

Bases de données

Souhila KACI

Partie 4

- Toujours le mot clé SELECT

```
SELECT ... FROM TABLE1, TABLE2 ...
```

- Certaines colonnes peuvent porter le même nom, d'où l'intérêt de les renommer.

SELECT * FROM ProduitCond,ProduitVrac;

ProduitCond			
codePC	Poids	Volume	codePV
'C012'	2.3	0.69	'P01'
'C253'	1	0.25	'P01'
'C258'	2	0.4	'P01'
'C693'	2	0.45	'P02'

ProduitVrac	
codePV	designation
'P01'	'sucre'
'P02'	'poivre'
'P03'	'sel'

codePC	Poids	Volume	codePV	codePV	designation
'C012'	2	0.69	'P01'	'P01'	'sucre'
'C253'	1	0.25	'P01'	'P01'	'sucre'
'C258'	2	0.4	'P01'	'P01'	'sucre'
'C693'	2	0.45	'P02'	'P01'	'sucre'
'C012'	2	0.69	'P01'	'P02'	'poivre'
'C253'	1	0.25	'P01'	'P02'	'poivre'
'C258'	2	0.4	'P01'	'P02'	'poivre'
'C693'	2	0.45	'P02'	'P02'	'poivre'
'C012'	2	0.69	'P01'	'P03'	'sel'
'C253'	1	0.25	'P01'	'P03'	'sel'
'C258'	2	0.4	'P01'	'P03'	'sel'
'C693'	2	0.45	'P02'	'P03'	'sel'

SAUF EXCEPTION : Il ne faut jamais utiliser le produit cartésien seul.

codePC	Poids	Volume	codePV	codePV	designation
'C012'	2	0.69	'P01'	'P01'	'sucre'
'C253'	1	0.25	'P01'	'P01'	'sucre'
'C258'	2	0.4	'P01'	'P01'	'sucre'
'C693'	2	0.45	'P02'	'P01'	'sucre'
'C012'	2	0.69	'P01'	'P02'	'poivre'
'C253'	1	0.25	'P01'	'P02'	'poivre'
'C258'	2	0.4	'P01'	'P02'	'poivre'
'C693'	2	0.45	'P02'	'P02'	'poivre'
'C012'	2	0.69	'P01'	'P03'	'sel'
'C253'	1	0.25	'P01'	'P03'	'sel'
'C258'	2	0.4	'P01'	'P03'	'sel'
'C693'	2	0.45	'P02'	'P03'	'sel'

Généralement, le produit cartésien doit être associé à une sélection.
On parle de **jointure**.

```
SELECT ... FROM TABLE1, TABLE2 WHERE cond;
```

- La jointure est l'opérateur de base pour gérer les clés étrangères.
- Je veux le code du produit conditionné associé à la désignation du produit vrac qu'il conditionne.

Exemple

SELECT codePC AS Code, designation AS Designation FROM
ProduitCond, ProduitVrac WHERE
ProduitCond.codePV = ProduitVrac.codePV;

ProduitCond			
codePC	Poids	Volume	codePV
'C012'	2.3	0.69	'P01'
'C253'	1	0.25	'P01'
'C258'	2	0.4	'P01'
'C693'	2	0.45	'P02'

ProduitVrac	
codePV	designation
'P01'	'sucre'
'P02'	'poivre'
'P03'	'sel'

Code	Designation
'C012'	'sucre'
'C253'	'sucre'
'C258'	'sucre'
'C693'	'poivre'

On peut remplacer ce type de jointure par le mot clé INNER JOIN

```
SELECT ... FROM t1 INNER JOIN t2 ON cond;
```

- `SELECT * FROM ProduitCond,ProduitVrac WHERE
ProduitCond.codePV=ProduitVrac.codePV;`
- `SELECT * FROM ProduitCond INNER JOIN ProduitVrac ON
ProduitCond.codePV = ProduitVrac.codePV;`

Requêtes avec la même table

- Il est possible de réaliser la jointure sur la même table.
- Il est alors nécessaire de renommer les tables.

```
SELECT ... FROM TABLE AS t1, TABLE AS t2 ... WHERE ...;
```


Exemple

- Les couples de commandes du même produit.

Commande			
numCom	codePC	quantite	DateCom
'CD01'	'C012'	25	12/05/2002
'CD02'	'C693'	14	09/11/2002
'CD03'	'C012'	4	03/07/2002
'CD04'	'C012'	11	02/06/2002
'CD05'	'C693'	71	09/01/2003

⇒

numCom	numCom
'CD03'	'CD01'
'CD04'	'CD01'
'CD04'	'CD03'
'CD05'	'CD02'

```
SELECT DISTINCT c1.numCom,c2.numCom FROM  
Commande AS c1,Commande AS c2 WHERE  
c1.codePC=c2.codePC AND c1.numCom>c2.numCom;
```

Les sous-requêtes

- Récupérer les produits conditionnés dont le volume est supérieur au volume moyen.
- Il faut utiliser la fonction d'agrégation **AVG**.

- **Une SOLUTION FAUSSE**

```
SELECT * FROM ProduitCond WHERE volume > AVG(Volume);
```

- Il est parfois nécessaire pour extraire certaines informations de devoir utiliser une requête à l'intérieur d'une autre.
- La requête interne est appelée sous-requête.
- Elle sert de critère de sélection pour la requête principale.
- Le nombre d'imbrications est théoriquement illimité.

```
SELECT ... FROM ... WHERE col ... (SELECT ... FROM ...);
```

- La sous-requête ne renvoie **qu'une seule et unique valeur**.
- Fonction d'agrégation, récupération de la clé primaire d'une table...
- Dans les autres cas, une erreur est générée.

- Afficher les produits conditionnés à base de sucre.

ProduitCond			
codePC	Poids	Volume	codePV
'C012'	2.3	0.69	'P02'
'C253'	1	0.25	'P01'
'C258'	2	0.4	'P01'
'C693'	2	0.45	'P01'

ProduitVrac	
codePV	designation
'P01'	'sucre'
'P02'	'poivre'
'P03'	'sel'

- Classiquement, on peut effectuer une jointure

```
SELECT codePC FROM ProduitVrac, ProduitCond WHERE  
ProduitVrac.codePV=ProduitCond.codePV AND  
designation='sucre';
```

- On peut procéder d'une autre façon
 - ① Chercher le code du produit dont la désignation est sucre.
 - ② Afficher ensuite les produits conditionnés correspondants.

```
SELECT codePV FROM ProduitVrac WHERE designation  
= 'sucre';
```

- On obtient une table à une seule ligne et une seule colonne :

codePV
'P01'

- On utilise cette requête pour trouver les produits conditionnés à base de sucre.

```
SELECT codePC FROM ProduitCond WHERE codePV =  
(SELECT codePV FROM ProduitVrac WHERE designation  
= 'sucre');
```

code
'C253'
'C258'
'C693'

- Sous-requête : codePV de sucre

```
SELECT codePV FROM ProduitVrac WHERE designation  
= 'sucre'
```

codePV
'P01'

- Produit conditionné dont le codePV est celui calculé par la sous-requête :

```
SELECT codePC FROM ProduitCond WHERE  
codePV='P01';
```

codePC
'C253'
'C258'
'C693'

Sous-requête renvoyant plusieurs valeurs

- Le résultat de la sous-requête ne peut pas être utilisé directement avec l'opérateur $=$.
- Par contre, on peut utiliser cette requête dans le prédicat IN.

Rappel

- **IN** : Vérifie si une valeur appartient à une liste de valeurs comparables.
- a IN (3,4.5)

Recherchons les codes de produits conditionnés dont le nom commence par un "s".

- Définition de la sous-requête : les codes des produits en vrac dont le nom commence par un "s". Cette sous-requête peut renvoyer plusieurs valeurs.
- Définition de la requête principale : les codes des produits conditionnés dont le codePV appartient à la liste des codes retournés par la sous-requête.

Exemple... La sous-requête

- les codes des produits en vrac dont le nom commence par un "s".
- Cette sous-requête peut renvoyer plusieurs valeurs.

ProduitVrac	
codePV	designation
'P01'	'sucre'
'P02'	'poivre'
'P03'	'sel'

⇒

codePV
'P01'
'P03'

```
SELECT codePV FROM ProduitVrac WHERE designation  
LIKE 's%';
```

Exemple... La requête principale

- Les codes des produits conditionnés dont le codePV appartient à la liste des codes retournés par la sous-requête.

ProduitCond			
codePC	Poids	Volume	codePV
'C012'	2.3	0.69	'P02'
'C253'	1	0.25	'P01'
'C258'	2	0.4	'P01'
'C693'	2	0.45	'P01'

⇒

codePC
'C253'
'C258'
'C693'

```
SELECT codePC from ProduitCond WHERE codePV IN  
(P01,P03);
```

Exemple ... Fin

ProduitCond			
codePC	Poids	Volume	codePV
'C012'	2.3	0.69	'P02'
'C253'	1	0.25	'P01'
'C258'	2	0.4	'P01'
'C693'	2	0.45	'P01'

ProduitVrac	
codePV	designation
'P01'	'sucre'
'P02'	'poivre'
'P03'	'sel'

SELECT codePC from ProduitCond WHERE codePV IN
(**SELECT** codePV FROM ProduitVrac WHERE designation
LIKE 's%');

codePC
'C253'
'C258'
'C693'

- Lister les commandes de 2003 dont la quantité est supérieure à toutes celles de 2002.
- Comment faire ?
- Il nous manque des opérateurs.

- On peut étendre les opérateurs de comparaison $=$, $<$, $>$... à une liste de valeurs retournée par une sous-requête.
- **ALL** : La condition sera vraie si elle est vraie pour chaque valeur renvoyée par la sous-requête.
- **ANY** : La condition sera vraie si elle est vraie pour au moins une des valeurs renvoyées par la sous-requête.

Exemple

- $v1 > \text{ALL (SELECT ... FROM ...)}$
- $v1 > \text{ANY (SELECT ... FROM ...)}$

Lister les commandes de 2003 dont la quantité est supérieure à **toutes** celles de 2002.

Procédé

- Récupérer la liste des commandes de 2002.
- Comparer avec celles de 2003.

Exemple ...2

- Récupérer les quantités des commandes de 2002 :
SELECT quantite **FROM** Commande **WHERE**
year(DateCom)=2002;
- Comparer avec celles de 2003.

Commande			
numCom	codePC	quantite	DateCom
'CD01'	'C012'	25	12/05/2002
'CD02'	'C693'	14	09/11/2002
'CD03'	'C012'	4	03/07/2002
'CD04'	'C012'	11	02/06/2002
'CD05'	'C693'	71	09/01/2003

⇒

quantite
71

SELECT numCom,quantite **FROM** Commande **WHERE**
year(DateCom)=2003 **AND** quantite>=ALL(**SELECT**
quantite **FROM** Commande **WHERE** year(DateCom)=2002);

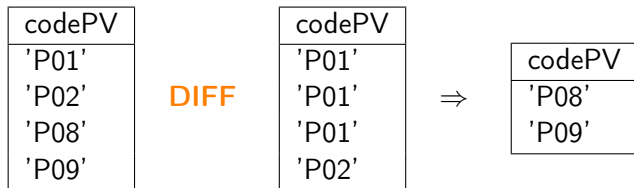
Le code du produit conditionné le plus lourd.

- Deux solutions
 - On utilise la fonction d'agrégation **MAX**
`SELECT code FROM produitCond WHERE
poids = SELECT (MAX(Poids) FROM ProduitCond);`
 - On utilise l'opérateur **ALL**
`SELECT code FROM produitCond WHERE
poids >= ALL (SELECT POIDS FROM produitCond);`

- L'opérateur MINUS est l'opérateur de différence de l'algèbre relationnelle.
- $A \text{ MINUS } B$: tous ceux qui sont dans A mais qui ne sont pas dans B.
- Certains SGBDR ne possèdent pas l'opérateur MINUS.
- On peut le simuler avec des sous-requêtes et les prédicats NOT et IN.

- Les codes de produits en vrac qui ne sont jamais utilisés.

```
SELECT codePV FROM ProduitVrac  
MINUS  
SELECT codePV FROM ProduitCond;
```



- Utilisation des sous-requêtes si l'opérateur MINUS n'est pas implanté.
- Procédé
 - Récupérer les codesPV utilisés :
SELECT codePV FROM ProduitCond;
 - Récupérer dans la table ProduitVrac, les produits non utilisés :

```
SELECT codePV FROM ProduitVrac WHERE  
codePV NOT IN (SELECT codePV FROM ProduitCond);
```

- Bien entendu, on peut utiliser des sous-requêtes à l'intérieur d'une condition HAVING.
- Code du produit en vrac le plus souvent utilisé.

```
SELECT codePV FROM ProduitCond GROUP BY codePV  
HAVING COUNT(*) >= ALL (SELECT COUNT(*) FROM  
ProduitCond GROUP BY codePV);
```