

Gérer les structures de stockage de base de données

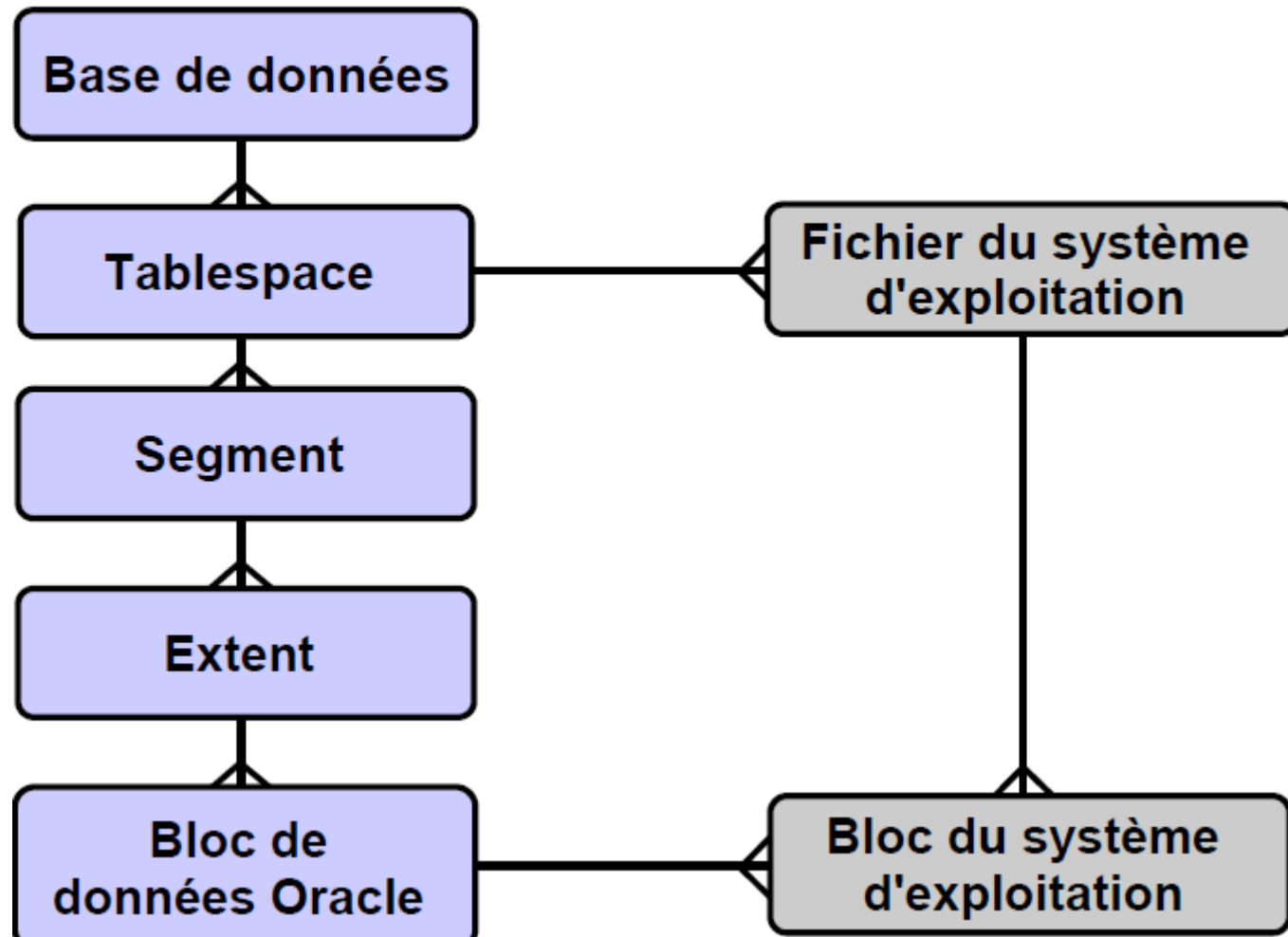
Plan adopté

- 1 Structure de stockage d'une base de données
- 2 Types de tablespaces
- 3 Types de segments
- 4 Mode de stockage des données d'une table
- 5 Gestion (manuelle/automatique des tablespaces)
- 6 Base de données pré-configurée
- 7 Commandes SQL pour la gestion des structures de stockage

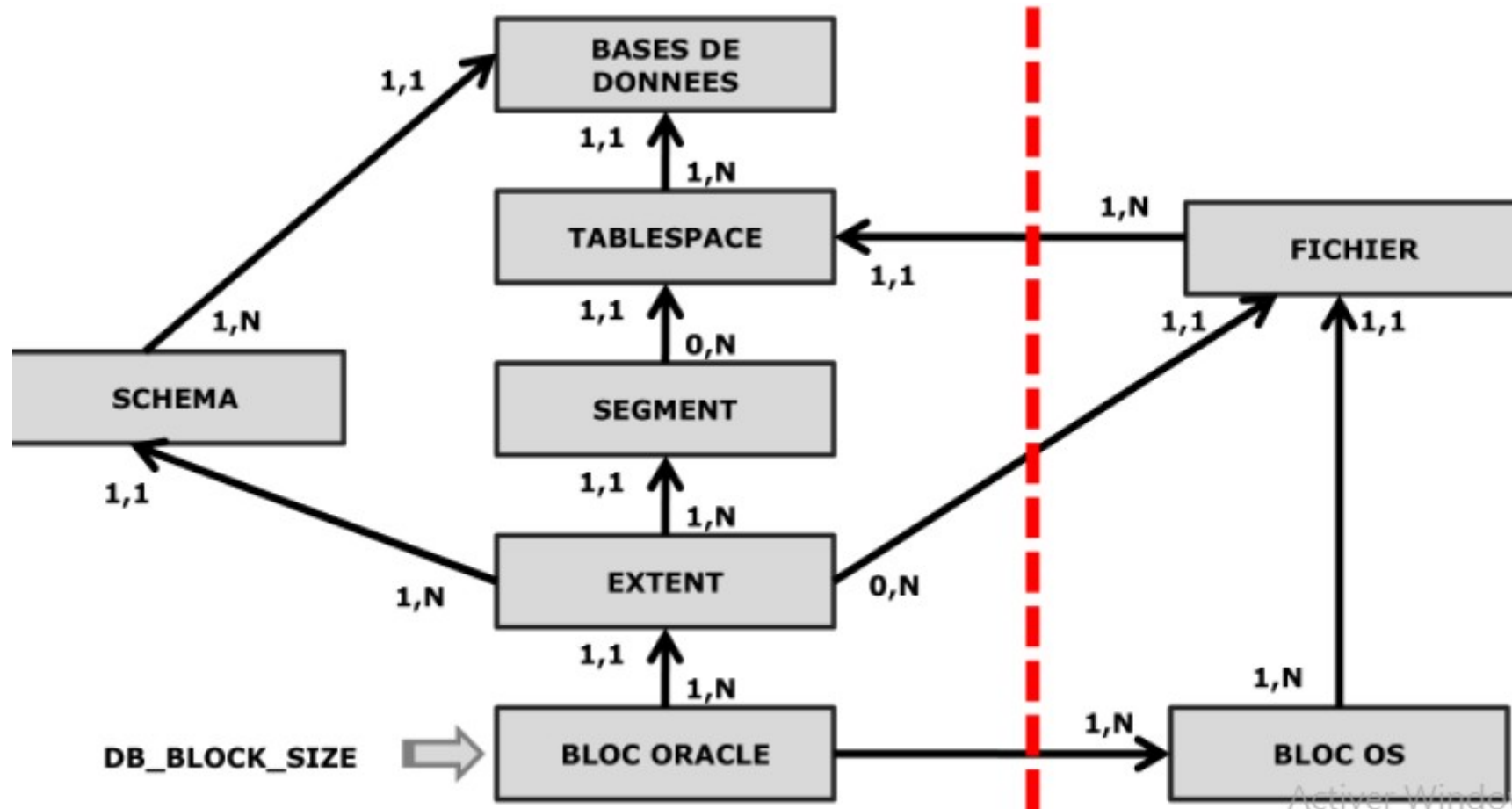
Structure de stockage d'une base de données

Structure logique

Structure physique

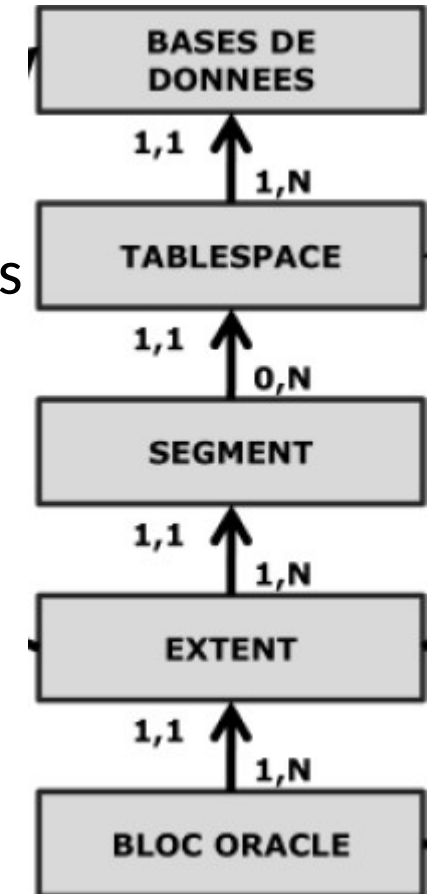


Structure de stockage d'une base de données



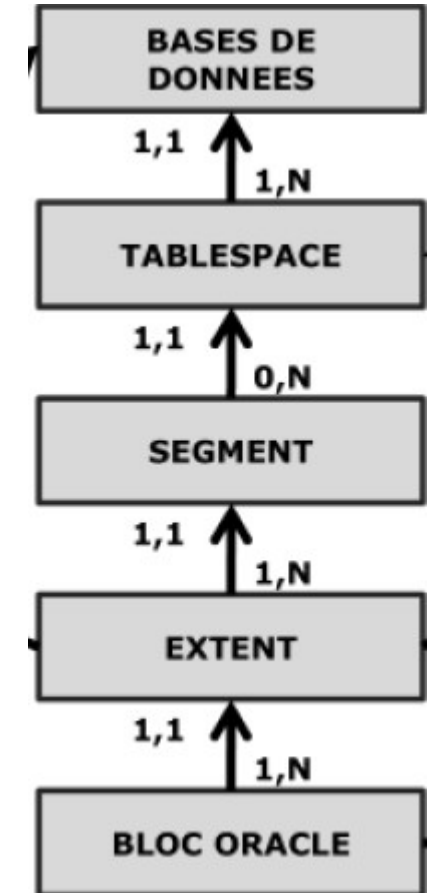
Structure logique

- ❑ Une base de données est divisée en unités de stockage logiques appelées tablespaces, qui peuvent être utilisées pour regrouper des structures logiques liées.
- ❑ Exemple : Tablespace de données est logiquement liée au stockage des objets (tables, index, vues...) créés par les utilisateurs. Tablespace Undo est logiquement liée aux données de gestion d'annulation...
- ❑ Les objets de base de données (Users, information UNDO, information temporary, tables, index....) sont stockés dans les tablespaces. Chaque tablespace contiendra les objets lui correspondant.
- ❑ Un tablespace est constitué de segments.



Structure logique

- ❑ Les objets cités précédemment sont stockés dans les tablespaces sous forme de segments. Mais chaque objet sur un segment différent
- ❑ Exemple : Un Tablespace de données contient les objets créés par les utilisateurs (tables, index, vues). Mais pour faire la différence entre un objet et un autre, chaque objet va résider sur un segment différent du tablespace. Donc les lignes d'une table résident sur un segment différent appartenant au tablespace de données
- ❑ Un segment appartient à un et un seul tablespace



Structure logique

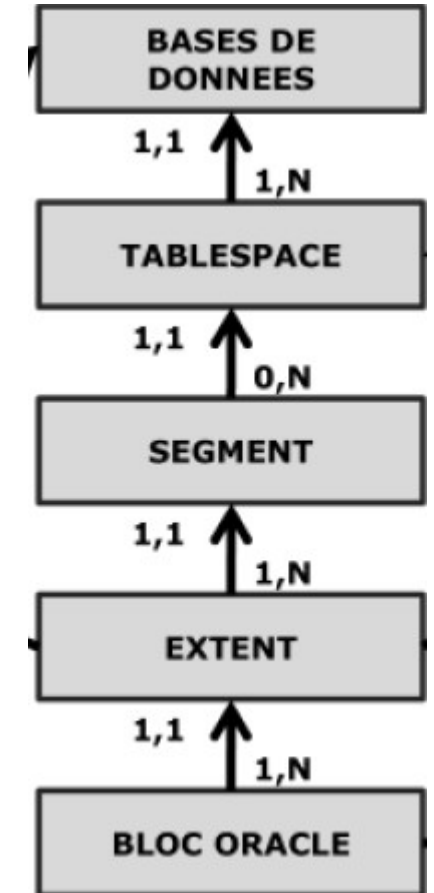
```
SYS@gescom>create table titi (num int) tablespace users;
```

Table créée.

```
SYS@gescom>select SEGMENT_NAME,SEGMENT_TYPE from dba_segments WHERE segment_name='TITI'
```

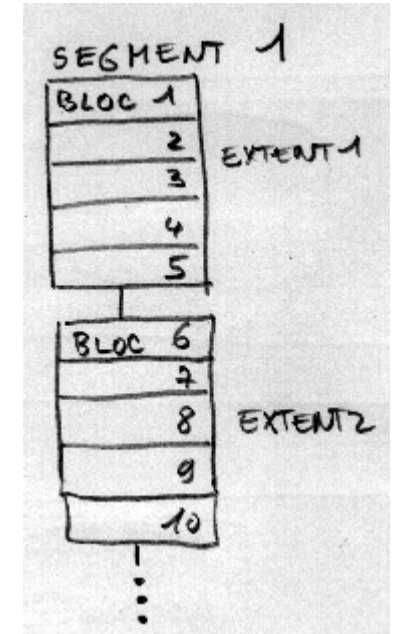
SEGMENT_NAME	SEGMENT_TY
TITI	TABLE

- ❑ Oracle a créé un segment qui s'appelle titi correspondant à l'objet (table titi) qui vient d'être créé.
- ❑ Un segment lui correspond un objet | | un objet lui correspond un segment
- ❑ Chaque segment contient un ou plusieurs extents



Structure logique

- ❑ Un extent est constitué d'un ensemble de blocs de contigus du disque.
- ❑ Exemple : Soit le segment S associé à la table T. En voulant rajouter des lignes à T oracle devra rajouter de l'espace mémoire à S (s'il est saturé). Cet espace mémoire étant l'extent. Qui est un ensemble de block
- ❑ Les blocs de données constituent le dernier niveau de granularité.
- ❑ La base de données Oracle alloue de l'espace de manière dynamique. Lorsque les extents existants d'un segment sont pleins, d'autres sont ajoutés. Les extents sont alloués en fonction des besoins. Par conséquent, les extents d'un segment peuvent ne pas être contigus sur le disque.




Structure logique

- ❑ La taille d'un bloc varie entre 2ko, 4ko, 8ko, 16ko et 32ko
- ❑ La taille du bloc de données peut être définie lors de la création de la base de données. La taille par défaut (8 ko) est adaptée à la plupart des bases de données.
- ❑ Les blocks oracle c'est le niveau de granularité le plus bas. Pour connaître la taille par défaut des blocks de données il suffit de faire :

```
SQL> show parameter DB_BLOCK_SIZE
```

NAME	TYPE	VALUE
db_block_size	integer	8192




- ❑ Il est à noter que DB_BLOCK_SIZE est la taille par défaut du block de donnée oracle (ici 8k) utilisée dans les tablespace. Mais on peut changer sa valeur d'un tablespace à un autre

Structure logique

- ❑ Exemple si on tape la commande suivante :

```
SYS@gescom>SELECT TABLESPACE_NAME,BLOCK_SIZE from dba_tablespaces;
```

TABLESPACE_NAME	BLOCK_SIZE
SYSTEM	8192
SYSAUX	8192
UNDOTBS1	8192
TMP	8192
USERS	8192
APP_INDEX_01	8192
BIG_TBS	8192
TEMP01	8192
TEMP02	8192
TBS_16K	16384
TBS_4K	4096



11 ligne(s) sélectionnée(s).

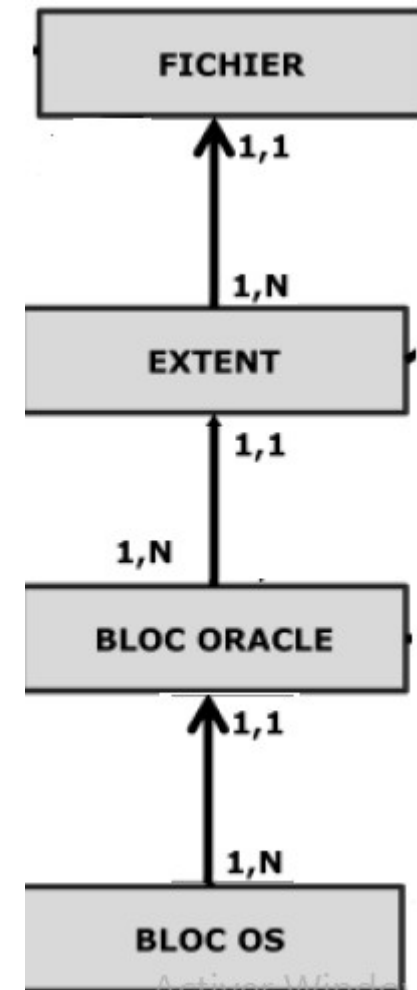
- ❑ Il est clair ici que la taille du block utilisé par tablespace peut changer
- ❑ L'unité utilisée est l'octet

Structure logique

- ❑ Si votre base de données prend en charge une application de data warehouse(entrepôt de données : est un gigantesque tas d'informations épurées, organisées, historisées et provenant de plusieurs sources de données, servant aux analyses et à l'aide à la décision) qui comporte des tables et des index volumineux, il est judicieux de définir une taille de bloc plus importante.
- ❑ Si votre base de données prend en charge une application transactionnelle dans laquelle les lectures et les écritures sont aléatoires, il peut s'avérer utile de définir une taille de bloc inférieure.

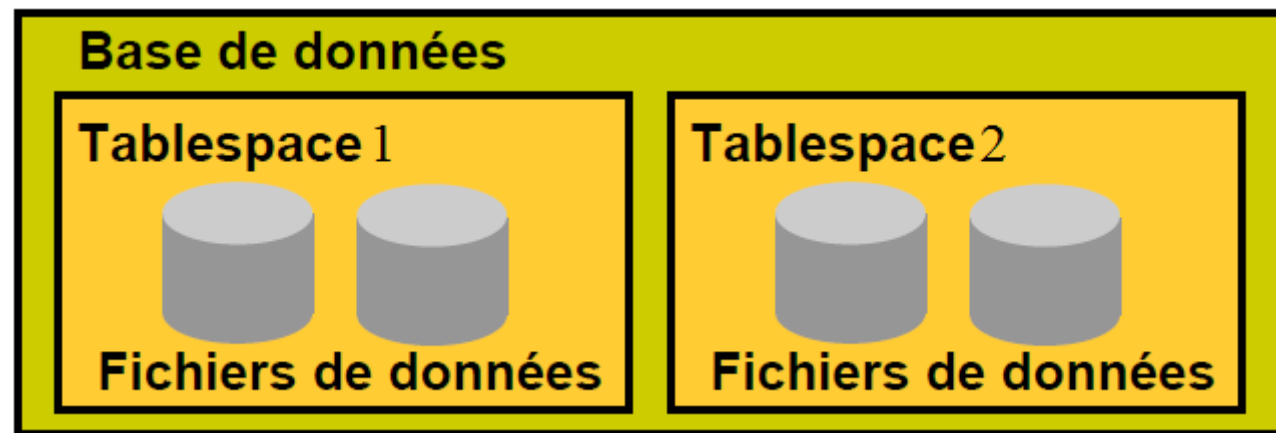
Structure physique

- ❑ Les données sont réellement stockées sur des fichiers. Se sont les conteneurs physique.
- ❑ Un fichier est contient un ensemble d'extent
- ❑ Un extent est un ensemble de bloc oracle
- ❑ Et un block oracle contient un ensemble de block OS



Relation entre structure physique et logique

- ❑ Un tablespace ne peut appartenir qu'à une seule base de données
- ❑ Chaque tablespace d'une base Oracle est constitué d'un ou de plusieurs fichiers appelés fichiers de données. Il s'agit de structures physiques sur lequel les données seront stockées.
- ❑ Les fichiers de données : ne peuvent appartenir qu'à un seul tablespace et à une seule base de données



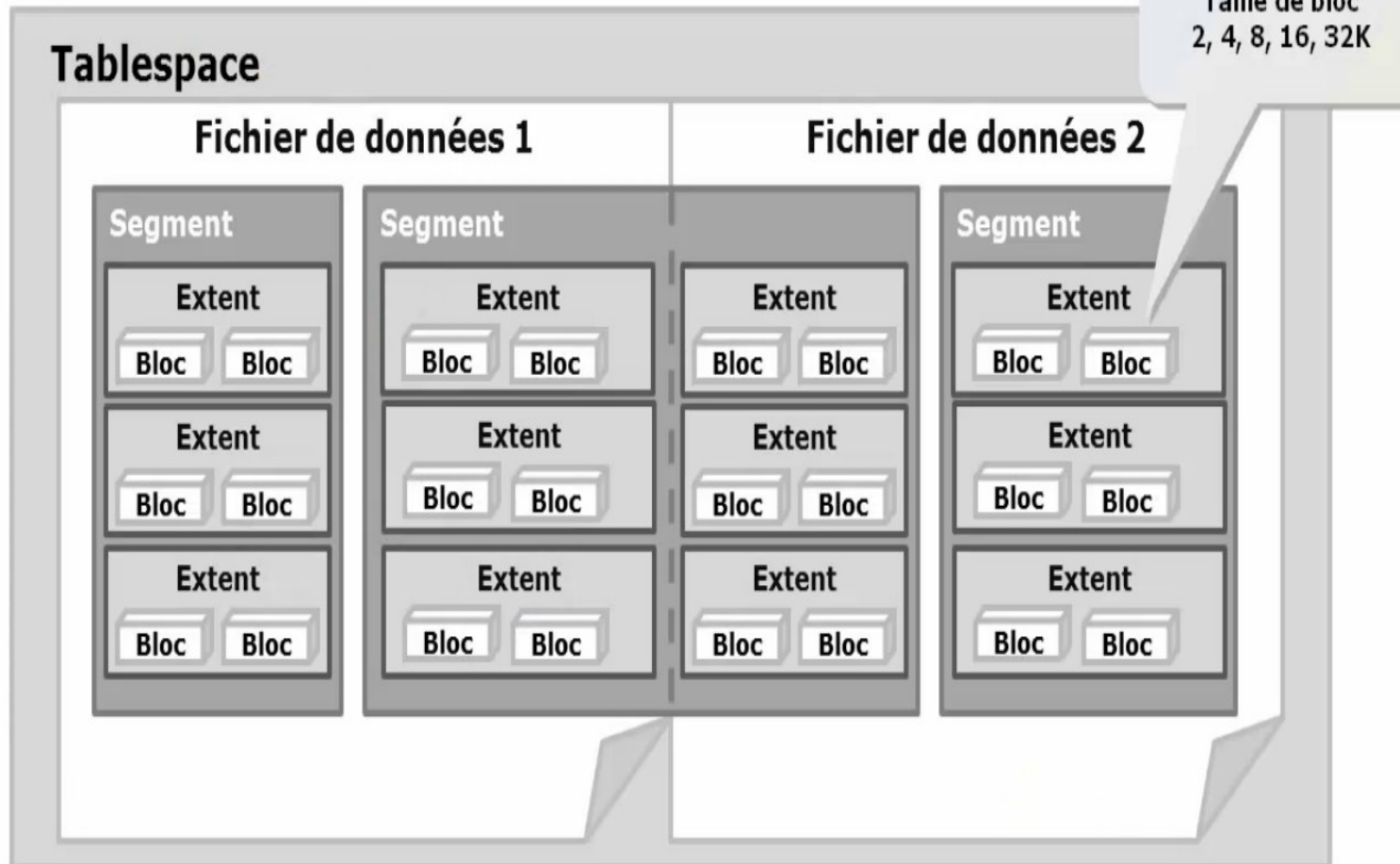
Relation entre structure physique et logique

- ❑ La commande suivante affiche l'ensemble des tablespace et les fichiers correspondants

TABLESPACE_NAME	FILE_NAME
SYSTEM	/u101/oradata/gescom/system_01.dbf
SYSAUX	/u102/oradata/gescom/sysaux_01.dbf
UNDOTBS1	/u102/oradata/gescom/undotbs1_01.dbf
USERS	/u101/oradata/gescom/users_01.dbf
APP_INDEX_01	/u105/oradata/gescom/app_index_01.dbf
APP_INDEX_01	/u105/oradata/gescom/app_index_02.dbf
BIG_TBS	/u106/oradata/gescom/big_tbs.dbf
TBS_4K	/u107/oradata/gescom/tbs_4k01.dbf

Relation entre structure physique et logiques

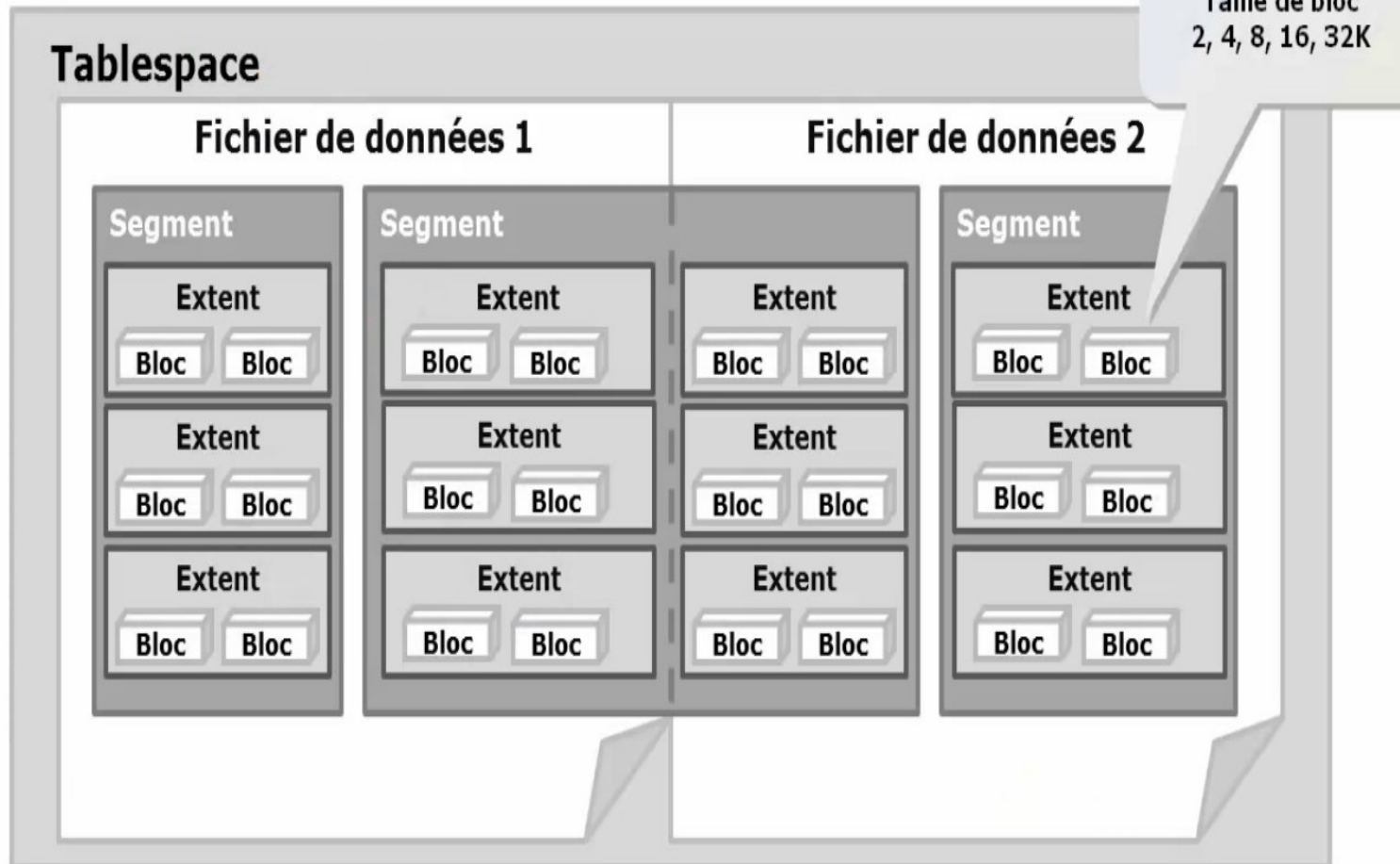
■ Architecture



- ❑ Dans un fichier de données on peut trouver plusieurs segments et donc différents objets chacun sur un segment
- ❑ Un segment peut s'étendre sur plus qu'un fichier. Donc un objet peut s'étendre sur plus qu'un fichier. Ceci dans le cas où le premier fichier est saturé et ne peut pas comprendre la totalité de l'objet

Relation entre structure physique et logiques

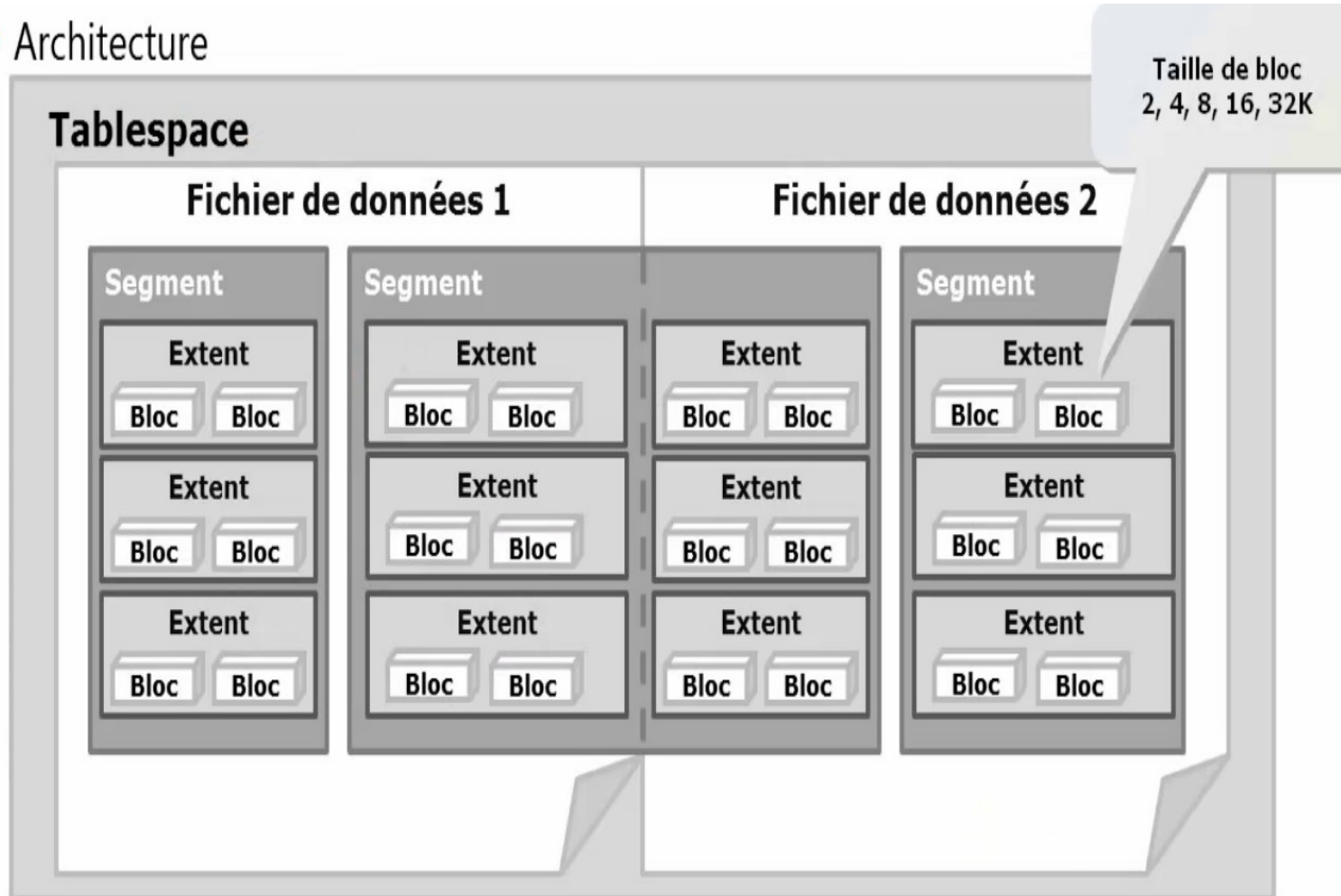
■ Architecture



- ❑ Lors de l'ajout de données d'un objet(exemple les lignes d'une table) et quand les extent déjà alloué d'un fichier sont saturé. Oracle alloue un autre extent.
- ❑ Un extent créé sera rattaché par Oracle à un seul segment
- ❑ Un extent créé sera rattaché par Oracle à un seul fichier

Relation entre structure physique et logiques

■ Architecture



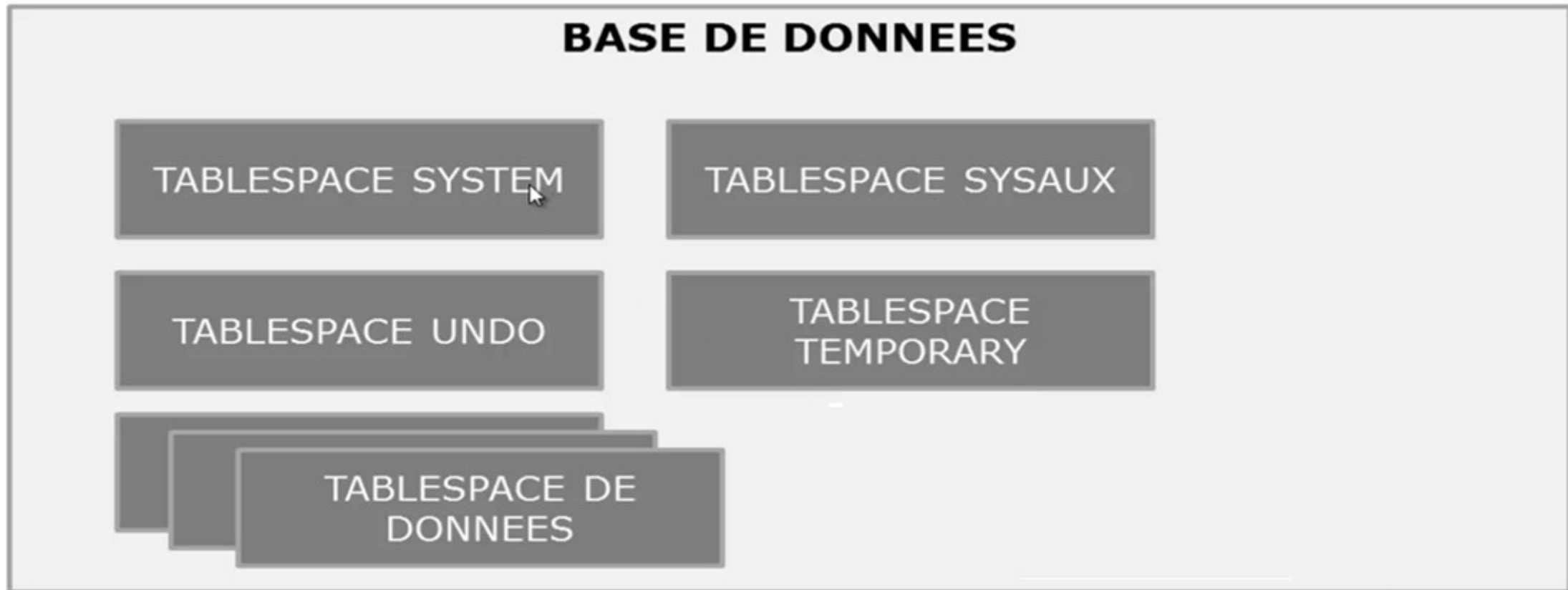
- ❑ Un block pris par Oracle du SE sera rattaché à un seul extent et à un seul segment
- ❑ Un block pris par Oracle du SE sera rattaché à un seul fichier

QCM récapitulatif

1. Une table peut elle appartenir à différents tablespaces
2. Une table peut elle appartenir à différents segments
3. Une table peut elle appartenir(ou contenir) différents fichiers
4. Une table peut elle appartenir(ou contenir) différents extents
5. Une table peut elle appartenir(ou contenir) différents blocks de données
6. Un extent peut il appartenir à différentes tables de données
7. Un extent peut il appartenir à différents fichiers
8. Un block peut appartenir à différents extents?
9. Un fichier peut contenir les données d'une table et d'une vue et d'autres objets au même temps
10. quand est ce ajout ou suppression d'extent

Types de tablespace

Les types de tablespace qu'on rencontre en général :



Types de tablespace

❑ **SYSTEM** : Le tablespace SYSTEM est utilisé par le serveur Oracle pour gérer la base de données. Il contient le dictionnaire de données et les tables comprenant les informations d'administration sur la base. Les utilisateurs normaux ne doivent pas manipuler ce tablespace seulement l'administrateur en a le droit. Ils ne sont accessibles que par l'utilisateur SYS ou par les autres administrateurs dotés du privilège approprié. Ce type de tablespace doit toujours rester en ligne.

❑ Le dictionnaire de données s'agit d'un ensemble de tables systèmes contenant les informations relatives à la structure de la base de données :

- Utilisateurs de la base (ainsi que leurs privilèges et leur rôle)
- Noms et caractéristiques des objets contenus dans la base (tables, vues, index, clusters, triggers, packages, ...)
- Contraintes d'intégrité
- Les noms de tablespaces et les fichiers correspondant ainsi que le taux d'allocation
-

Types de tablespace

- ❑ Exemple de tables système du dictionnaire de données : dba_segments, dba_tablespace....
- ❑ Ce dictionnaire de donnée existe dans le tablespace SYSTEM. Ce dernier doit être alors toujours en ligne
- ❑ SYSAUX : Il s'agit d'un tablespace auxiliaire du tablespace SYSTEM. Certains composants et produits logiciel Oracle qui utilisaient le tablespace SYSTEM ou leurs propres tablespaces dans les versions antérieures de la base de données Oracle utilisent désormais le tablespace SYSAUX. Chaque instance de base de données Oracle doit comporter un tablespace SYSAUX. Ce tablespace doit rester en ligne pour faire fonctionner ces composants

Types de tablespace

- ❑ Un tablespace temporaire « TEMPORARY » est défini pour héberger les opérations de tri de données. En effet, lors d'importantes opérations de tri (telles que select distinct, union et create index), si la taille de la zone PGA du processus utilisateur ne suffit pas, Oracle va stocker dans les tablespaces TEMPORARY de la base de données des informations concernant le tri des enregistrements avant de retourner l'information aux utilisateurs. En raison de leur nature dynamique, ces espaces de tris ne devraient pas être stockés avec d'autres types de Tablespace.
- ❑ Il est recommandé de définir un tablespace temporaire par défaut pour la base de données. Sauf indication contraire, ce tablespace est affecté à chaque utilisateur nouvellement créé.

Types de tablespace

- ❑ Certains utilisateurs d'une base de données Oracle peuvent avoir besoin de volumes de stockage temporaires beaucoup plus grands que ceux de tous les autres utilisateurs de l'application. Dans ce cas, vous pouvez créer plusieurs tablespaces temporaires « TEMPORARY », pour distribuer les espaces de stockages des utilisateurs ayant des besoins semblables sur les mêmes tablespaces.
- ❑ tablespace « UNDO » : Toutes les données d'annulation(de rollback) sont stockées dans ce tablespace. Se sont toutes les traces de modification faites sur les objets de la base de données.(fichier redoLog...)
- ❑ Tablespace de données contient (les lignes des tables, les index, les vues....) Il en existe plusieurs car l'administrateur peut les distribuer sur les classes d'utilisateurs.

Types de segments

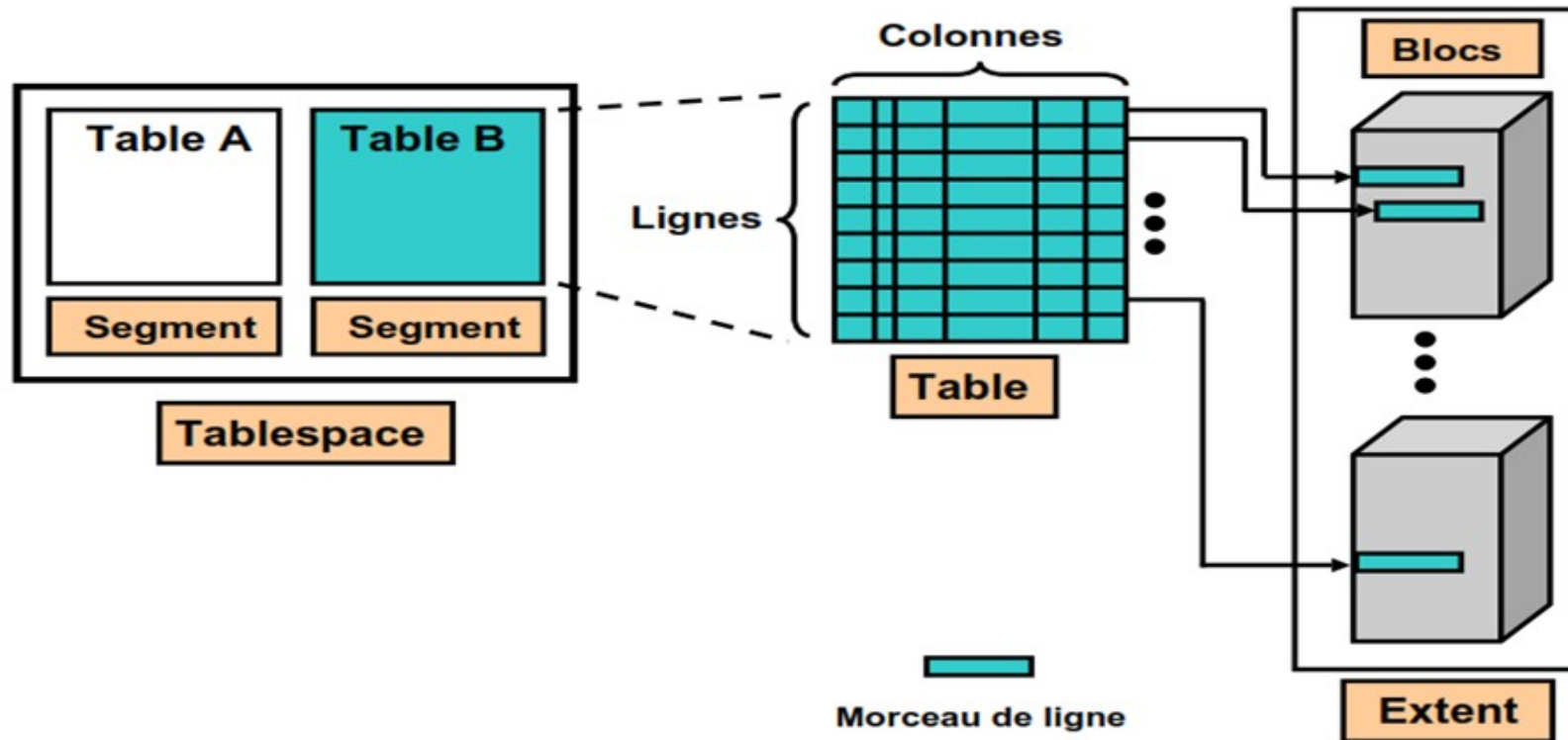
Quatre types de segment sont possibles :

- ❑ Les segments de données : Quand on est dans un tableSpace données les segments dedans peuvent être de type données. Chaque table de la base porte sur un segment de ce tablespace. Et donc L'ensemble des données de cette table sont stockées dans les extents du segment de données de cette table. Quand tu tape « create table ou create vue » ceci crée un segment de donnée associé à cet objet dans le tablespace par défaut du compte user.
- ❑ Le segment d'index : L'objet index sont différent dans leur gestion des autre objets. Chaque index créé de la base porte sur un segment d'index. Un segment de ce type va exister dans un tablespace donnée. Donc l'ensemble des données de cet index sont stockées dans les extents du segment d'index quand tu tape create index ceci crée un segment d'index assosié à cet objet dans le tablespace par défaut du compte user. et donc le résultat de l'opération sera mis dans ce nouveau segment.

Types de segments

- ❑ Les segments temporaires Quand on est dans un tableSpace temporaire les segments dedans sont de type temporaire. Les segments temporaires sont créés par la base de données Oracle (expl: l'exécution d'une instruction SQL requiert une zone de travail)
- ❑ Les segments undo(d'annulation): L'administrateur de base de données crée un tablespace d'annulation (UNDO) pour la base de donnée afin de stocker de manière temporaire les informations d'annulation. Les segments créés au sein de ce tablespace sont des segments d'annulation.
 - ➔ Comme déjà noté chaque type de segment sert à modéliser un type d'information(données, index, undo, temporary) et cette différence influe sur la manière avec laquelle Oracle va gérer ces segment
 - ➔ Exemple : parmi ces types de segment quels sont ceux que oracle doit vider et ceux non

Mode de stockage des données d'une table



Lorsqu'une table est créée, un segment de données est créé pour le stockage de ses données. On parle de "morceau de ligne" car, dans certaines conditions, une ligne n'est pas stockée intégralement dans le même emplacement (donc pas dans le même block). C'est le cas lorsqu'une ligne insérée est trop volumineuse pour tenir dans un bloc unique ou lorsque, suite à une mise à jour, la taille d'une ligne existante dépasse l'espace en cours du block.

Commandes SQL pour la gestion des structures de stockage

1. Création de tablespace permanent ou de données:

```
CREATE [ SMALLFILE | BIGFILE ] TABLESPACE 'tablespace_name'  
[ DATAFILE 'file_name' ]  
[ SIZE 'value' [ K | M | G | T ] ]  
[ AUTOEXTEND { OFF | ON [ NEXT 'value' [ K | M | G | T ] ] } ]  
[ MAXSIZE { UNLIMITED | 'value' [ K | M | G | T ] } ],  
[ OTHER DATAFILE..... ]  
[ LOGGING | NOLOGGING ]  
[ ONLINE | OFFLINE | READ ONLY ]  
[ BLOCKSIZE 'value' [ K ] ]
```

Commandes SQL pour la gestion des structures de stockage

1. Création de tablespace permanent ou de données:

```
CREATE [ SMALLFILE | BIGFILE ] TABLESPACE 'tablespace_name'  
[ DATAFILE 'file_name' ]  
[ SIZE 'value' [ K | M | G | T ] ]  
[ AUTOEXTEND { OFF | ON [ NEXT 'value' [ K | M | G | T ] ] } ]  
[ MAXSIZE { UNLIMITED | 'value' [ K | M | G | T ] } ],  
[ OTHER DATAFILE..... ]  
[ LOGGING | NOLOGGING ]  
[ ONLINE | OFFLINE ]  
[ BLOCKSIZE 'value' [ K ] ]
```

- ☐ Par défaut SMALLFILE
- ☐ Le cas SMALLFILE peut contenir plusieurs fichiers. Par contre un BIGFILE ne peut contenir qu'un seul fichier.
- ☐ Le cas SMALLFILE chaque fichier peut aller dans sa taille jusqu'à (2^{22}) blocs
- ☐ Le cas BIGFILE le fichier peut avoir la taille (2^{32}) blocs

Commandes SQL pour la gestion des structures de stockage

1. Création de tablespace permanent ou de données:

```
CREATE [ SMALLFILE | BIGFILE ] TABLESPACE 'tablespace_name'  
[ DATAFILE 'file_name' ]  
[ SIZE 'value' [ K | M | G | T ] ]  
[ AUTOEXTEND { OFF | ON [ NEXT 'value' [ K | M | G | T ] ] } ]  
[ MAXSIZE { UNLIMITED | 'value' [ K | M | G | T ] } ],  
[ OTHER DATAFILE..... ]  
[ LOGGING | NOLOGGING ]  
[ ONLINE | OFFLINE ]  
[ BLOCKSIZE 'value' [ K ] ]
```

❑ Ici on précise les fichiers qui seront rattaché au tablespace

Commandes SQL pour la gestion des structures de stockage

1. Création de tablespace permanent ou de données:

```
CREATE [ SMALLFILE | BIGFILE ] TABLESPACE 'tablespace_name'  
[ DATAFILE 'file_name' ]  
[ SIZE 'value' [ K | M | G | T ] ]  
[ AUTOEXTEND { OFF | ON [ NEXT 'value' [ K | M | G | T ] ] } ]  
[ MAXSIZE { UNLIMITED | 'value' [ K | M | G | T ] } ],  
[ OTHER DATAFILE..... ]  
[ LOGGING | NOLOGGING ]  
[ ONLINE | OFFLINE ]  
[ BLOCKSIZE 'value' [ K ] ]
```

❑ Ici on précise si le fichier est en mode incrémentation automatique ou pas. Si on active l'option ON alors on doit préciser le pas d'incrémentation (NEXT value)

Commandes SQL pour la gestion des structures de stockage

1. Création de tablespace permanent ou de données:

```
CREATE [ SMALLFILE | BIGFILE ] TABLESPACE 'tablespace_name'  
[ DATAFILE 'file_name' ]  
[ SIZE 'value' [ K | M | G | T ] ]  
[ AUTOEXTEND { OFF | ON [ NEXT 'value' [ K | M | G | T ] ] } ]  
[ MAXSIZE { UNLIMITED | 'value' [ K | M | G | T ] } ],  
[ OTHER DATAFILE..... ]  
[ LOGGING | NOLOGGING ]  
[ ONLINE | OFFLINE ]  
[ BLOCKSIZE 'value' [ K ] ]
```

❑ Ici on peut limiter la taille du fichier (par défaut c'est UNLIMITED). Si on veut limiter la taille on donne une valeur

Commandes SQL pour la gestion des structures de stockage

1. Création de tablespace permanent ou de données:

```
CREATE [ SMALLFILE | BIGFILE ] TABLESPACE 'tablespace_name'  
[ DATAFILE 'file_name' ]  
[ SIZE 'value' [ K | M | G | T ] ]  
[ AUTOEXTEND { OFF | ON [ NEXT 'value' [ K | M | G | T ] ] } ]  
[ MAXSIZE { UNLIMITED | 'value' [ K | M | G | T ] } ],  
[ OTHER DATAFILE..... ]  
[ LOGGING | NOLOGGING ]  
[ ONLINE | OFFLINE ]  
[ BLOCKSIZE 'value' [ K ] ]
```

- ☐ Par défaut LOGGING
- ☐ Cette option précise si on veut que les transaction lancées sur les objets de ce tablespace seront enregistrées dans les fichier journaux redolog ou pas

Commandes SQL pour la gestion des structures de stockage

1. Création de tablespace permanent ou de données:

```
CREATE [ SMALLFILE | BIGFILE ] TABLESPACE 'tablespace_name'  
[ DATAFILE 'file_name' ]  
[ SIZE 'value' [ K | M | G | T ] ]  
[ AUTOEXTEND { OFF | ON [ NEXT 'value' [ K | M | G | T ] ] } ]  
[ MAXSIZE { UNLIMITED | 'value' [ K | M | G | T ] } ],  
[ OTHER DATAFILE..... ]  
[ LOGGING | NOLOGGING ]  
[ ONLINE | OFFLINE ]  
[ BLOCKSIZE 'value' [ K ] ]
```

- ☐ Par défaut ONLINE
- ☐ Cette option précise si on veut que ce tablespace soit en ligne ou pas. S'il l'est alors on peut créer des objets au sein de ce tablespace. Sinon il est hors ligne la création d'objets dans ce tablespace ne sera pas possible.

Commandes SQL pour la gestion des structures de stockage

2. Création de tablespace temporaire:

```
CREATE [ SMALLFILE | BIGFILE ] TEMPORARY TABLESPACE 'tablespace_name'  
[ TEMPFILE 'file_name' ]  
[ SIZE 'value' [ K | M | G | T ] ]  
[ AUTOEXTEND { OFF | ON [ NEXT 'value' [ K | M | G | T ] ] } ]  
[ MAXSIZE { UNLIMITED | 'value' [ K | M | G | T ] } ],  
[ OTHER DATAFILE.....]
```

Commandes SQL pour la gestion des structures de stockage

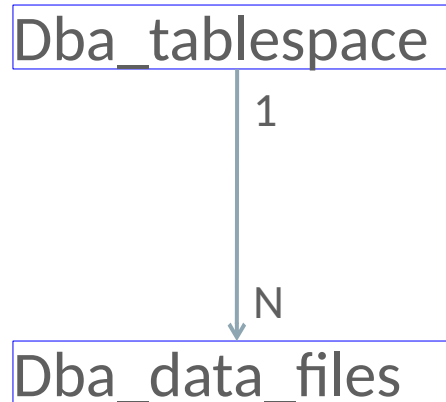
3. Création de tablespace UNDO:

```
CREATE [ SMALLFILE | BIGFILE ] UNDO TABLESPACE 'tablespace_name'  
[ DATAFILE 'file_name' ]  
[ SIZE 'value' [ K | M | G | T ] ]  
[ AUTOEXTEND { OFF | ON [ NEXT 'value' [ K | M | G | T ] ] } ]  
[ MAXSIZE { UNLIMITED | 'value' [ K | M | G | T ] } ],  
[ OTHER DATAFILE.....]
```

Commandes SQL pour la gestion des structures de stockage

VUES UTILES POUR L'ADMINISTRATION DES TABLESPACE:

1. **dba_data_files** (FILE_NAME, #TABLESPACE_NAME, BLOCKS, AUTOEXTENSIBLE, MAXBLOCKS, INCREMENT_BY) Affiche tous les fichiers de la base de données
2. **dba_tablespaces** (TABLESPACE_NAME, STATUS(ONLINE,OFFLINE,READ ONLY), LOGGING(LOGGING, NOLOGGING), BIGFILE(YES, NO)) Affiche tous les tablespace de la base de données
3. **user_tablespaces** contient les même champs que la vue précédente mais affiche les tablespace accessibles par l'utilisateur courant



Commandes SQL pour la gestion des structures de stockage

MODIFICATION ET SUPPRESSION DE TABLESPACE:

1. ALTER TABLESPACE 'TABLESPACE_NAME' [OFFLINE|ONLINE|READ ONLY]
2. DROP TABLESPACE 'TABLESPACE_NAME' INCLUDING CONTENTS AND DATAFILES Cette commande supprime le tablespace en question avec tout ce qui est lié
3. ALTER DATABASE DATAFILE 'FILE_NAME' [ONLINE|OFFLINE] Cette commande rend le fichier en question accessible ou pas
4. ALTER DATABASE DATAFILE 'FILE_NAME' RESIZE value, cette commande donne une nouvelle valeur pour la taille d'un fichier
5. ALTER TABLESPACE 'TABLESPACE_NAME' ADD DATAFILE 'FILE_NAME' SIZE 'VALUE' Cette commande ajoute un fichier au tablespace ceci impose que ce dernier soit un SMALLFILE

Commandes SQL pour la gestion des structures de stockage

MODIFICATION ET SUPPRESSION DE TABLESPACE:

1. ALTER DATABASE DATAFILE 'FILE_NAME' AUTOEXTEND [OFF | ON NEXT VALUE MAXSIZE VALUE]
2. ALTER TABLESPACE 'TABLESPACE_NAME' RESIZE 'VALUE' Cette commande redimensionne le fichier du tablespace
3. ALTER TABLESPACE 'TABLESPACE_NAME' AUTOEXTEND ON NEXT 'VALUE' Cette commande rend le fichier En autoextend on
4. ALTER TABLESPACE 'TABLESPACE_NAME' AUTOEXTEND OFF Cette commande rend le fichier En autoextend on

Commandes SQL pour la gestion des structures de stockage

AUGMENTER LA TAILLE DE TABLESPACE:

1. Ajouter un fichier au tablespace en question s'il est SMALLFILE

```
ALTER TABLESPACE tbs ADD DATAFILE '/oracle/oradata5/fich.dbf' SIZE 2G
```

2. Augmenter la taille du fichier déjà utilisé par le tablespace en question

```
ALTER DATABASE DATAFILE '/oracle/oradata2/fich.dbf' resize 5G;
```

3. Activer l'option autoextend ON

```
ALTER DATABASE DATAFILE '/oracle/oradata2/fich.dbf' AUTOEXTEND ON NEXT 512M MAXSIZE 5G
```

SCENARIO DE TEST

2. Affichage des tablespaces de cette base:

```
SQL> select tablespace_name, status, logging, bigfile from dba_tablespaces;
```

TABLESPACE_NAME	STATUS	LOGGING	BIG
SYSTEM	ONLINE	LOGGING	NO
SYSAUX	ONLINE	LOGGING	NO
UNDOTBS1	ONLINE	LOGGING	NO
TEMP	ONLINE	NOLOGGING	NO
USERS	ONLINE	LOGGING	NO

```
SQL>
```


SCENARIO DE TEST

3. Affichage des fichiers correspondant à ces tablespaces:

```
SQL> select tablespace_name, file_name from dba_data_files;
```

```
TABLESPACE_NAME
```

```
-----
```

```
FILE_NAME
```

```
-----
```

```
USERS
```

```
C:\ORACLEXE\APP\ORACLE\ORADATA\XE\USERS.DBF
```

```
SYSAUX
```

```
C:\ORACLEXE\APP\ORACLE\ORADATA\XE\SYSAUX.DBF
```

```
UNDOTBS1
```

```
C:\ORACLEXE\APP\ORACLE\ORADATA\XE\UNDOTBS1.DBF
```

```
SYSTEM
```

```
C:\ORACLEXE\APP\ORACLE\ORADATA\XE\SYSTEM.DBF
```

SCENARIO DE TEST

4. Ajout de tablespaces :

```
SQL> create tablespace app_test datafile 'C:/ORACLEXE/APP/ORACLE/ORADATA/XE/app_test.dbf' size 512M autoextend off;  
Tablespace created.
```

5. Vérification:

```
SQL> select tablespace_name, file_name, bytes/1024/1024 from dba_data_files;  
TABLESPACE_NAME  
FILE_NAME  
-----  
BYTES/1024/1024  
-----  
USERS  
C:\ORACLEXE\APP\ORACLE\ORADATA\XE\USERS.DBF  
100  
SYSaux  
C:\ORACLEXE\APP\ORACLE\ORADATA\XE\sysaux.DBF  
660  
TABLESPACE_NAME  
FILE_NAME  
-----  
BYTES/1024/1024  
-----  
UNDOTBS1  
C:\ORACLEXE\APP\ORACLE\ORADATA\XE\UNDOTBS1.DBF  
380  
SYSTEM  
C:\ORACLEXE\APP\ORACLE\ORADATA\XE\SYSTEM.DBF  
TABLESPACE_NAME  
FILE_NAME  
-----  
BYTES/1024/1024  
-----  
360  
APP_TEST  
C:\ORACLEXE\APP\ORACLE\ORADATA\XE\APP_TEST.DBF  
512
```

SCENARIO DE TEST

6. Ajout de tablespaces BIGFILE :

```
SQL> create BIGFILE tablespace app_big datafile 'C:/ORACLEXE/APP/ORACLE/ORADATA/XE/app_big.dbf' size 512M autoextend off;  
Tablespace created.
```

7. Ajout de fichiers pour les deux tablespaces:

```
SQL> alter tablespace app_test add datafile 'C:/ORACLEXE/APP/ORACLE/ORADATA/XE/APP_TEST2.DBF' size 128M;  
Tablespace altered.  
  
SQL> alter tablespace app_big add datafile 'C:/ORACLEXE/APP/ORACLE/ORADATA/XE/APP_BIG2.DBF' size 128M;  
alter tablespace app_big add datafile 'C:/ORACLEXE/APP/ORACLE/ORADATA/XE/APP_BIG2.DBF' size 128M  
*  
ERROR at line 1:  
ORA-32771: cannot add file to bigfile tablespace
```

SCENARIO DE TEST

8. Vérifier l'état des tablespaces :

```
SQL> select tablespace_name, bigfile from dba_tablespaces;
```

TABLESPACE_NAME	BIG
SYSTEM	NO
SYSAUX	NO
UNDOTBS1	NO
TEMP	NO
USERS	NO
APP_TEST	NO
APP_BIG	YES

```
7 rows selected.
```

9. Redimensionner la taille d'un fichier:

```
SQL> alter database datafile 'C:\ORACLEXE\APP\ORACLE\ORADATA\XE\APP_TEST2.DBF' r  
esize 800M;
```

```
Database altered.
```

```
SQL> select file_name,bytes/1024/1024 from dba_data_files;
```

```
FILE_NAME
```

```
BYTES/1024/1024
```

```
C:\ORACLEXE\APP\ORACLE\ORADATA\XE\USERS.DBF  
100
```

```
C:\ORACLEXE\APP\ORACLE\ORADATA\XE\SYSAUX.DBF  
660
```

```
C:\ORACLEXE\APP\ORACLE\ORADATA\XE\UNDOTBS1.DBF  
380
```

```
FILE_NAME
```

```
BYTES/1024/1024
```

```
C:\ORACLEXE\APP\ORACLE\ORADATA\XE\SYSTEM.DBF  
360
```

```
C:\ORACLEXE\APP\ORACLE\ORADATA\XE\APP_TEST.DBF  
512
```

```
C:\ORACLEXE\APP\ORACLE\ORADATA\XE\APP_BIG.DBF  
512
```

```
FILE_NAME
```

```
BYTES/1024/1024
```

```
C:\ORACLEXE\APP\ORACLE\ORADATA\XE\APP_TEST2.DBF  
800
```

```
7 rows selected.
```

SCENARIO DE TEST

10. Vérifier le statut des tablespaces :

```
SQL> select tablespace_name, status from dba_tablespaces;

TABLESPACE_NAME          STATUS
-----
SYSTEM                   ONLINE
SYSaux                   ONLINE
UNDOTBS1                 ONLINE
TEMP                     ONLINE
USERS                    ONLINE
APP_TEST                 ONLINE
APP_BIG                  ONLINE

7 rows selected.
```

11. Changer le statut d'un tablespace à offline:

```
SQL> alter tablespace APP_TEST OFFLINE;

Tablespace altered.

SQL> create table test(chmp int) tablespace APP_TEST;
create table test(chmp int) tablespace APP_TEST
*
ERROR at line 1:
ORA-01542: tablespace 'APP_TEST' is offline, cannot allocate space in it
```

Parmi les autres effets : même les objet de ce tablespace ne seront pas accessible

SCENARIO DE TEST

12. Effet du statut READ ONLY :

```
SQL> create table test(chmp int) tablespace APP_BIG;
Table created.

SQL> alter tablespace APP_BIG read only;
Tablespace altered.

SQL> insert into test values (15);
insert into test values (15)
      *
ERROR at line 1:
ORA-00372: file 6 cannot be modified at this time
ORA-01110: data file 6: 'C:\ORACLEXE\APP\ORACLE\ORADATA\XE\APP_BIG.DBF'
```