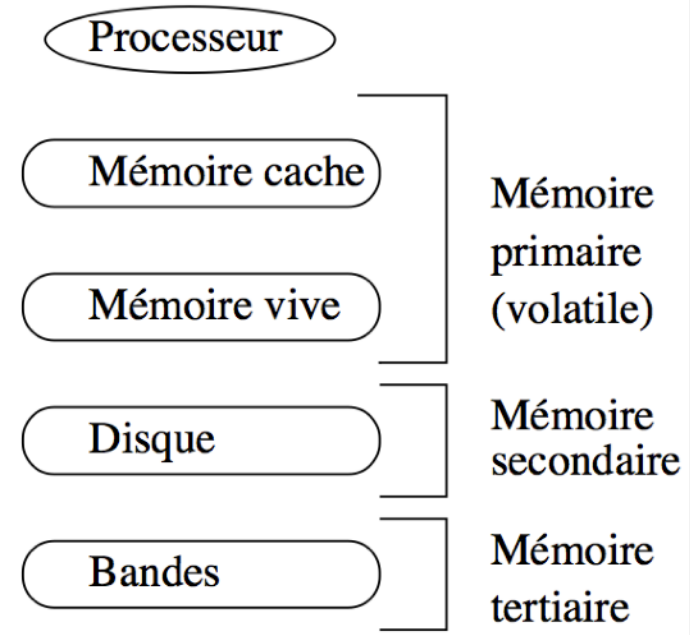
Stockage de données

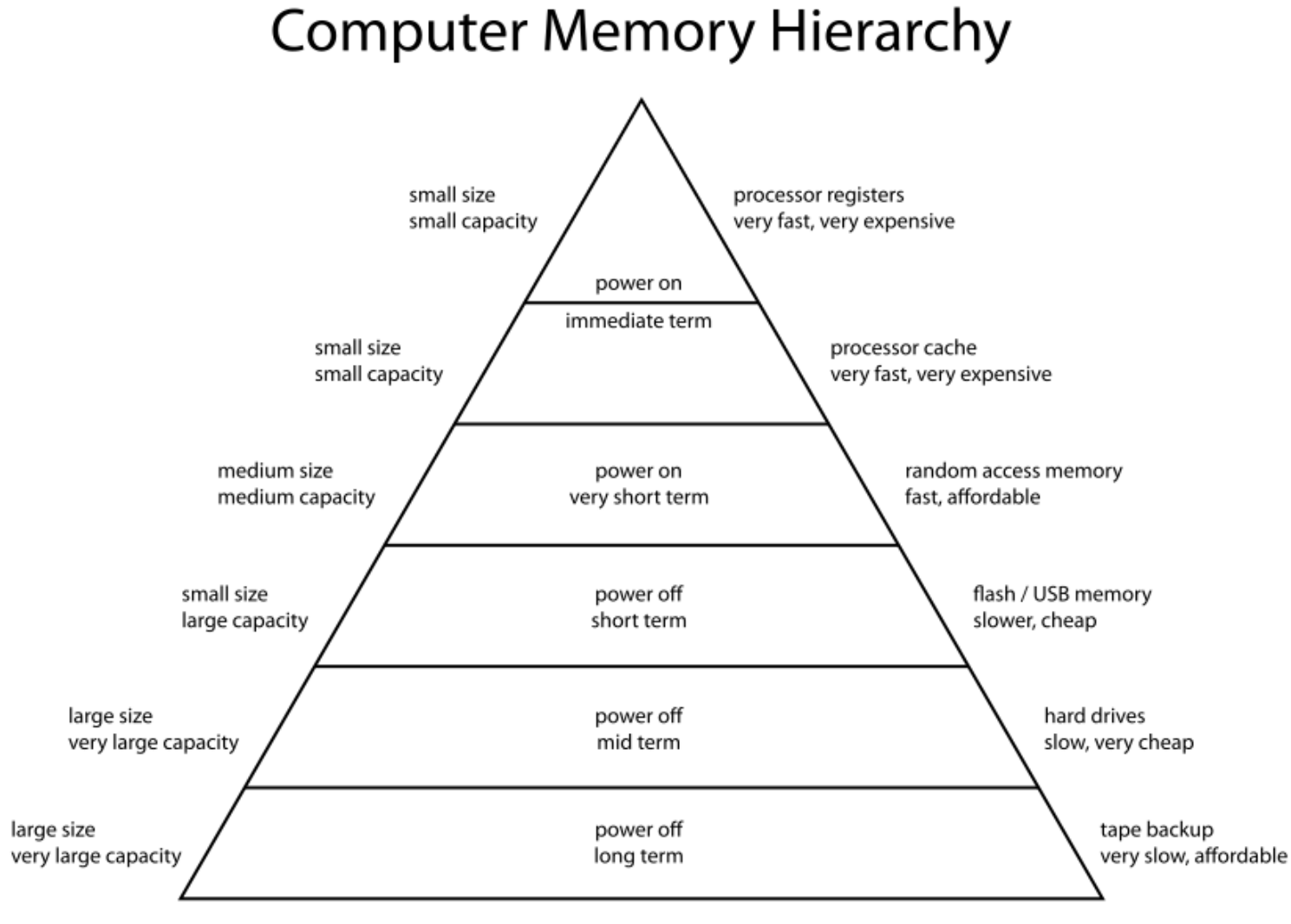
Patricia Serrano Alvarado
D'après les slides de
Sylvie Cazalens et Philippe Rigaux

Les mémoires d'un ordinateur

- Hiérarchie des mémoires



- Plus la mémoire est petite plus elle est rapide



- Un accès disque est (environ) un million de fois plus coûteux (en temps) qu'un accès en mémoire principale !

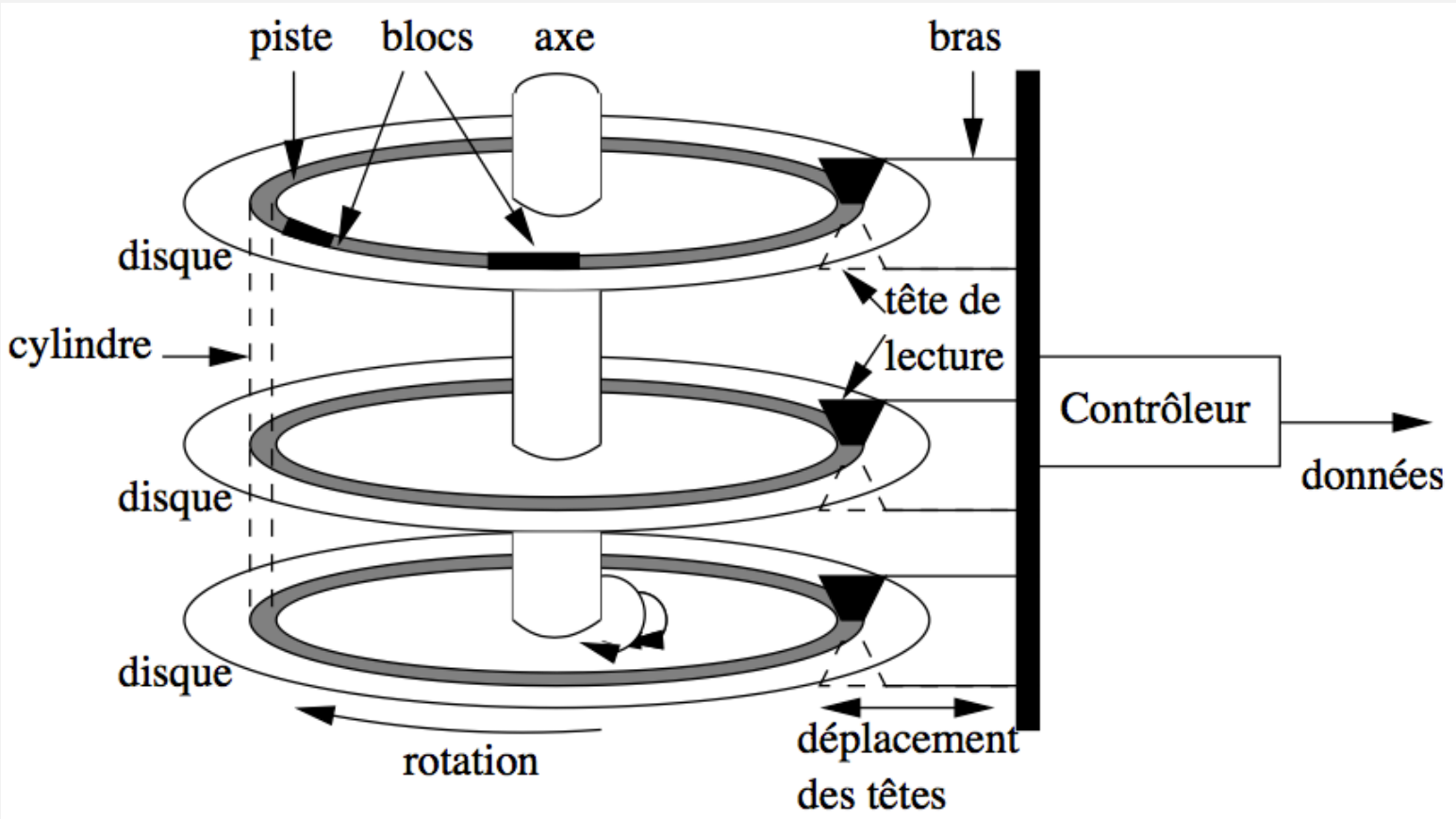
Importance dans les BD

- Un SGBD doit ranger sur disque les données pour accélérer leur accès
  - parce qu'elles sont trop volumineuses
  - parce qu'elles sont persistantes (et doivent survivre à un arrêt du système).
- le SGBD doit toujours amener les données en mémoire primaire pour les traiter ;
  - si possible, les données utiles devraient résider le plus possible en mémoire primaire.
- La capacité à gérer efficacement les transferts mémoire primaire-secondaire est un facteur important de performance d'une application bases de données.

Organisation d'un disque

- Disque : surface magnétique, stockant des 0 ou des 1, divisé en secteurs
  - Un **secteur** est l'unité de stockage la plus petite, pour un disque la taille d'un secteur est en générale de 512 octets.
- Dispositif : les surfaces sont entrainées dans un mouvement de rotation ; les têtes de lecture se déplacent dans un plan fixe.
  - le **bloc** est un ensemble de secteurs (page: nb fixe de blocs contigus, possibilité de taille de 4 Ko à 256 Mo)
  - la **piste/track** est l'ensemble des blocs d'une surface lus au cours d'une rotation ;
  - le **cylindre** est un ensemble de pistes situées sous les têtes de lecture.

Structure d'un disque



Exemple de disque

- Le Megatron 747
  - 8 disques donc 16 faces
  - 2^{16} ou 65,536 pistes par face
  - En moyenne 2^{8} ou 256 secteurs par piste
  - 2^{12} ou 4096 octets par secteur
- Cela fait une capacité d'environ 1 teraoctet
  - $16*65,536*256*4096=1,099,511,627,776 = 2^{40} = 10^{12}$

Disque=mémoire d'accès

- Adresse = numéro du disque ; de la piste où se trouve le bloc ; du numéro du bloc sur la piste.
  - **délai de positionnement** pour placer la tête sur la bonne piste
  - **délai de latence** pour attendre que le bloc passe sous la tête de lecture
  - **temps de transfert** pour attendre que le (ou les) bloc(s) soient lus et transférés.
- Important : on lit toujours au moins un bloc, même si on ne veut qu'un octet ; c'est l'unité d'entrée/sortie !
- ***Principe de localité* :** si deux données sont « proches » du point de vue applicatif, alors elles doivent être proches sur le disque (idéalement, dans le même bloc).

Le principe de localité et ses conséquences

- Principe de localité : l'ensemble des données utilisées par une application pendant une période donnée forme souvent un groupe bien identifié.

1. **Localité spatiale** : si une donnée d est utilisée, les données proches de d ont de fortes chances de l'être également ;

2. **Localité temporelle** : quand une application accède à une donnée d, il y a de fortes chances qu'elle y accède à nouveau peu de temps après.

3. **Localité de référence** : si une donnée d1 référence une donnée d2, l'accès à d1 entraîne souvent l'accès à d2.

- Les systèmes exploitent ce principe en déplaçant dans la hiérarchie des mémoires des groupes de données proches de la donnée utilisée à un instant t
  ⇒ le pari est que l'application accèdera à d'autres données de ce groupe.

Composition d'enregistrement

- Entête : région de valeur fixe avec information sur l'enregistrement
  - Pointeur vers le schéma de l'enregistrement stocké
  - La taille de l'enregistrement
  - Timestamps (dernière modification ou lecture) important pour les transactions
- Valeurs : les valeurs de l'enregistrement
  - Pratique si la taille des attributs est en multiples de 4 ou de 8

Un enregistrement

```
CREATE TABLE MovieStar(
Name CHAR(30) PRIMARY KEY,
Address VARCHAR(255),
Gender CHAR(1),
Birthdate DATE);
```

- Entête
  - Pointeur : 4 octets ; taille : 4 octets ; timestamp : 4 octets
- Valeurs :
  - Name : 30 + 2 octets (pour que ça soit en multiples de 4)
  - Address : 255 + 1
  - Gender : 1+3
  - Birthdate : chaine de caractères de 10 octets + 2
- Cela fait une taille d'enregistrement de 316 octets

Enregistrements de taille variable ?

- Oui car
  - Ca permet d'économiser de l'espace de stockage
  - Si valeurs NULL
  - Si attributs de taille inconnue (ex. éléments xml)
  - Attributs de grande taille (attributs pour stocker de vidéos)
- Entête doit avoir
  - Taille réelle de l'enregistrement
  - Pointeurs vers le début des attributs de taille variable

Blocs et enregistrements

- Un bloc contient
  - Une entête
    - Liens vers les blocs qui font partie d'un réseau utiles pour l'indexation
    - Information sur le rôle du bloc dans ce réseau
    - Information sur la table des enregistrements du bloc
    - Répertoire donnant le pointeur de chaque enregistrement dans le bloc
    - Timestamp de la dernière modification et/ou accès
    - Quantité d'espace libre
  - Les enregistrements
    - En général un enregistrement doit être dans un seul bloc

Manipulation d'enregistrements

- On considère qu'il n'y pas d'index (les enregistrement ne suivent pas un ordre au moment du stockage)
- Insertion
  - On cherche un bloc avec espace vide ou on obtient un nouveau bloc
  - On ajoute l'enregistrement
- Modification
  - Si cela entraine une modification de la taille de l'enregistrement alors il faut une réorganisation des enregistrements dans le bloc
- Recherche
  - Lecture séquentielle des enregistrements du bloc

Stockage des données dans Oracle

- Les données relatives à une base sont stockées dans des fichiers (structure physique). On peut spécifier les fichiers, leur taille, etc.
- La base de données physique est constituée de plusieurs fichiers, de types différents.

Les fichiers physiques 1/2

- Fichiers de données - data files
  - données utilisateurs
  - structures logiques pour gestion de la base (e.g., indexes)
- Fichiers de reprise - redo log files
  - traces des dernières modifications
- Fichiers de contrôle - control files
  - infos sur la structure physique de la base, indispensable au démarrage (état de la base, cohérence...)

Les fichiers physiques 2/2

- Fichier de mots de passe
  - authentification des utilisateurs
- Fichier de paramètres
  - paramètres de démarrage de la base et de définition de l'environnement
- Fichiers reprise archivés
  - copie de fichiers de reprise avant réutilisation.

Fichiers de données

- Taille modifiable après création
- Les données peuvent être séparées ou regroupées physiquement en plaçant les fichiers sur des disques différents (mais problème d'optimisation)

Fichiers Redo Log

- Journal de toutes les transactions qui ont eu lieu dans la base, pour rétablir les transactions dans leur ordre correct en cas de problème de la base.
- Ne se stockent pas au même endroit que les fichiers de données.
- Il y en a au minimum 3, où Oracle écrit de manière cyclique.

Fichiers de contrôle

- Associés à une seule base de données. Fichiers binaires nécessaires au démarrage et au fonctionnement de la base (maintien de l'architecture physique).
- Contient toutes les informations sur les autres fichiers de la base. Il est continuellement mis à jour (ajout/retrait d'un fichier de données ou redo).
- Contient : le nom de la BD, nom et situation des différents fichiers, information sur les tablespaces,...

Tablespace

- L’utilisateur n'a pas à raisonner avec les fichiers. C'est la notion de Tablespace qui fait le lien entre les objets du schéma (tables, vues, index..) et les fichiers.
- Les tablespaces, ou espaces logiques (structure logique) permettent d'associer données et fichiers. Il est « plus facile » de raisonner en termes de tablespaces.

Tablespace

- Assimilable à un espace disque logique
- Regroupe des objets ayant des caractéristiques de stockage identiques.
- Une base de données est composée d'un ou plusieurs tablespaces ; on peut en rajouter.
- Un tablespace n'appartient qu'à une seule base de données.

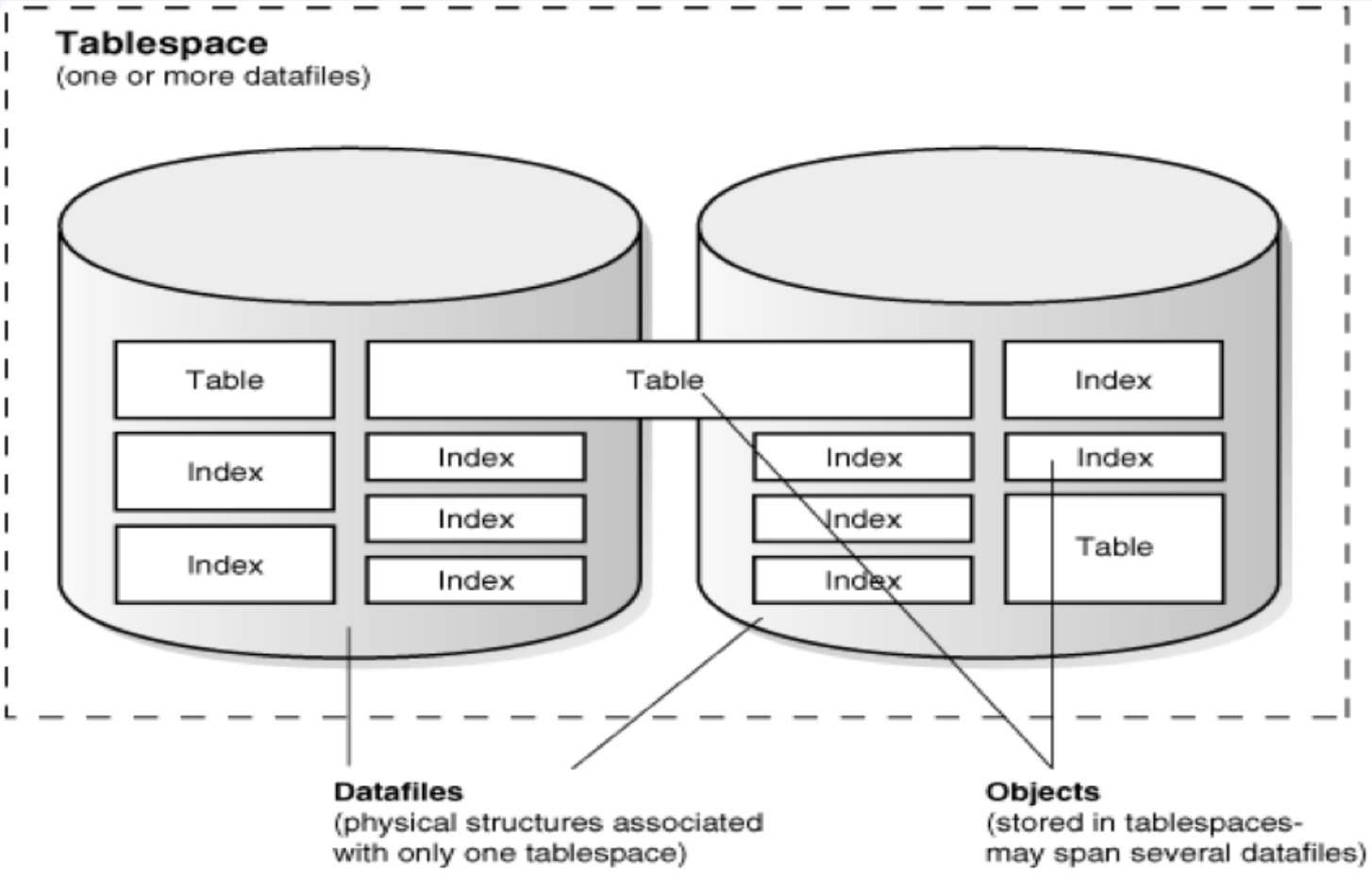
Tablespace

- À un tablespace correspondent (un ou) plusieurs fichiers physiques ; on peut en rajouter.
- Toute table est créée dans un tablespace. Ses données et indexes ne peuvent s'étendre à un autre tablespace.
- Un fichier de données ne peut appartenir qu'à un seul tablespace. Ne peut être supprimé ni ajouté à un autre tablespace.

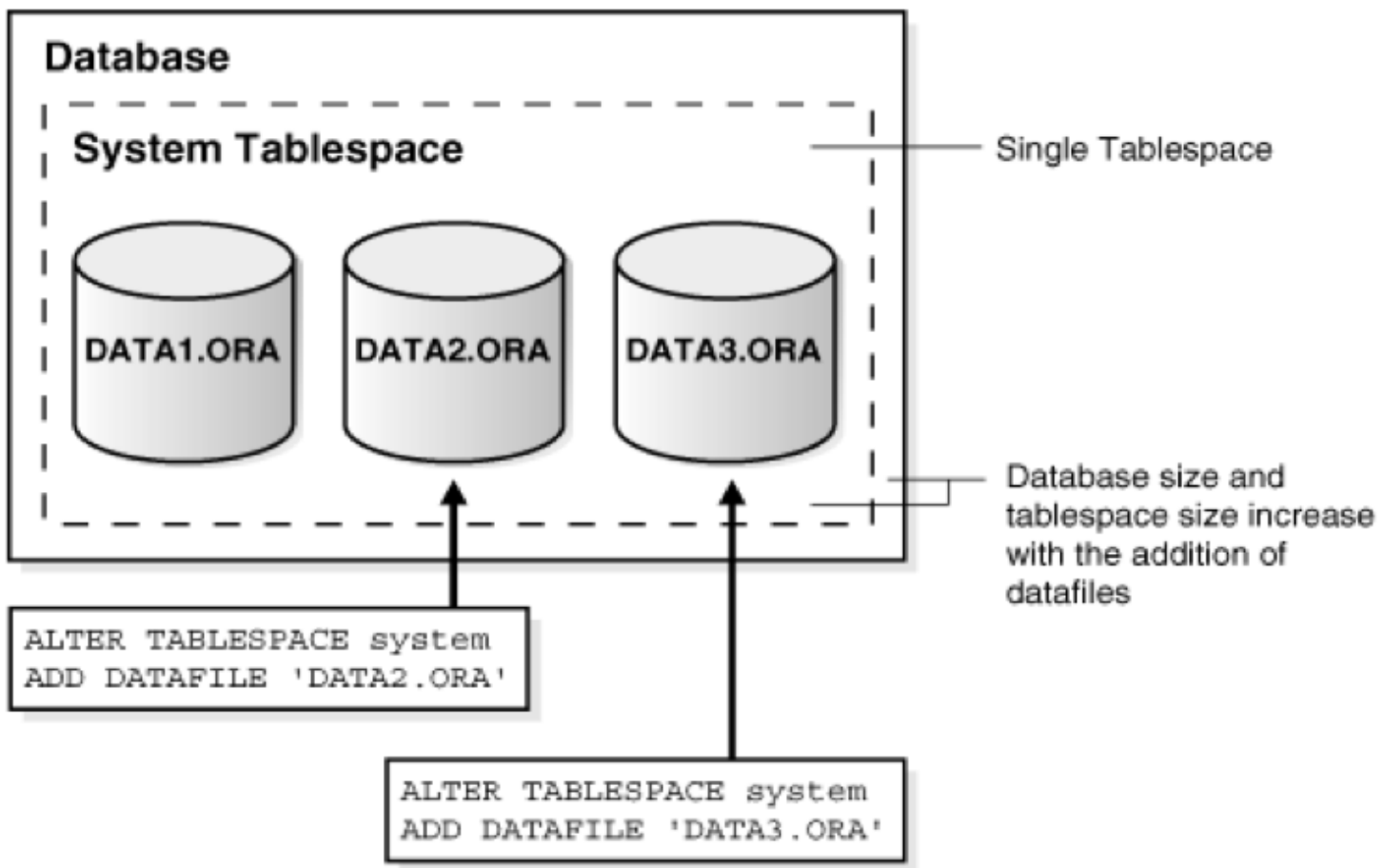
Tablespaces SYSTEM et USER

- Créés pour chaque BD
- SYSTEM : contient toujours les tables du dictionnaire des données de la base entière et toutes les informations relatives aux programmes PL/SQL
  - C'est au DBA de prévoir assez de place.
- USERS : pour les objets des utilisateurs (tables, indexes, vues...).

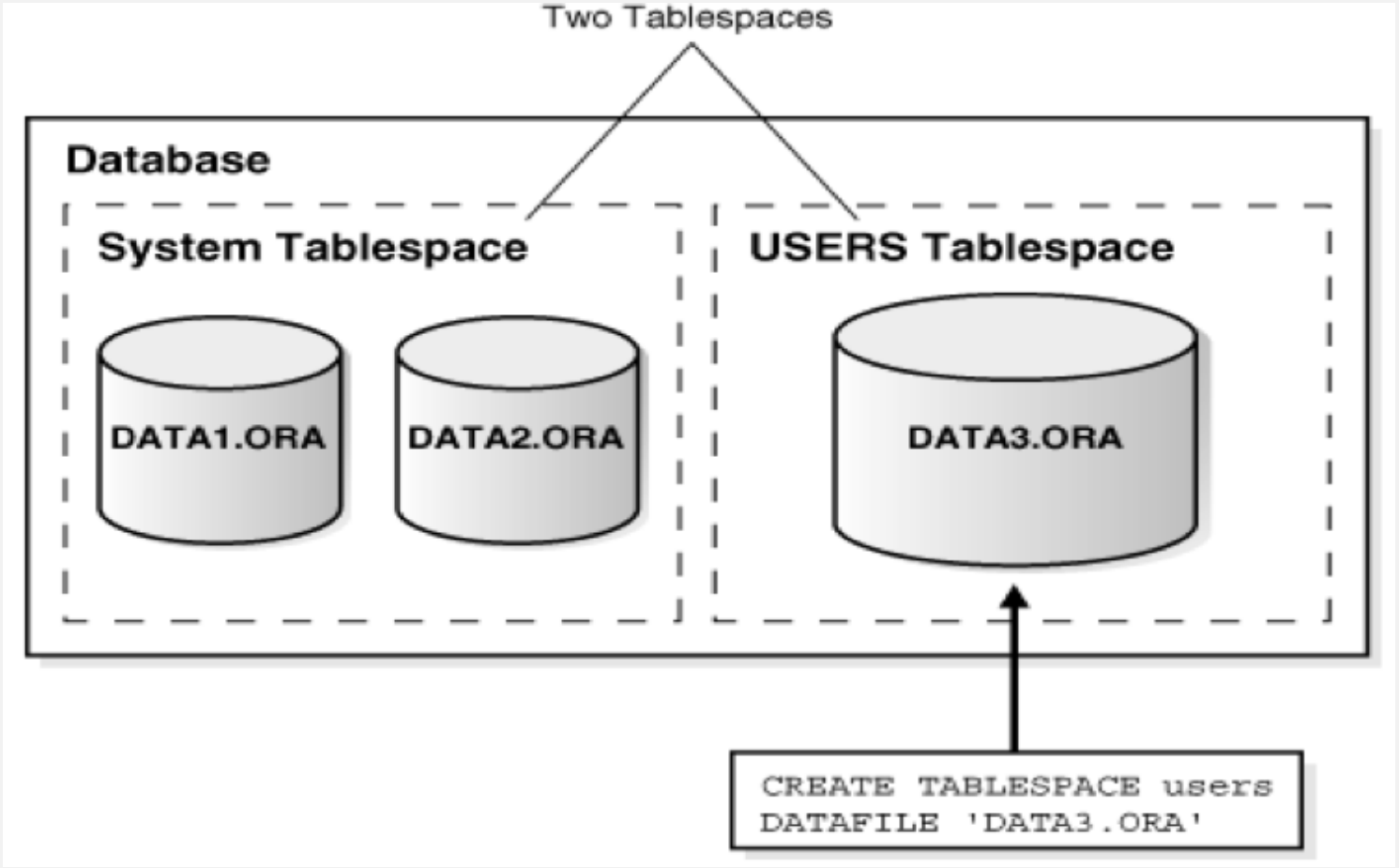
Tablespace et fichiers de données : exo



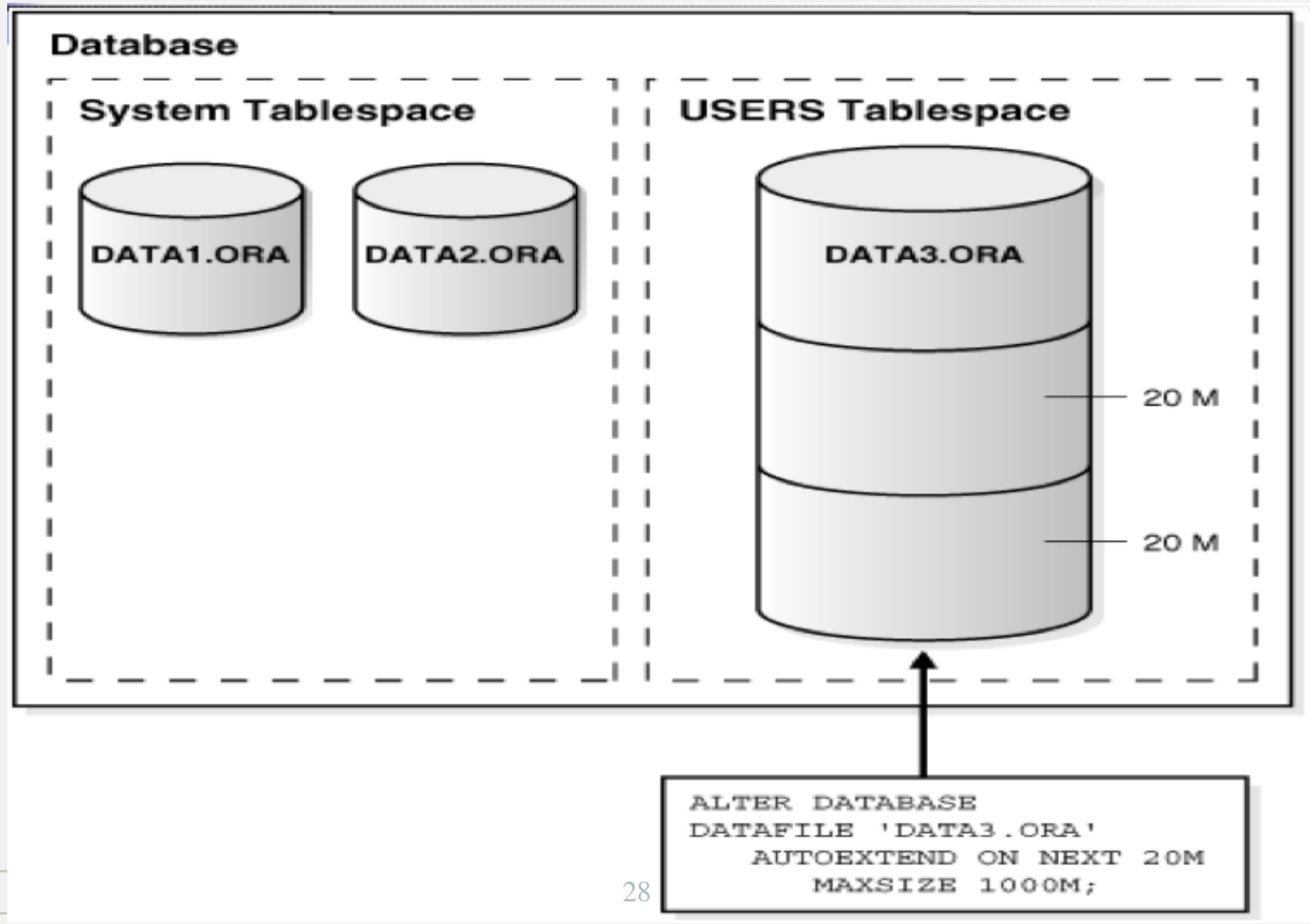
Ajout d'un fichier de données



Ajout d'un tablespace



Modif. de la taille d'un fichier



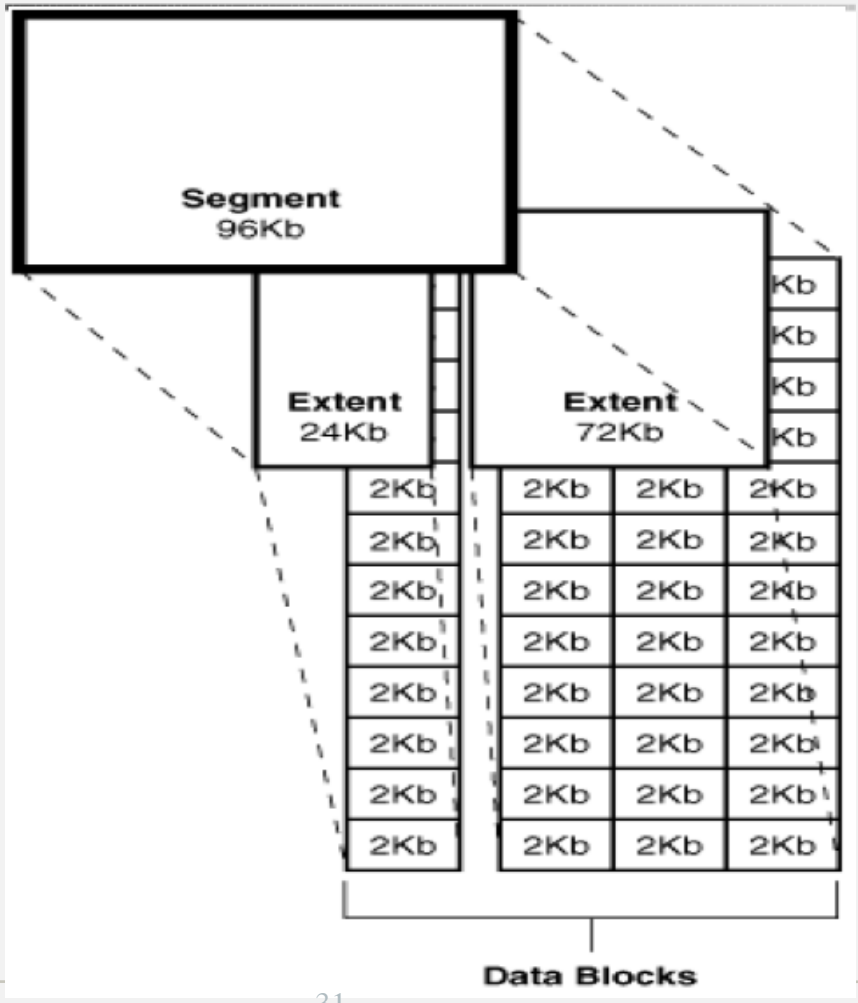
Taille de la base de données

- Taille d'un tablespace = somme des tailles de tous les fichiers associés à ce tablespace.
- Taille d'une base de données = somme des tailles de tous les tablespaces de la base

Gestion des disques

- Bloc : un certain nombre d'octets ; unité logique de transfert entre disque et mémoire centrale. La taille d'un bloc ORACLE est un multiple de la taille des blocs du système
- Extension : un certain nombre de blocs consécutifs alloués simultanément à un objet de schéma.
- Segment : un ensemble d'extensions allouées au même objet.

Segment, extension, bloc



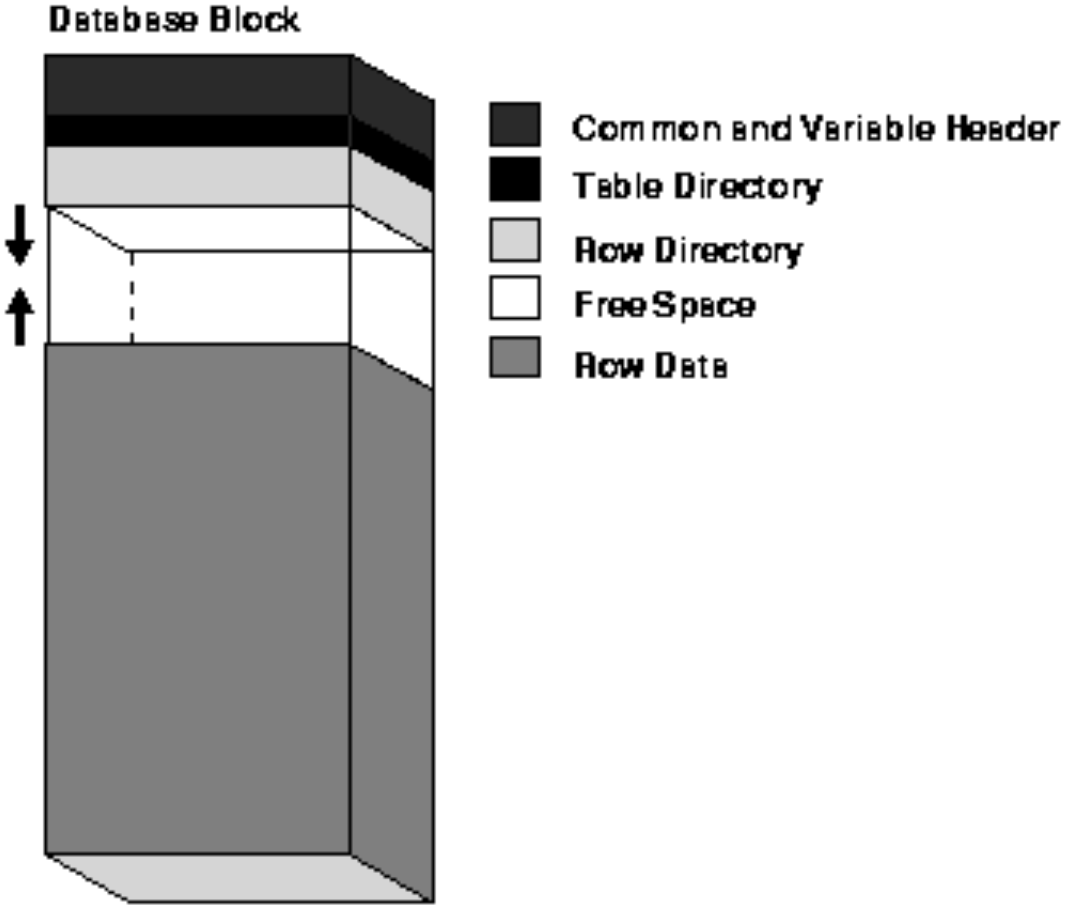
Segments

- Plusieurs types
  - Le segment de données.
  - Le segment d'index.
  - Le *rollback segment* utilisé pour les transactions.
  - Le segment temporaire (utilisé pour les tris).
- Moins il y a d'extensions dans un segment, plus il est efficace
- Relatif à un seul tablespace
- Peut s'étendre sur plusieurs fichiers

Blocs

- Taille d'un bloc (DB_BLOCK_SIZE, fichier init.ora, multiple de la taille des blocs manipulés par le système d'exploitation.)
- Organisation identique (indépendant du type données)
  - En-tête (partie fixe + partie variable : adresse du bloc, type de segment (données, index))
  - Répertoire des tables ayant des lignes dans le bloc
  - Répertoire des lignes contenues dans le bloc
  - Espace libre : zone de débordement pour les mises à jour du bloc
  - Espace des lignes : les données ou indexes stockés

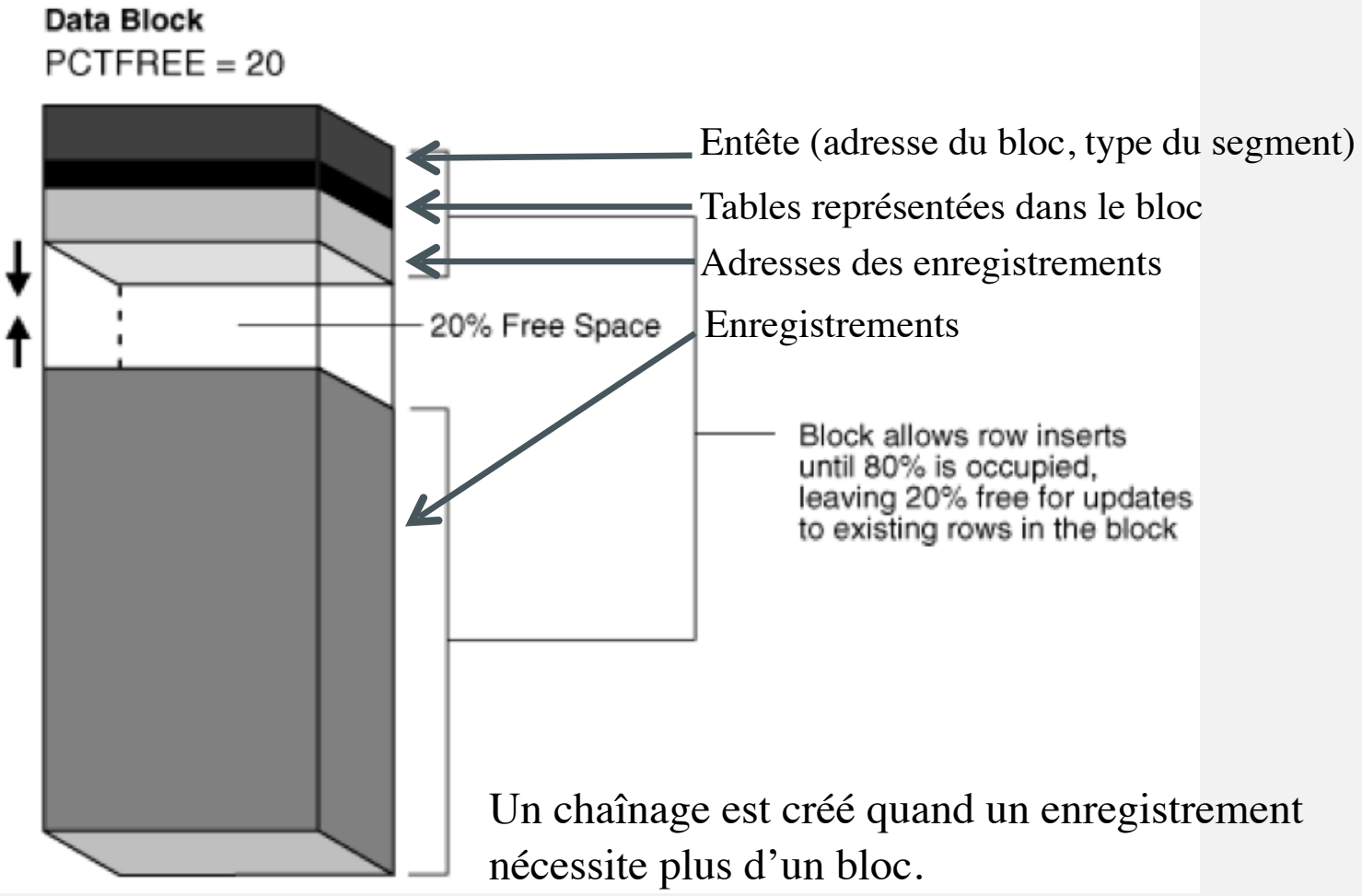
Bloc


Database Block
Common and Variable Header
Table Directory
Row Directory
Free Space
Row Data

Remplissage d'un bloc

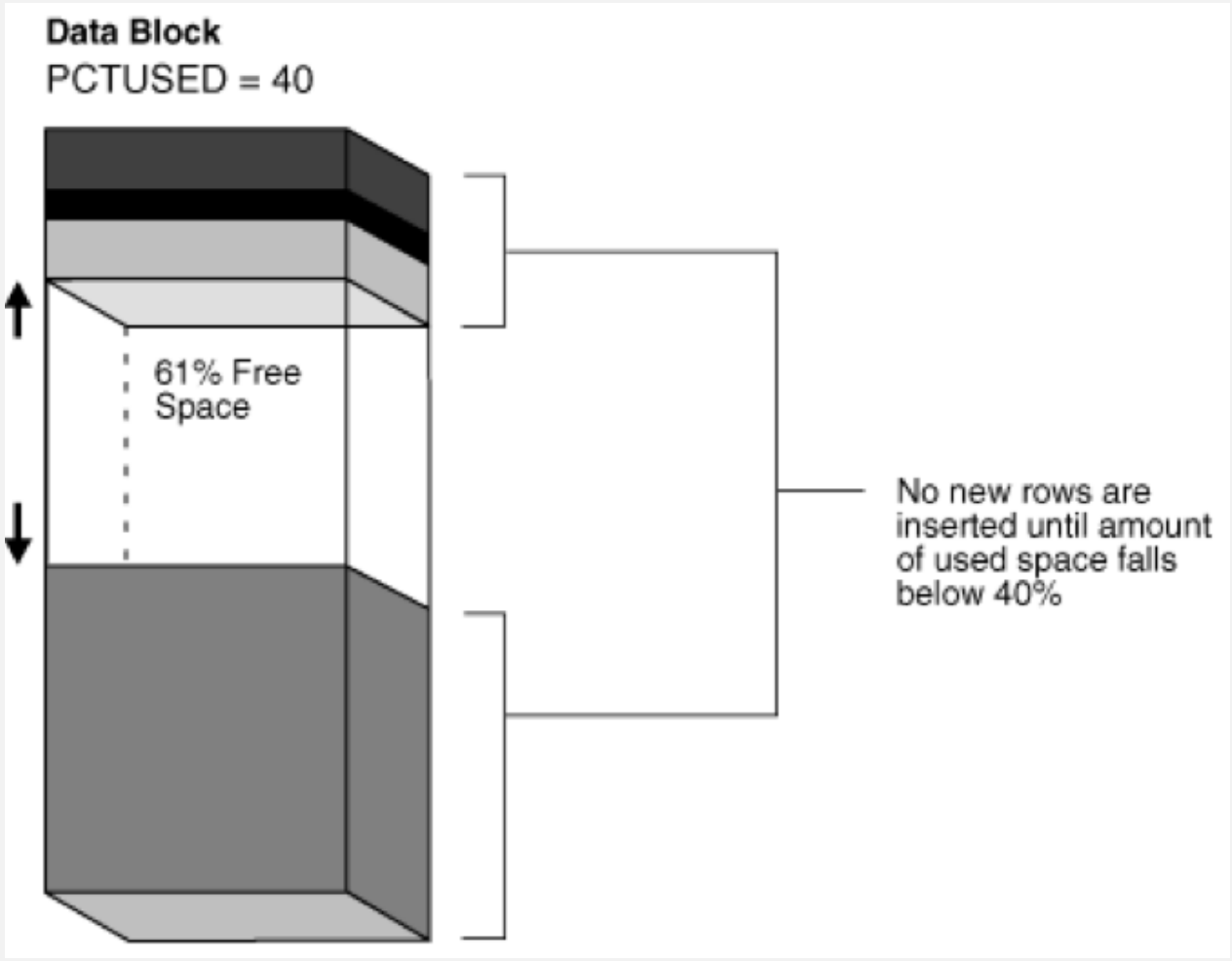
- Deux paramètres pour améliorer
  - les perfs des écritures/lectures
  - réduire l'espace inutilisé
  - réduire le nombre de chainage de lignes entre les blocs
- PCTFREE : pourcentage minimum d'espace libre dans un bloc, pour les modifs ultérieures. Défaut 10%
  - Si seuil atteint uniquement possibilité de *udaptes* et *deletes*
  - Cela jusqu'à que le % utilisé < % PCTUSED
- PCTUSED : pourcentage d'occupation maximum autorisé pour que l'on puisse insérer **à nouveau** de nouvelles lignes. Défaut 40%
  - Si seuil atteint plus de possibilité d'insertion de nouvelles lignes

Paramètre PCTFREE



Un chaînage est créé quand un enregistrement nécessite plus d'un bloc.

Paramètre PCTUSED



Connaître les paramètres PCTFREE et PCTUSED

- Par défaut PCTFREE =10 et PCTUSED = 40

```
SELECT pct_free
FROM user_tables
WHERE table_name = 'EMP';
```

USER_TABLES

```
DESC USER_TABLES;
SELECT pct_free
FROM user_tables
WHERE table_name = 'EMP';
```

```
Name                  Null     Type
--------------------- -------- ------------
TABLE_NAME            NOT NULL VARCHAR2(30)
TABLESPACE_NAME                VARCHAR2(30)
CLUSTER_NAME                   VARCHAR2(30)
IOT_NAME                       VARCHAR2(30)
STATUS                         VARCHAR2(8)
PCT_FREE                       NUMBER
PCT_USED                       NUMBER
INI_TRANS                      NUMBER
MAX_TRANS                      NUMBER
INITIAL_EXTENT                 NUMBER
NEXT_EXTENT                    NUMBER
MIN_EXTENTS                    NUMBER
MAX_EXTENTS                    NUMBER
PCT_INCREASE                   NUMBER
FREELISTS                      NUMBER
FREELIST_GROUPS                NUMBER
LOGGING                        VARCHAR2(3)
BACKED_UP                      VARCHAR2(1)
NUM_ROWS                       NUMBER
BLOCKS                         NUMBER
EMPTY_BLOCKS                   NUMBER
AVG_SPACE                      NUMBER
CHAIN_CNT                      NUMBER
AVG_ROW_LEN                    NUMBER
AVG_SPACE_FREELIST_BLOCKS      NUMBER
```

```
Name                  Null     Type
--------------------- -------- ------------
NUM_FREELIST_BLOCKS            NUMBER
DEGREE                         VARCHAR2(10)
INSTANCES                      VARCHAR2(10)
CACHE                          VARCHAR2(5)
TABLE_LOCK                     VARCHAR2(8)
SAMPLE_SIZE                    NUMBER
LAST_ANALYZED                  DATE
PARTITIONED                    VARCHAR2(3)
IOT_TYPE                       VARCHAR2(12)
TEMPORARY                      VARCHAR2(1)
SECONDARY                      VARCHAR2(1)
NESTED                         VARCHAR2(3)
BUFFER_POOL                    VARCHAR2(7)
FLASH_CACHE                    VARCHAR2(7)
CELL_FLASH_CACHE               VARCHAR2(7)
ROW_MOVEMENT                   VARCHAR2(8)
GLOBAL_STATS                   VARCHAR2(3)
USER_STATS                     VARCHAR2(3)
DURATION                       VARCHAR2(15)
SKIP_CORRUPT                   VARCHAR2(8)
MONITORING                     VARCHAR2(3)
CLUSTER_OWNER                  VARCHAR2(30)
DEPENDENCIES                   VARCHAR2(8)
COMPRESSION                    VARCHAR2(8)
COMPRESS_FOR                   VARCHAR2(12)
DROPPED                        VARCHAR2(3)
READ_ONLY                      VARCHAR2(3)
SEGMENT_CREATED                VARCHAR2(3)
RESULT_CACHE                   VARCHAR2(7)
```

Insertion/suppression de lignes

- Insertion : de manière séquentielle dans un bloc. Lorsque plus de place, choix d'un autre bloc dans la liste chainée.
  - En générale, un enregistrement est stocké dans un seul bloc. L'adresse physique d'un enregistrement est le ***ROWID*** :
    - Le numéro du bloc dans le **fichier**.
    - Le numéro du n-uplet dans la block (ou page).
    - Le numéro du fichier.

  Exemple : 00000DD5.001.001 est l'adresse du premier n-uplet du bloc DD5 dans le premier fichier.

- Modification : si taille non modifiée, pas de changement de bloc ; si taille augmente, reconstruction dans le même bloc ou divisée en plusieurs morceaux répartis sur plusieurs blocs ; si taille diminue, création d'un espace disponible (après validation transaction)

Gestion de l'espace physique pour objet de schéma

- A la création : une extension est réservée pour l'objet. Lorsque tous ses blocs sont pleins, allocation d'une nouvelle extension
- Gestion par différents paramètres :
  - INITIAL : taille en octets allouée au 1er segment,
  - NEXT, PCTINCREASE : taille de chaque nouveau segment, pourcentage d'accroissement
  - MINEXTENTS : nombre d'extensions allouées à la création
  - MAXEXTENTS : nombre maximum d'extensions.

Création d'une table : syntaxe (incomplète)

create table <nom_table>

(<description des colonnes et des contraintes>)

[pctfree <val>] [pctused <val>]

[tablespace <nom_tbsp>]

[storage (INITIAL <val>

NEXT <val>

MAXEXTENTS <val>

MINEXTENTS <val>)]

[cluster] [enablecontr][disablecontr]..;

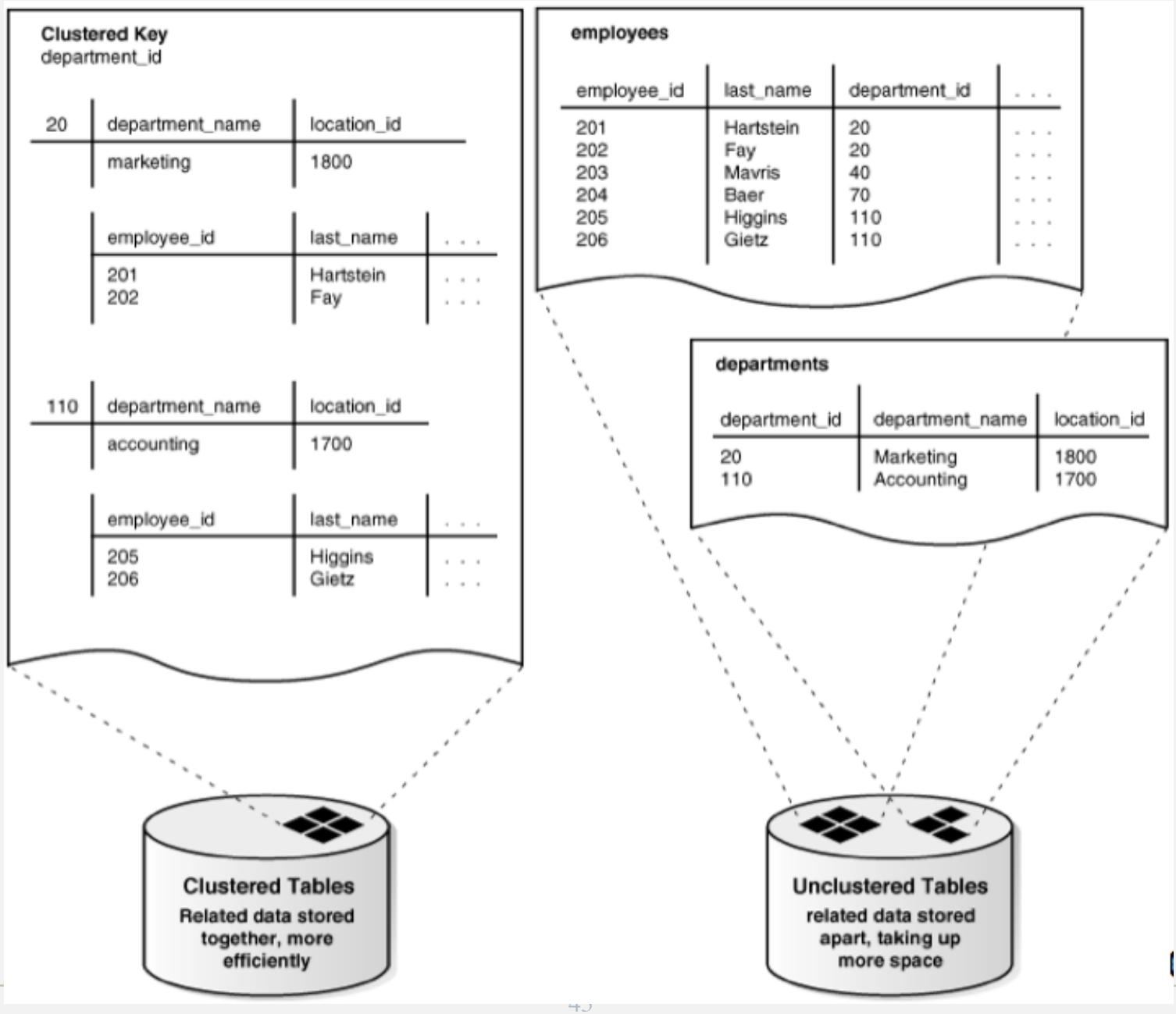
Exo

```
create table etudiants(
numetu varchar2(20) primary key,
nom varchar2(30),
prenom varchar2(30))
PCTFREE 20
PCTUSED 50
TABLESPACE USERS
STORAGE (INITIAL 25K
        NEXT 10K
        MAXEXTENTS 10
        MINEXTENTS 3);
```

- On suppose des insertions jusqu'à 80%
- Alors, possibilité que de suppressions et modifications.
- Est-ce qu'on peut continuer à insérer des lignes si taux de remplissage
  - = 60% ?
  - = 80% ?
  - = 40% ?

Les clusters

- Cluster Oracle : espace dont les blocs peuvent contenir des enregistrements provenant de différentes tables.
- Permet de favoriser le rapprochement physique de données reliées (par exemple, enregistrements qui font souvent l'objet de jointures)


Clustered Key
department_id
20
department_name
location_id
marketing
1800
employee_id
last_name
201
Hartstein
202
Fay
110
department_name
location_id
accounting
1700
employee_id
last_name
205
Higgins
206
Gietz
employees
employee_id
last_name
department_id
201
Hartstein
20
202
Fay
20
203
Mavris
40
204
Baer
70
205
Higgins
110
206
Gietz
110
departments
department_id
department_name
location_id
20
Marketing
1800
110
Accounting
1700
Clustered Tables
Related data stored together, more efficiently
Unclustered Tables
related data stored apart, taking up more space

Création d'un cluster (exo)

CREATE CLUSTER personnel (department NUMBER(4))

SIZE 512 STORAGE (initial 100K next 50K);

Ensuite, les tables concernées peuvent être crées

On peut utiliser ALTER TABLE pour ajouter l'usage d'un cluster

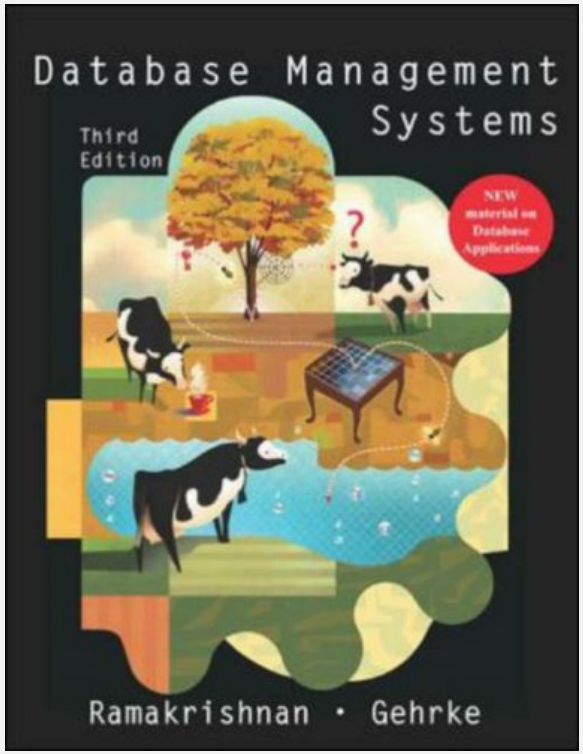
Exo

```
CREATE CLUSTER emp_dept (
deptno NUMBER(3))
  SIZE 600
  TABLESPACE users
  STORAGE (INITIAL 200K
    NEXT 300K
    MINEXTENTS 2
    PCTINCREASE 33);
```

```
CREATE TABLE emp (
  empno NUMBER(5) PRIMARY KEY,
  ename VARCHAR2(15) NOT NULL,
  . . .
  deptno NUMBER(3) REFERENCES dept)
  CLUSTER emp_dept (deptno);

CREATE TABLE dept (
  deptno NUMBER(3) PRIMARY KEY, . . . )
  CLUSTER emp_dept (deptno);
```

Références



Oracle documentation
http://docs.oracle.com/cd/B28359_01/server.111/b28318/logical.htm

Objets de schéma

- Le terme « objet de schéma » désigne tout objet qui peut être propriété d'un utilisateur : clusters, triggers, indexes, classes JAVA, tables, types, sequences, procédures et fonctions stockées, vues,...
- Ne sont pas des objets de schéma : rôles, tablespaces, users...