Création, ouverture, démarrage et arrêt d'une base de données

III.4.2.1/ Outil du DBA

`SVRMGR[L/M]` (parfois SQLDBA) : Outil principal du DBA est remplacé depuis la version 9i d'Oracle par :

```
sqlplus /nolog
connect  system/mot-passe  as sysdba
```

Cette session spéciale permet :
- de créer, démarrer et d'arrêter une base,
- de surveiller son activité en temps réel, d'effectuer, de façon générale, les opérations de gestion physique de la base (sauvegarde, restauration, ...)

Principales commandes de cette session DBA :

```
STARTUP { NOMOUNT | MOUNT | OPEN}
CREATE DATABASE ….
SHUTDOWN {NORMAL | IMMEDIATE | TRANSACTIONAL|ABORT}
```

Archivage :

```
ALTER DATABASE {ARCHIVELOG | NOARCHIVELOG}
```

Sauvegarde :

```
ALTER TABLESPACE nom_ts BEGIN BACKUP
```

Copier par une commande de l'OS tous les fichiers du tablespace nom_ts…

```
ALTER TABLESPACE nom_ts END BACKUP
```

Restauration de la base : `IMP et EXP`

Maintenance des fichiers :

```
ALTER DATABASE {ADD | DROP| RENAME}
LOGFILE …

CREATE ROLLBACK SEGMENT Y TABLESPACE X
DROP ROLLBACK SEGMENT Y
```

Cette session permet également d'effectuer des opérations de maintenance :

- create database .... (créer une base)
- alter database ... (modifier les paramètres)
- recover ... (restaurer une base)

III.4.2.2/ Statuts d'une base :

Une BD peut être active ou non. Quand elle n'est pas active, elle consiste en un ensemble de fichiers sur disque, non accessibles aux utilisateurs. On dit que la base est ARRÊTÉE.

Quand elle est active, une base peut avoir plusieurs statuts. Une base active est associée à une instance, matérialisée par des process et une zone : la SGA (System Global Area). Une base arrêtée devient active suite à l'ordre de démarrage : startup.

Les 3 statuts d'une base actives :

- **base non montée** : peut être obtenu par l'ordre : startup ... nomount. Ce statut est nécessaire pour créer la base la première fois. Il peut aussi être nécessaire pour tester les paramètres du fichier INIT.ORA : il faut se connecter en interne sur une base non montée.
- **base montée non ouverte** : ce statut peut être obtenu par la commande : startup ... mount. Peut également être obtenu par la demande de passage du statut "non montée" à "montée" (alter database mount). Ce status est nécessaire pour effectuer des opérations de maintenance : restauration (recover), ajoût ou suppression de fichiers redo log, le renommage des noms de fichiers, passage au mode archivage ou non (alter database add ou drop ou rename ou archivelog/noarchivelog). Ce statut ne permet pas d'accéder aux tables du dictionnaire et des utilisateurs). Il faut se connecter en interne.
- **base montée et ouverte** : peut être obtenu par la commande : startup ... open. ou par le passage du statut "montée" à "ouverte" (alter database open). C'est le statut de la base en mode production (utilisation normale). On s'y connecte par un nom d'utilisateur et un mot de passe.

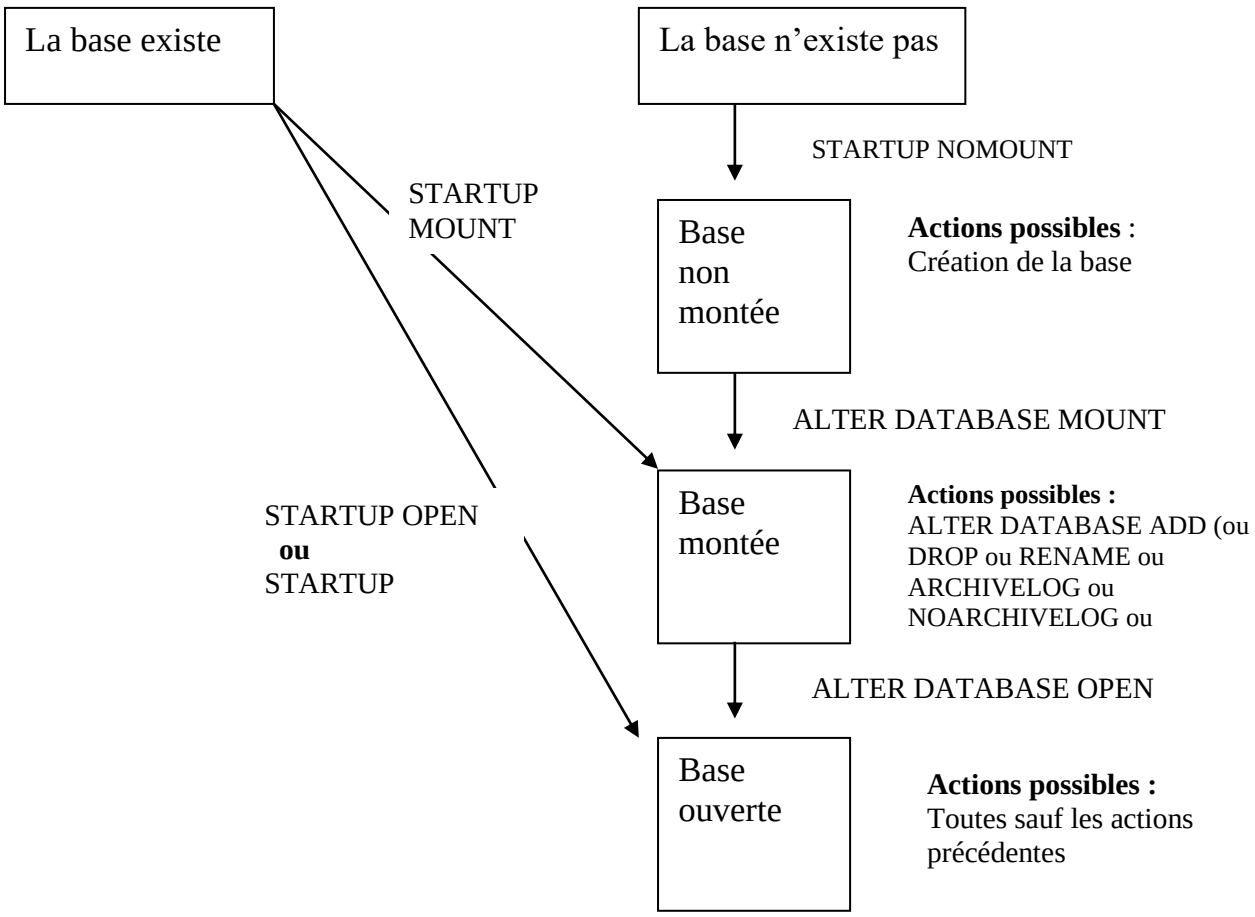


Figure : statuts d'une base de données

III.4.2.3/ Démarrage de la base

Le dba peut démarrer une base dans un des 3 statuts :

- **non montée :**

```
sqlplus /nolog
CONNECT / AS SYSDBA
Startup nomount pfile =
oracle/product/10.1.0/admin/orclp4/pfile/init.ora.1129200517498;
```

- **montée :**

```
alter database mount ;
```

- **ouverte :**

```
alter database open ;
```

On remarque que avec `alter` on peut toujours passer d'un statut à un autre.

On remarque également que si l'on veut utiliser le fichier d'initialisation créé avec l'installation d'Oracle, il faut bien repérer son chemin d'accès. Dans l'exemple précédent c'est le fichier `init.ora.1129200517498` du répertoire est : `oracle/product/10.1.0/admin/orclp4/pfile/`

Démarage en un seul ordre :

```
sqlplus /nolog
CONNECT / AS SYSDBA
Startup pfile =
  oracle/product/10.1.0/admin/orclp4/pfile/init.ora.1129200517498;
```

III.4.2.4/ Arrêt de la base

C'est la commande **shutdown avec** 3 variantes :

- **shutdown normal** : oracle attend que tous les utilisateurs se déconnectent pour fermer la base, mais aucun nouvel utilisateur ne peut se connecter.
- **shutdown transactional** : oracle attend la fin des transactions en cours, déconnecte les utilisateurs et arrête la base dans un état cohérent.
- **shutdown immediate** : oracle défait les transactions en cours (rollback), déconnecte les utilisateurs et arrête la base dans un état cohérent.
- **shutdown abort** : déconnexion immédiate des utilisateurs, sans défaire les transaction en cours éventuelles, et arrêt de la base dans un état qui peut être incohérent. La cohérence est rétablie au prochain démarrage. C'est le moyen le plus rapide pour arrêter une base.

Fermeture de la BD : C'est la rendre indisponible à l'ensemble des utilisateurs, mais elle reste montée : `Alter database close immediate ;`

Démontage de la BD : Cette action supprime l'association entre les fichiers de la base et l'instance : `Alter database dismount ;`

III.4. Création, démarrage et arrêt d'une base de données

Rappel : Une fois l'installation terminée, quatre comptes utilisateurs sont créés dès l'installation :

- le compte **internal** (avec comme mot de passe initial **oracle**),
- le compte **sys** (avec comme mot de passe initial **change_on_install**),
- le compte **system** (gratifié du mot de passe **manager**),
- et le compte **scott** (avec le mot de passe **tiger**).

III.4.1/ Le fichier d'initialisation (paramètres)

Il s'agit du fichier `init.ora`. A chaque démarrage d'une instance d'ORACLE, ce fichier sera lu, ces données seront utilisées pour paramétrer l'instance Oracle.

Le plus simple, quand on n'a pas l'habitude d'écrire un tel fichier est d'en reprendre un déjà existant et de le modifier en fonction de nos besoins. Justement, lors de l'installation d'Oracle, une base initiale à était créée : copions-y le fichier en question.

Remarque : Il serait préférable d'avoir créé un nouveau répertoire pour contenir la nouvelle base.

Exemple du fichier init.ora

```
db_block_size=8192

open_cursors=300

db_domain="iut.univ-lehavre.fr"

db_name=orclp4

background_dump_dest=C:\oracle\product\10.1.0\admin\orclp4\bdump
```

```
control_files=
        ("C:\oracle\product\10.1.0\oradata\orclp4\control01.ctl",
        "C:\oracle\product\10.1.0\oradata\orclp4\control02.ctl",
        "C:\oracle\product\10.1.0\oradata\orclp4\control03.ctl")
processes=150
```

Description de quelques paramètres du fichier `init.ora`

bd_name : ce premier paramètre correspond au nom de la base. Chaque nom de BD doit être différent. Ce nom apparaîtra dans la colonne name de V$DATABASE : `SELECT * FROM V$DATABASE`.

control_files : ce paramètre sert à fixer les noms (et donc la localisation) des fichiers de contrôle de la BD, ils sont visualisable par : `SELECT * FROM V$CONTROLFILE;`

db_files : Le paramètre db_file fixe la nombre de fichiers utilisables par une instance de la BD.

log_files : Ce paramètre permet de limiter le nombre de fichiers de Redo Log pour l'instance en cours, mais ne permet pas de dépasser MAXLOGFILES.

db_block_size **: Ce paramètre sert à fixer la taille des blocks pour la BD. U*ne fois fixée, la valeur de ce paramètre ne pourra plus jamais être modifiée pour cette BD.***

III.4.2.5/ Création d'une nouvelle base de données

L'application DBCA permet de créer la base en utilisant une interface graphique

Utiliser la commande SQL `DATABASE`**.**

Exemple :

```
connect SYS/change_on_install as SYSDBA
startup nomount pfile="/u01/app/oracle/admin/.../init.ora";

CREATE DATABASE basename
 LOGFILE GROUP 1 ('/disk1/base/base_log1.log',
             '/disk2/base/base_log1.log') SIZE 30K,
          GROUP 2 ('/disk1/base/base_log2.log',
             '/disk2/base/base_log2.log') SIZE 30K

DATAFILE('/disk3/base/datafile1.dbf','/disk4/base/datafile1.dbf')SIZE
20M;
```

La tablespace SYSTEM est créé par l'ordre CREATE DATABASE. La base précédente est donc créée dans le même tablespace que le dictionnaire de données. Cela est peu recommandé.