

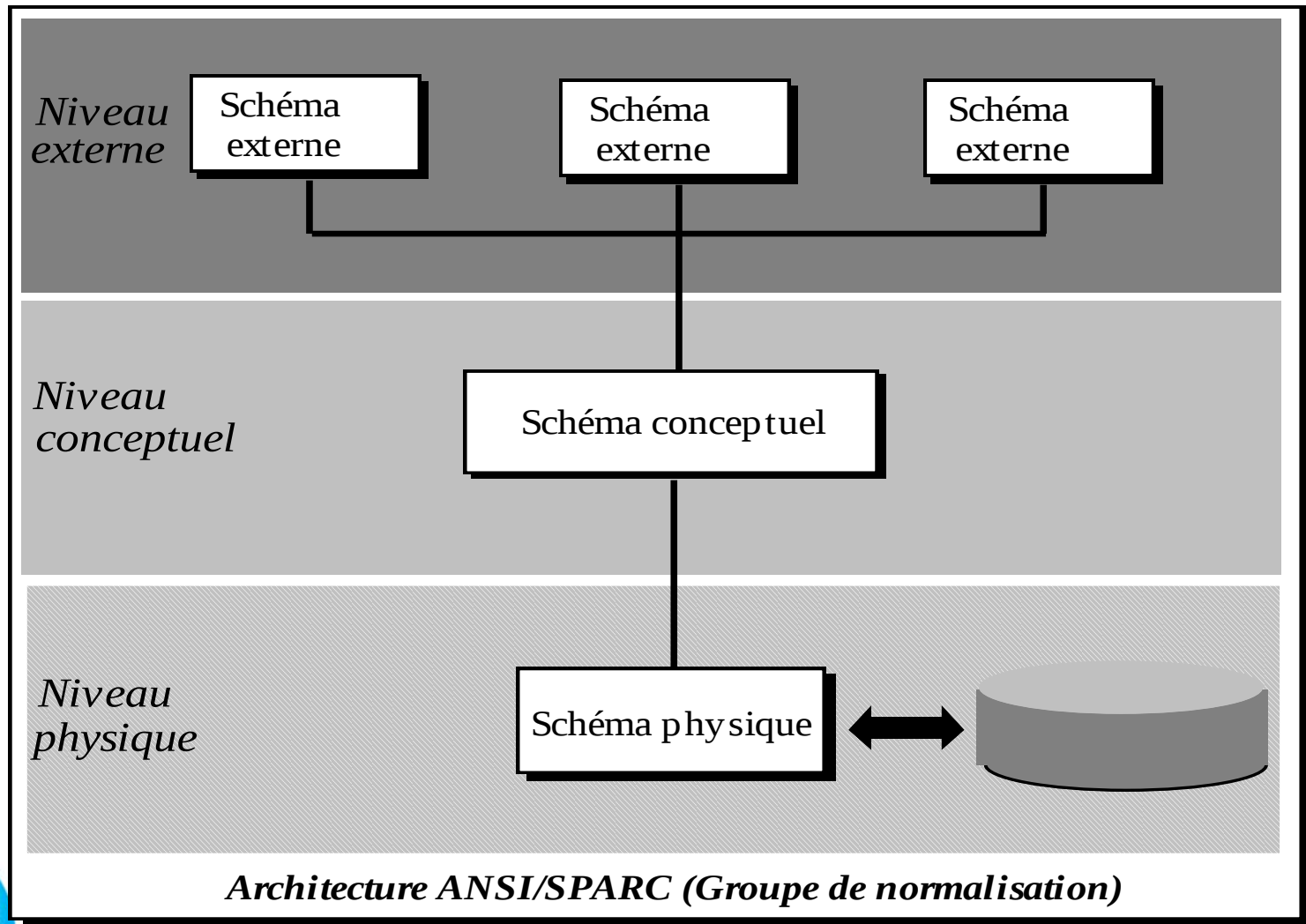
Bases de données avancées

Administration Oracle

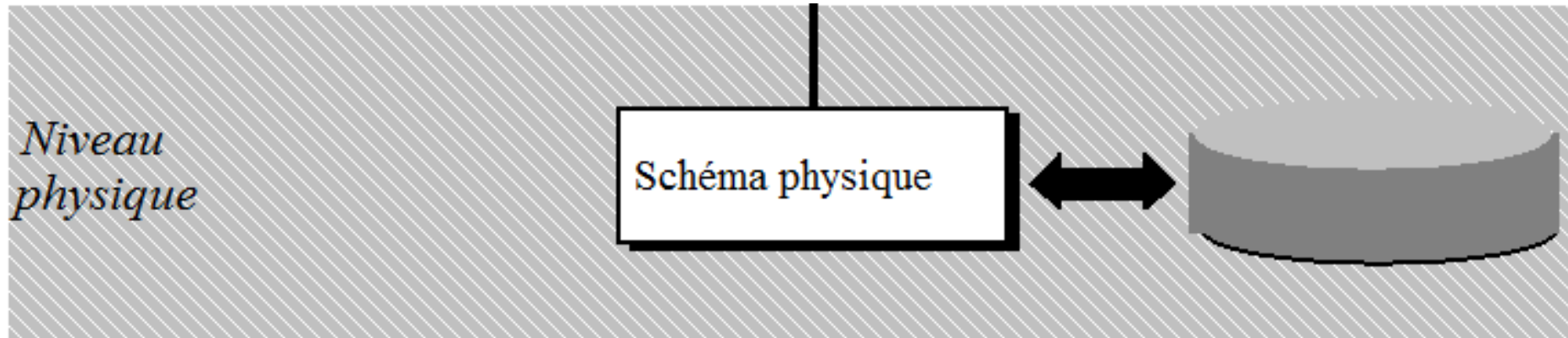
Tablespaces	
Accélérateurs	Clusters
	Indexs

2012/2013

Architecture SPARC



Architecture SPARC



- Stockage des données
- Accélérateurs
- Optimisation et plan de requêtes

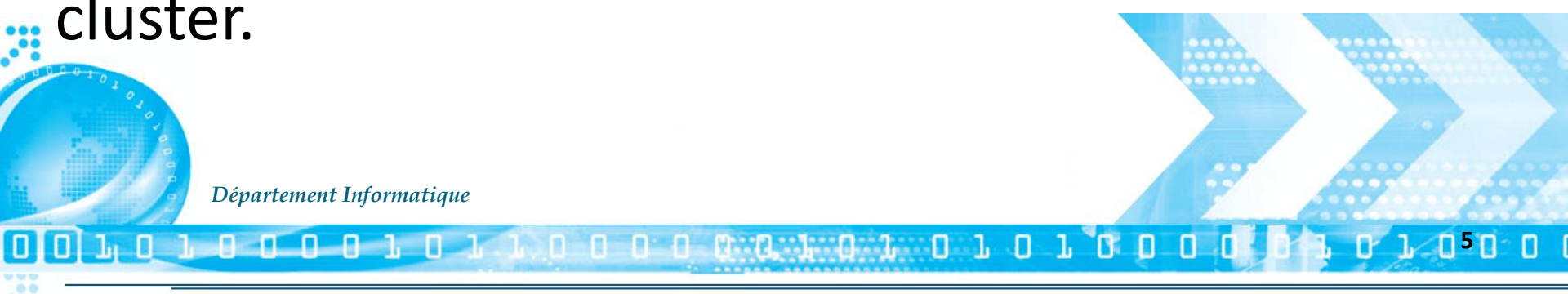
Structure logique d'une base de données

Sur chaque OS
Oracle redéfinit le système de gestion de fichiers

Base de données	Oracle											
Tablespaces	SYSTEM						USERS					
Fichiers	DBS1.ORA						USERS1.ORA				USERS2.ORA	
Segments	Données		Index	Rollback		Bootstrap	Temporaire	Données		Index		
Extensions												
Blocs logiques												
Blocs phy siques												

Segments de données

- Ils servent à stocker les données des tables utilisateurs et système et celles des tables groupées (clusters).
- Chaque table a un et un seul segment qui est créé automatiquement lors de la création de la table.
- Les données des tables groupées sont stockées dans un segment qui est créé lors de la création du cluster.



Département Informatique

00101000010110000001010101000001010000010100

5

- 

Les segments d'index

- Ces segments servent à stocker les données d'index. Ces données peuvent donc être stockées dans un tablespace distinct des données des tables.
- Un segment d'index est créé automatiquement lors de la création de l'index. On peut préciser lors de la création d'un index, le tablespace dans lequel sera créé le segment.



Segments temporaires

- Ces segments sont utilisés par Oracle pour le traitement des requêtes SQL nécessitant un espace disque temporaire.
- Les segments temporaires sont créés en cas de besoin et supprimés après l'exécution de la commande.
- Le tablespace dans lequel sont créés ces segments est défini lors de la création et modification d'un utilisateur. Si ce tablespace n'est pas défini, alors c'est le tablespace **SYSTEM** qui est utilisé par défaut.

CREATE USER ... TEMPORARY TABLESPACE ...

Segment d'amorçage

- Ce segment est créé dans le tablespace SYSTEM. Il contient les définitions du dictionnaire de données qui sont chargées lors de l'ouverture de la base
- Ce segment n'est pas géré par l'administrateur.



Segment d'annulation

- Ces segments (rollback segments) contiennent les données avant modification due à une transaction.
- Ils permettent d'annuler leur effet en cas de besoin.



Les fichiers d'Oracle

- **Les fichiers de données** (dont l'extension est .dbf). Ces fichiers contiennent l'ensemble des données de la base (les tables, les vues, les procédures stockées, ...).
- **Les fichiers Redo Log** (dont l'extension est .rdo ou .log). Ces fichiers contiennent l'historique des modifications effectuées sur la base de données



Les fichiers d'Oracle

- **Les fichiers de contrôle** (dont l'extension est .ctl voire .dbf). Ces fichiers permettent de stocker les informations sur l'état de la base de données (emplacement des fichiers, dates de création, ...)

Une base de données Oracle nécessite au minimum un fichier de données, deux fichiers redo Log et un fichier de contrôle.

- Tous ces fichiers sont référencés dans un fichier d'initialisation nommé en général *init.ora*

Le fichier d'initialisation (Exemple XE)

#####

Cursors and Library Cache

#####

open_cursors=300

#####

Database Identification

#####

db_name=XE



Le fichier d'initialisation (Exemple XE)

```
#####  
# File Configuration  
#####  
control_files=("C:\oraclexe\oradata\XE\control.dbf")  
  
#####  
# Processes and Sessions  
#####  
sessions=20
```



Le fichier d'initialisation (Exemple XE)

```
#####
```

```
# SGA Memory
```

```
sga_target=768M
```

```
#####
```

```
# Shared Server
```

```
dispatchers="(PROTOCOL=TCP) (SERVICE=XEXDB)"
```

```
shared_servers=4
```

```
#####
```

```
# Zone mémoire pour : Sort, Hash Joins, Bitmap Indexes
```

```
pga_aggregate_target=256M
```



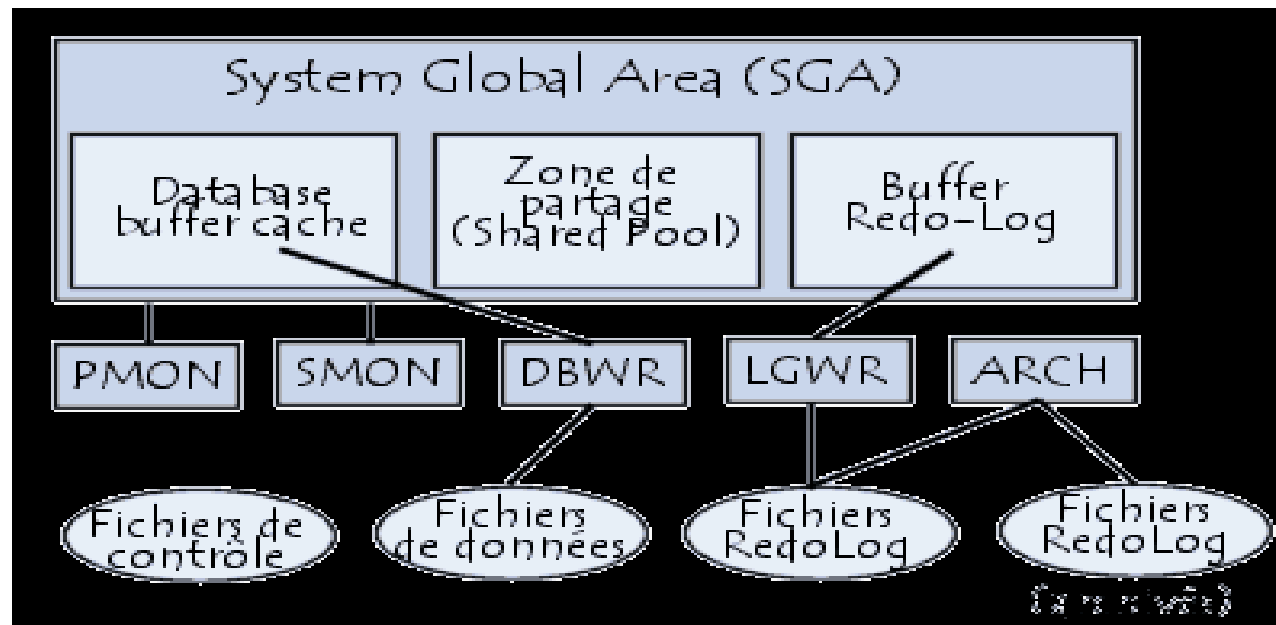
System Global Area

DBWR (*DataBase Writer* ou *Dirty Buffer Writer*), le processus chargé d'écrire le contenu des buffers dans les fichiers de données

LGWR (*Log Writer*), le processus chargé d'écrire le contenu des buffers dans les fichiers Redo Log

PMON (*Process Monitor*), le processus chargé de nettoyer les ressources, les verrous et les processus utilisateurs non utilisés

SMON (*System Monitor*), le processus chargé de vérifier la cohérence de la base de données et éventuellement sa restauration lors du démarrage si besoin



Accélérateurs: Les indexs

Une sélection sur une grande table peut être longue

On peut indexer une table sur une ou plusieurs colonnes

Table d'indexs triée (physiquement) sur les colonnes indexées. Il y aura donc deux tables : principale et indexs

Recherche : utilisation de la table d'indexs quand la Clause WHERE fait intervenir les colonnes indexées. La recherche est plus rapide.

MAJ : En plus de la table, l'index est mis à jour en maintenant l'ordre des colonnes indexées. La mise à jour est plus longue



Accélérateurs: Sélectivité

La sélectivité d'un index est égal au rapport :
Nombre de valeurs différentes de l'index) /
Nombre de lignes de la table

On parlera de sélectivité faible quand il y a peu de valeurs différentes de l'index

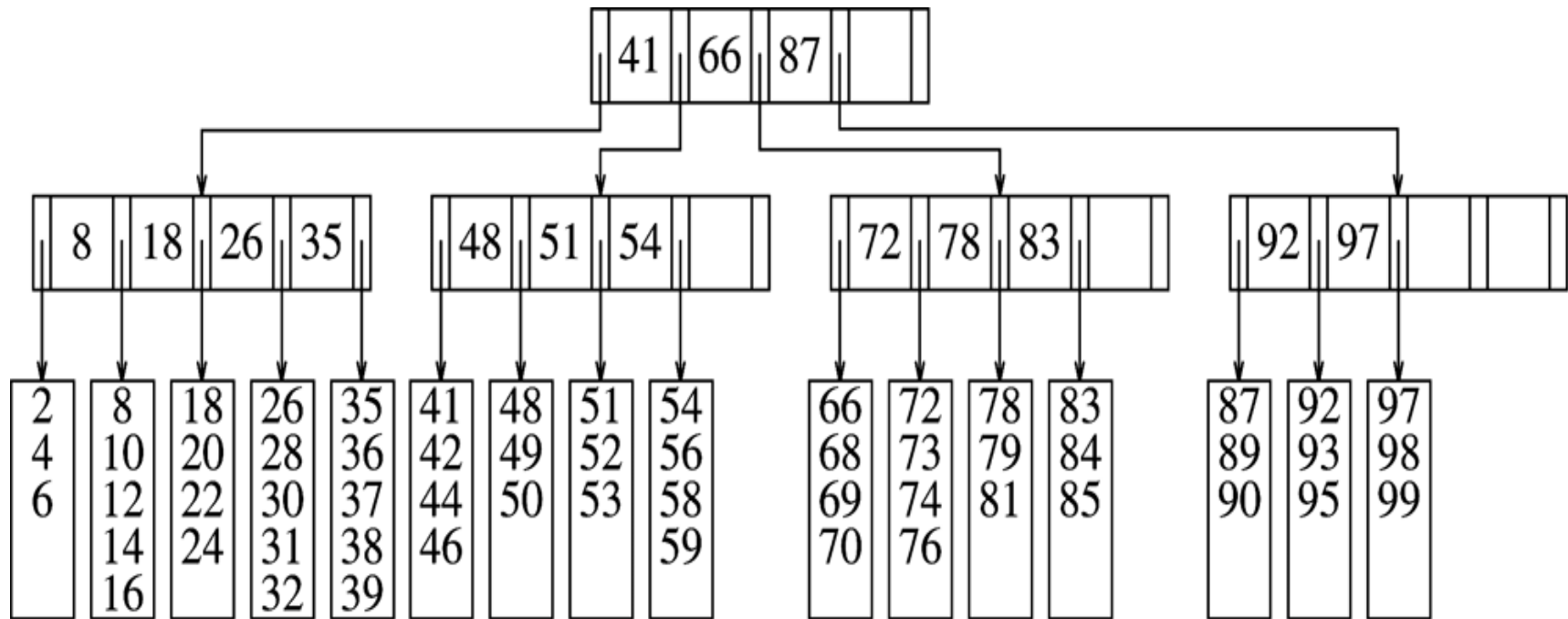
A l'inverse, on parle de sélectivité forte quand il y a beaucoup de valeurs différentes de l'index.

La valeur extrême d'une sélectivité forte vaut 1.
C'est le cas d'un index sur une clé primaire



Accélérateurs: Indexs B-TREE

B-Tree (Arbre équilibré) : Les indexs sont rangés dans un arbre trié et équilibré



Cet arbre est d'autant plus rentable que la sélectivité est élevée

Cet index ne gère pas les valeurs null. Donc, il n'est pas utilisé pour rechercher des t-uples non renseignés sur les colonnes indexées

Accélérateurs: Indexs Bitmap

Index Bitmap
Pour Classe →

N° tuple	12	11	10	9	8	7	6	5	4	3	2	1
Equipage	1	0	0	1	1	0	0	0	0	0	0	0
1ère classe	0	0	0	0	0	1	0	1	0	0	0	1
2ème classe	0	0	1	0	0	0	1	0	0	1	0	0
3ème classe	0	1	0	0	0	0	0	0	1	0	1	0

Classe	Age	Sexe	Survivant
1ère	Adulte	Femme	Oui
3ème	Adulte	Homme	Oui
2ème	Enfant	Homme	Oui
3ème	Adulte	Homme	Oui
1ère	Adulte	Femme	Oui
2ème	Adulte	Homme	Non
1ère	Adulte	Homme	Oui
Equipage	Adulte	Femme	Non
Equipage	Adulte	Femme	Oui
2ème	Adulte	Homme	Non
3ème	Adulte	Homme	Non
Equipage	Adulte	Homme	Non

Table avec des colonnes
à faible sélectivité

Tableaux extraits du mémoire de DEA ECD de Cécile FAVRE

Accélérateurs: Indexs Bitmap

Mise à jour de la table $\Rightarrow$ reconstruction totale de l'index bitmap

Null est une valeur comme une autre $\Rightarrow$ l'index gère la valeur null

Ces indexs sont donc conseillés sous les conditions suivantes :

- Faible sélectivité des colonnes indexées
- Beaucoup de lignes dans votre table
- Très peu d'activités de mise à jour



Oracle : Création d'indexs

Un index B-TREE : index par défaut

```
CREATE [UNIQUE] INDEX nomIndex  
ON Nom_de_la_table (NomChamp [ASC/DESC], ...)
```

La clause **UNIQUE** impose des valeurs différentes des colonnes indexées d'une ligne à une autre ligne

Un index B-TREE inversé

```
CREATE [UNIQUE] INDEX nomIndex  
ON Nom_de_la_table (NomChamp [ASC/DESC], ...) REVERSE
```

Les clés sont inversées : la valeur 'abc' est indexée avec 'cba'

Cet index est utile dans les filtres de la forme **col like '%bc'**

Pour annuler l'inversion :

```
ALTER INDEX nomIndex REBUILD NOREVERSE;
```



Oracle : Création d'indexs

Un index par fonction : on peut indexer sur les valeurs de retour d'une fonction

```
CREATE INDEX nomIndex
```

```
ON Nom_de_la_table (f(NomChamp) [ASC/DESC], ...)
```

Attention : la fonction doit être déterministe. Une série de valeurs en entrée ne correspond qu'à une seule valeur en sortie

Quand on écrit une fonction, on doit préciser qu'elle est déterministe : `CREATE FUNCTION ..... RETURN xxx DETERMINISTIC`

La fonction `SYSDATE` n'est pas déterministe

Un index Bitmap simple (**possible qu'en version entreprise**)

```
CREATE BITMAP INDEX nomIndex
```

```
ON Nom_de_la_table (NomChamp [ASC/DESC], ...)
```

Accélérateurs: Clusters

- Dans un MCD, Les associations (1,1) (1,n) sont très fréquentes
- En base de données, elles sont traitées par l'ajout d'une clé étrangère côté (1,1) qui est la clé primaire côté (1,n)
- Les grands classiques :
 - Une facture est associée à un client
 - Une ligne de commande est associée à une commande
- Vocabulaire : On parle de relations Maître-Détails



Accélérateurs: Clusters

- Problème : On fait souvent des requêtes de jointure naturelle entre la table **maître** et la table **détail**
- Pour optimiser ces requêtes, on peut créer des clusters
- Un cluster permet de stocker des tables qui ont des colonnes en commun. Il remplace le tablespace
- Stockage optimisé : les lignes de tables différentes ayant les mêmes valeurs sur les colonnes communes seront proches sur le disque dur
- Les colonnes communes ne seront pas physiquement dupliquées. Un index de cluster sur ces colonnes est créé.

Accélérateurs: Clusters

- `CREATE CLUSTER Nom(col1 type1 [col2 type2 ...]) SIZE tailleBloc TABLESPACE nomTablespace`
- `CREATE TABLE nomTable(...) CLUSTER nomCluster(colT1, [colT2 ...] où colTi sont des colonnes de la table.`
- Mise en cluster d'une table existante :
 1. créer une nouvelle table avec l'option cluster
 2. copier le contenu de la table initiale
 3. supprimer l'ancienne table
 4. renommer éventuellement la nouvelle table.

