Clés des tables

- La clé primaire Exer'une table correspond à la notion d'identifiant dans le formalisme conceptuel
- Un SBGDR vérifie l'unicité de la clé, chaque fois que l'on insère un uplet, au sein de la table
- Une contrainte réferentielle est un lien sémantique entre deux tables
- Ce lien est réalisé par la duplication de la clé primaire d'une table sur l'autre table
- Cette clé dupliqué s'appelle clé étrangère

Type des données

- Le type `CHAR` : chaîne de caractères de longueur fixe.
  de 1 (défaut) à 2000 carcatères

- Le type `VARCHAR` : chaîne de caractères de longueur variable. de 1 à 4000 caractères

- le Type `NUMBER` : données numériques. Comprend les nombres entiers et les nombres à virgule flottante. Il est possible de donner une précision et une échelle à un type `NUMBER`

- Le type `DATE` : données de type date ou de type heure

Requêtes Simples

Introduction

- Le mot clé `SELECT` est essentiel

- Utilisé pour

  ✓ La projection
  ✓ La selection
  ✓ Le produit cartésien
  ✓ La jointure

La projection

- Afficher certaines colonnes d'une table
- Les colonnes selectionnées doivent obligatoirement appartenir à la table

```
SELECT att1, att2 ...  FROM TABLE;
```

Vins1	Cru	Mill	Région	Type
	Chenas	1983	Beaujolais	Rouge
	Tokay	1980	Alsace	Blanc
	Aix	1995	Provence	Rouge
	Aix	2000	Provence	Rouge
	Cassis	2000	Provence	Blanc

$\Rightarrow$

Cru	Région
Chenas	Beaujolais
Tokay	Alsace
Aix	Provence
Aix	Provence
Cassis	Provence

SELECT Cru,Région FROM Vins1;

Suppression des doublons

- Par défaut les doublons ne sont pas supprimés
- Pour le faire : Utiliser le mot clé `DISTINCT`

Vins1	Cru	Mill	Région	Type
	Chenas	1983	Beaujolais	Rouge
	Tokay	1980	Alsace	Blanc
	Aix	1995	Provence	Rouge
	Aix	2000	Provence	Rouge
	Cassis	2000	Provence	Blanc

$\Rightarrow$

Cru	Région
Chenas	Beaujolais
Tokay	Alsace
Aix	Provence
Cassis	Provence

SELECT DISTINCT Cru,Région FROM Vins1;

Renommage

- Pour des raisons de commodités ou de clarté, il est possible de renommer les colonnes de la table résultat

- Le mot clé `AS`

Vins	Cru	Mill	Région	Type
	Chenas	1983	Beaujolais	Rouge
	Tokay	1980	Alsace	Blanc
	Aix	1995	Provence	Rouge
	Aix	2000	Provence	Rouge
	Cassis	2000	Provence	Blanc

$\Rightarrow$

AOC	Région
Chenas	Beaujolais
Tokay	Alsace
Aix	Provence
Cassis	Provence

SELECT DISTINCT Cru AS AOC,Région FROM Vins;

Cas particulier

- Le caractère générique $*$
- Il sert à selectionner toutes les colonnes pour la projection

Vins	Cru	Mill	Région	Type
	Chenas	1983	Beaujolais	Rouge
	Tokay	1980	Alsace	Blanc
	Aix	1995	Provence	Rouge
	Aix	2000	Provence	Rouge
	Cassis	2000	Provence	Blanc

$\Rightarrow$

Cru	Mill	Région	Type
Chenas	1983	Beaujolais	Rouge
Tokay	1980	Alsace	Blanc
Aix	1995	Provence	Rouge
Aix	2000	Provence	Rouge
Cassis	2000	Provence	Blanc

SELECT * FROM Vins;

Un peu d'ordre

- Par défaut, le résultat des requêtes n'est pas ordonnée
- Heureusement, il est possible de faire cet ordonnancement
- On peut ordonner suivant plusieurs colonnes
- Le mot clé : `ORDER BY`
- Pour chaque critère, le tri peut être ascendant ou descendant `DESC`
- Par défaut il est ascendant
- Le tri peut également être fait à partir d'une expression construite à partir d'attributs

Un peu d'ordre ... 2

ORDER BY nom_col1 || num_col1 [DESC] ...

Vins	Cru	Mill	Région	Type
	Chenas	1983	Beaujolais	Rouge
	Tokay	1980	Alsace	Blanc
	Aix	1995	Provence	Rouge
	Aix	2000	Provence	Rouge
	Cassis	2000	Provence	Blanc

$\Rightarrow$

Cru	Mill	Région	Type
Tokay	1980	Alsace	Blanc
Chenas	1983	Beaujolais	Rouge
Aix	1995	Provence	Rouge
Aix	2000	Provence	Rouge
Cassis	2000	Provence	Blanc

SELECT * FROM Vins1 ORDER BY 2;
SELECT * FROM Vins1 ORDER BY Mill;

Exemple

Vins	Cru	Mill	Région	Type
	Chenas	1983	Beaujolais	Rouge
	Tokay	1980	Alsace	Blanc
	Aix	1995	Provence	Rouge
	Aix	2000	Provence	Rouge
	Cassis	2000	Provence	Blanc

$\Rightarrow$

Cru	Mill	Région	Type
Tokay	1980	Alsace	Blanc
Chenas	1983	Beaujolais	Rouge
Aix	1995	Provence	Rouge
Cassis	2000	Provence	Blanc
Aix	2000	Provence	Rouge

SELECT * FROM Vins1 ORDER BY Mill,Cru DESC;

La restriction - Selection

- Permet l'affichage de certaines lignes, qui vérfie un critère donné
- Le critère est une expression booléenne plus ou moins compliquée
- Le mot clé `WHERE`

SELECT att1,att2 ... FROM table WHERE condition

Vins	Cru	Mill	Région	Type
	Chenas	1983	Beaujolais	Rouge
	Tokay	1980	Alsace	Blanc
	Aix	1995	Provence	Rouge
	Aix	2000	Provence	Rouge
	Cassis	2000	Provence	Blanc

⇒

Cru	Mill	Région	Type
Aix	1995	Provence	Rouge
Aix	2000	Provence	Rouge

SELECT * FROM Vins1 WHERE (Cru="Aix");

Opérateurs de comparaison

Opérateur	Numérique	Chaîne de caractères	Date/Heure
$=$	égal	identique	en même temps
$<>$	différent	different	pas en même temps
$>$	supérieur	supérieure	après
$<$	inférieur	inférieure	avant $\forall x$
		...	

- Et les connecteurs logiques habituels : AND, OR, NOT
- Attention aux priorités
- Il en existe d'autres, nous les verrrons plus tard

Un petit exercice

Voiture	Immatriculation	Marque	Année	Prix	IdProprio
	1111 AA 01	Toyota	1997	16 000	Id01
	2222 BB 02	Peugeot	2000	31 200	Id01
	3333 CC 03	Fiat	1997	2 000	Id03
	4444 DD 13	Fiat	1995	30 300	Id02
	5555 EE 62	Renault	1997	21 000	Id02
	6666 FF 59	Opel	1999	2 900	Id01
	7777 ZZ 75	Ford	1998	22 222	Id03

Personnes	IdProprio	Nom	Prénom	Naissance
	Id01	Martin	Paul	01/02/1967
	Id02	Duval	Jean	03/09/1980
	Id03	Dupond	Laurence	01/01/1945
	Id04	Durand	Julie	03/03/1985

la suite ...

- Liste des immatriculations
- Liste des voitures de 1996
- Voitures qui appartiennent au proprio Id01
- Liste des voitures entre 10000 et 20000 euros
- Différentes marques de voitures
- Id des propriétaires ayant des voitures de plus de 20000 euros

Le produit cartésien

- Toujours le mot clé `SELECT`

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

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

Exemple

Vins	Cru	Région	Type
	Chenas	Beaujolais	Rouge
	Tokay	Alsace	Blanc
	Aix	Provence	Rouge

Années	Mill
	1988
	2000

Cru	Région	Type	Mill
Chenas	Beaujolais	Rouge	1988
Tokay	Alsace	Blanc	1988
Aix	Provence	Rouge	1988
Chenas	Beaujolais	Rouge	2000
Tokay	Alsace	Blanc	2000
Aix	Provence	Rouge	2000

SELECT * FROM Vins ,Années;

La jointure

- Produit cartésien
  - ✓ Explosion du nombres de lignes
  - ✓ SAUF EXCEPTION : Il ne faut jamais utiliser le produit cartésien seul

- Le produit cartésien associé à une selection

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

Exemple

Vins1	Cru	Mill	Région
	Chenas	1983	Beaujolais
	Tokay	1980	Alsace
	Aix	1995	Provence

Vins2	Cru	Type
	Bandol	Rosé
	Tokay	Blanc
	Cassis	Blanc
	Aix	Rouge

Cru	Mill	Région	Type
Tokay	1980	Alsace	Blanc
Aix	1995	Provence	Rouge

SELECT * FROM Vins1,Vins2 WHERE Vins1.cru = Vins2.cru;

Autre syntaxe

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

SELECT ... FROM t1 INNER JOIN t2 ON cond

- **SELECT * FROM Vins1,Vins2 WHERE Vins1.cru = Vins2.cru;**
- **SELECT * FROM Vins1 INNER JOIN Vins2 ON Vins1.cru = Vins2.cru;**

Requetes 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 t1,TABLE t2 ... WHERE ...;
```

Exemple

- Les mêmes crus avec leurs années de millésimes

Vins	Cru	Mill	Région
	Chenas	1983	Beaujolais
	Tokay	1980	Alsace
	Aix	1995	Provence
	Aix	1996	Provence
	Tokay	1995	Provence
	Aix	2000	Provence

Cru	Mill	Mill	Région
Tokay	1980	1995	Alsace
Aix	1995	1996	Provence
Aix	1995	2000	Provence
Aix	1996	2000	Provence

SELECT v1.Cru AS Cru ,v1.Mill as Mill1,v2.Mill AS Mill2 ,v1.Region AS Région

FROM Vins v1, Vins v2 WHERE v1.Cru=v2.Cru AND v1.Mill $<$ v2.Mill;

Plan

- Attributs calculés
- Regroupements de lignes

Attributs Calculés

Création de nouvelles colonnes

- On peut créer des nouvelles colonnes qui sont “construites” à partir de colonne existantes

ProduitsCond		
code	Poids	Volume
C012	2,3	0,69
C253	1	0,25
C258	2	0,4
C693	2	0,45

$\Rightarrow$

PoidsNet
2,07
0,9
1,8
1,8

SELECT poids*0.9 AS PoidsNet FROM ProduitCond;

Exemple

ProduitsCond		
code	Poids	Volume
C012	2,3	0,69
C253	1	0,25
C258	2	0,4
C693	2	0,45

$\Rightarrow$

Code	Poids	Volume	Densit
C012	2,3	0,69	0,3
C253	1	0,25	0,25
C258	2	0,4	0,2
C693	2	0,45	0,22

SELECT code,poids, volume, volume/poids AS densité FROM ProduitCond;

Fonctions d'agrégation

- Obtention de valeurs agrégées sur une colonne
  - ✓ Somme `SUM`, moyenne `AVG`
  - ✓ Minimum `MIN`, Maximum `MAX`
  - ✓ Nombre de lignes `COUNT`

Exemple

ProduitsCond		
code	Poids	Volume
C012	2,3	0,69
C253	1	0,25
C258	2	0,4
C693	2	0,45

- MIN(code) = 'C012'
- SUM(Poids) = 7.3
- Max(Volume) = 0,69
- Count(code) = 4

Syntaxe

- **SELECT SUM(c1),AVG(c2) ... FROM ... WHERE ...;**
- Ceci nous donne, pour un nombre quelconque de lignes, une seule et unique valeur qu'il n'est pas possible de rajouter à chacune des lignes
- Nous obtenons un résultat agrégé à partir de l'ensemble des valeurs d'une colonne

Exemple

ProduitsCond		
code	Poids	Volume
C012	2,3	0,69
C253	1	0,25
C258	2	0,4
C693	2	0,45

$\Rightarrow$

Sum(Poids)
7,3

SELECT SUM(Poids) FROM ProduitCond;

Exemple

ProduitsCond		
code	Poids	Volume
C012	2,3	0,69
C253	1	0,25
C258	2	0,4
C693	2	0,45

$\Rightarrow$

nbP	VolMoyen	VolMin
4	0,45	0,25

SELECT COUNT(*) AS nbP, AVG(volume) AS volMoyen, MIN(volume) AS volMin FROM PorduitCond;

Attention

- Le résultat d'un opérateur d'agrégation fournit une unique valeur
- Une table à une seule ligne

ProduitsCond		
code	Poids	Volume
C012	2,3	0,69
C253	1	0,25
C258	2	0,4
C693	2	0,45

$\Rightarrow$ IMPOSSIBLE

SELECT Code, SUM(Poids) FROM ProduitCond;

Attributs calculés et Selection

- Lorsqu'une requête combine des opérations de selection et de calcul, la selection est toujours faite AVANT le calcul

ProduitsCond		
code	Poids	Volume
C012	2,3	0,69
C253	1	0,25 ✓
C258	2	0,4 ✓
C693	2	0,45 ✓

$\Rightarrow$

Sum(Poids)
5

SELECT SUM(Poids) FROM ProduitCond WHERE Volume < 0.5;

Exemple

ProduitsCond		
code	Poids	Volume
C012	2,3	0,69
C253	1	0,25
C258	2	0,4
C693	2	0,45

✓ ✓ ✓

$\Rightarrow$

pBrut	pNet
5	4,5

SELECT SUM(Poids) AS pBrut, SUM(Poids)*0.9 AS pNet
FROM ProduitCond WHERE Volume*2 $<$ 1;

Opérateurs d'agrégation

Opérateur	Numérique	Chaîne	Date/Heure
Count	Nombre de valeurs connues		
MAX	Le plus grand	Le plus grand	Le plus tard
MIN	Le plus petit	Le plus petit	Le plus tôt
SUM	La somme	#####	#####
AVG	La moyenne	#####	#####

Remarque : COUNT

ProduitsCond		
code	Poids	Volume
C012	2,3	0,69
C253	1	0,25
C258	2	0,4
C693		0,45

$\Rightarrow$

Nb
3

SELECT COUNT(Poids) as Nb FROM ProduitsCond

Remarque : COUNT

ProduitsCond		
code	Poids	Volume
C012	2,3	0,69
C253	1	0,25
C258	2	0,4
C693		0,45

$\Rightarrow$

Nb
4

SELECT COUNT(*) as Nb FROM ProduitsCond

Exercice

- Afficher le nombre total de voitures
- Afficher le nombre total de FIAT
- Afficher le prix moyen et le prix total de l'ensemble des voitures en francs et en euros
- Idem, mais uniquement pour les voitures de moins de 1996
- Afficher le nombre de voitures qui coutent moins de 20 000 euros TTC (TVA à 18.6%)
- Afficher le nombre de voitures que possède Laurence Dupond

Regroupements de Lignes

Introduction

- SQL permet de regrouper en une seule ligne les informations relatives à un ensemble de données ayant un caractère commun

ProduitsCond			
code	Poids	Volume	codePV
C012	2,3	0,69	P01
C253	1	0,25	P01
C258	2	0,4	P01
C693	2	0,45	PO2

$\Rightarrow$

codePV	NbP
P01	3
P02	1

- Afficher pour chaque codePV, le nombre de références de produits conditionnés qui lui correspondent

- Il faut utiliser l'opérateur d'agrégat COUNT

NON : **SELECT DISTINCT CodePV,COUNT(*) FROM ProduitCount;**

Le regroupement

- Regrouper ensemble les lignes qui ont le même codePV
- Créer des sous tables
- Utiliser les fonctions d'agrégations sur les sous tables et non sur les talbes entières
- Générer une ligne unique par sous table
- Le mot clé pour regrouper des lignes GROUP BY

SELECT CodePV, count(*) AS nbProduits FROM ProduitCond GROUP BY CodePV

Fonctionnement : 1ere étape

- Les lignes sont regroupés par codePV identiques

ProduitsCond			
code	Poids	Volume	codePV
C012	2,3	0,69	P01
C253	1	0,25	P01
C258	2	0,4	P01
C693	2	0,45	PO2

Foncionnement : 2eme étape

- Chaque ensemble de lignes est regroupé en une seule
- Bien sur, les attributs code, Poids, Volume ne peuvent pas être conservés

ProduitsCond			
code	Poids	Volume	codePV
C012	2,3	0,69	P01
C253	1	0,25	P01
C258	2	0,4	P01
C693	2	0,45	PO2

$\Rightarrow$

codePV
P01
P02

Fonctionnement :3eme étape

- On ajoute la colonne COUNT(*) qui nous indique, pour chaque codePV, le nombre de lignes qui ont été regroupés

ProduitsCond			
code	Poids	Volume	codePV
C012	2,3	0,69	P01
C253	1	0,25	P01
C258	2	0,4	P01
C693	2	0,45	PO2

$\Rightarrow$

codePV	NbP
P01	3
P02	1

SELECT CodePV, count(*) AS nbProduits FROM ProduitCond GROUP BY CodePV

Remarque

- Le résultat comporte une valeur apparaissant dans la colonne regroupement
- Les valeurs des colonnes sur laquelle on effectue le regroupement peuvent être affichées
- Les autres colonnes NE peuvent PAS l'être
- Les fonctions d'agrégation agissent sur les sous tables générées par le regroupement

Exemple

Commande		
numCom	code	quantité
CD01	C012	25
CD02	C693	14
CD03	C012	4
CD04	C012	11
CD05	C693	71

$\Rightarrow$

code	NbC	Qte
C012	3	40
C693	2	85

- Afficher pour chaque produit conditionné le nombre de commandes passées et la quantité totale commandée

SELECT code, COUNT(*) as NbC, SUM(Quantité) as Qte
FROM Commande GROUP BY code;

Groupement et Selection

- La clause de selection WHERE permet de ne conserver que les lignes qui correspondent au critère enoncé. Cette selection agit AVANT le regroupement.

- Les lignes sont sélectionnées puis regroupées

Exemple

Commande		
numCom	code	quantité
CD01	C012	25
CD02	C693	14
CD03	C012	4
CD04	C012	11
CD05	C693	71

$\Rightarrow$

code	NbC	Qte
C012	1	25
C693	2	85

- Afficher pour chaque produit conditionné le nombre de commandes passées dont la quantité dépasse 12 et la quantité totale commandée

SELECT code, COUNT(*) as NbC, SUM(Quantité) as Qte FROM Commande WHERE Quantité¿12 GROUP BY code;

Exercice

- Afficher le nombre de voitures par identifiant de propriétaire
- Afficher le capital de chaque propriétaire
- Afficher le prix moyen par marque
- Afficher le nombre de voitures de plus de 20000 euros pour chaque propriétaire
- Afficher le nombre de voitures mises en circulation après 1996 pour chaque personne

Regroupement et Sélection

- Avec la clause WHERE on sélectionne d'abord et on regroupe après
- Comment faire si on veut regrouper après la sélection
- Exemple : Liste des produits conditionnés qui apparaissent dans au moins 3 commandes ?
- Dans ce cas il faut utiliser le mot clé **HAVING**

Comment ça marche

1. On regroupe les lignes qui possèdent le même code
2. On compte le nombre de lignes regroupées
3. On supprime celles pour lesquelles le compte est inférieur à 2

Exemple

Commande		
numCom	code	quantité
CD01	C012	25
CD02	C693	14
CD03	C012	4
CD04	C012	11
CD05	C693	71

$\Rightarrow$

code	NbC	Qte
C012	3	40

SELECT code, COUNT(*) as NbC FROM Commande GROUP BY code HAVING COUNT(*) $>$ 2;

Regrouper plusieurs colonnes

- Les regroupements peuvent concerner plusieurs colonnes en même temps

Commande			
numCom	code	quantité	CodeClient
CD01	C012	25	CL1
CD02	C693	14	CL2
CD03	C012	4	CL2
CD04	C012	11	CL2
CD05	C693	71	CL1

$\Rightarrow$

code	CodeClient	Qte
C012	CL1	25
C012	CL2	15
C693	CL1	71
C693	CL2	11

- Nombre de produits par clients et par référence

SELECT code,CodeClient,SUM(Quantité) as Qte FROM Commande GROUP BY code,CodeClient;

Exercice

- Afficher le capital et l'identifiant des propriétaires dont le capital est supérieur à 50 000 euros
- Capital de chaque propriétaire (par identifiant) par année
- Idem mais avec le nom et le prénom de chaque propriétaire
- Valeur moyenne des voitures par année et par marque
- Identifiant des propriétaires de plus de 2 voitures qui coutent plus de 20 000 euros

Un petit exercice

Voiture	Immatriculation	Marque	Année	Prix	IdProprio
	1111 AA 01	Toyota	1997	16 000	Id01
	2222 BB 02	Peugeot	2000	31 200	Id01
	3333 CC 03	Fiat	1997	2 000	Id03
	4444 DD 13	Fiat	1995	30 300	Id02
	5555 EE 62	Renault	1997	21 000	Id02
	6666 FF 59	Opel	1999	2 900	Id01
	7777 ZZ 75	Ford	1998	22 222	Id03

Personnes	IdProprio	Nom	Prénom	Naissance
	Id01	Martin	Paul	01/02/1967
	Id02	Duval	Jean	03/09/1980
	Id03	Dupond	Laurence	01/01/1945
	Id04	Durand	Julie	03/03/1985

Quelques Compléments

Opérations sur les dates

- SQL permet de gérer efficacement les dates :
  - ✓ Différence entre deux dates : écart en nombre de jours
  - ✓ Date + une constante entière k : augmentation de k jours
  - ✓ year(date) : Année
  - ✓ month(date) : Le numéro du mois
  - ✓ day(date) : Le numéro du jour
  - ✓ weekday(date) : Le numéro du jour de la semaine
  - ✓ now() : Date du jour
  - ✓ ...

Exemple

Commande			
numCom	code	quantité	Date
CD01	C012	25	17-jan-2003
CD02	C693	14	30-dec-2002
CD03	C012	4	12-may-2001
CD04	C012	11	03-jun-2000
CD05	C693	71	14-03-2001

date	jour	mois	année	di
17-jan-2003	17	01	2003	37
30-dec-2002	30	12	2002	28
12-may-2001	12	05	2001	60
03-jun-2000	03	06	2000	11
14-03-2001	14	03	2001	54

SELECT date,day(date) as jour, month(date) as mois, year(date) as année, now() - date as diff FROM Commande

Prédicat IsNull

- Le prédicat **IS NULL** teste la valeur NULL d'une colonne
- La valeur NULL correspond à l'absence de valeur
- On peut bien sur prendre la négation de cette condition

Commande		
numCom	code	quantité
CD01	C012	25
CD02	C693	
CD03	C012	4
CD04	C012	
CD05	C693	71

$\Rightarrow$

numCom
CD02
CD04

SELECT numCom FROM Commande WHERE quantité IS NULL

Prédicat Between

- Crée un raccourci qui permet d'exprimer les conditions $\leq$ et $\geq$
- a BEETWEEN 0 AND 3 $\Leftrightarrow$ a$\geq$ 0 AND a$\leq$ 3
- On peut bien sur prendre la négation de cette condition
  - ✓ NOT (a BEETWEEN 0 AND 3)
- On peut l'utiliser sur des chaînes de caractères
- Utilité !!!

Prédicat Like

- Opérateur de comparaison de chaînes de caractères
- c1 **LIKE** 'forme'
- On peut prendre la négation de ce prédicat
- On peut introduire des expressions régulières
  - ✓ _ représente un caractère quelconque
  - ✓ % représente une chaîne de caractère quelconque
- Exemple
  - ✓ **LIKE** 'R__' : 3 caractères dont le premier est un R
  - ✓ **LIKE** '%phone' : chaîne que se termine par phone

Prédicat IN

- Vérifie si une valeur appartient à une liste de valeurs comparables
- a **IN**(3,5,67)
- On peut prendre la négation de ce prédicat
  - ✓ NOT (a IN (3,5,7)

Opérateurs ensemblistes

- Les opérateurs classique de l'algèbre relationnelle ont leur équivalent en langage SQL
  - ✓ UNION un doublon n'apparaît qu'une seule fois
  - ✓ INTERSECT
  - ✓ DIFF
- sur certains SGBDR, ces opérateurs ne sont pas implantés
- Attention, les attributs des deux tables doivent être identiques

SELECT .. FROM ... WHERE ...

UNION

SELECT .. FROM ... WHERE ...

Exemple

- Les produits conditionnés dont le poids est inférieur à 2 ou le volume inférieur à 0,4

ProduitsCond		
code	Poids	Volume
C012	2,3	0,69
C253	1	0,25
C258	2	0,4
C693	2	0,45

$\rightarrow$

ProduitsCond		
code	Poids	Volume
C253	1	0,25
C258	2	0,4
C693	2	0,45

SELECT code FROM ProduitsCond WHERE poids <= 2

UNION

SELECT code FROM ProduitsCond WHERE volume <= 0,4

Sous Requêtes

Introduction

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

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

Sous requête renvoyant une seule ligne

- La sous requête ne renvoie qu'une seule et unique valeur
- Dans les autres cas, une erreur est générée

Exemple

- Afficher les produits conditionnés à base de sucre

ProduitsCond			
code	Poids	Volume	codePV
C012	2,3	0,69	P02
C253	1	0,25	P01
C258	2	0,4	P01
C693	2	0,45	P01

ProduitsVrac	
codePV	Désignation
P01	sucre
P02	poivre
P03	sel

- On peut effectuer une jointure

```
SELECT code FROM produitvrac, produitcond WHERE
produitvrac.codePV=produitcond.codePV AND désignation='sucre
```

Exemple ... suite

- On peut faire cela d'une autre façon

1. Chercher le code du produit dont la désignation est sucre
2. Afficher ensuite les produits conditionnés correspondant

SELECT codePV FROM produitvrac WHERE désignation = 'sucre'

- On obtient une table à une seule ligne

codePV
P01

Exemple ... suite

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

```
SELECT code FROM produitcond WHERE codePV =
(SELECT codePV FROM produitvrac WHERE désignation =
'sucre')
```

code
C253
C258
C693

Exemple ... fin

SELECT codePV FROM produitvrac
WHERE désignation = 'sucre'

→

codePV
P01

SELECT code FROM produitcond
WHERE codePV='P01';

→

code
C253
C258
C693

Sous requète renvoyant plusieurs valeurs

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

Exemple

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

SELECT codePV FROM produitvrac WHERE désignation LIKE 's%'

SELECT code from produitcond WHERE codePV IN (SELECT codePV FROM produitvrac WHERE désignation LIKE 's%')

Exemple ... suite

ProduitsCond			
code	Poids	Volume	codePV
C012	2,3	0,69	P02
C253	1	0,25	P01
C258	2	0,4	P01
C693	2	0,45	P01

ProduitsVrac	
codePV	Désignation
P01	sucre
P02	poivre
P03	sel

SELECT code from produitcond WHERE codePV IN (SELECT codePV FROM produitvrac WHERE désignation LIKE 's%')

code
C253
C258
C693

Les opérateurs ALL et ANY

- On peut étendre la 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 valeur renvoyée par la sous requête

Exemple

- v1 $>$ ALL (SELECT ... FROM ...)
- v1 $>$ ANY (SELECT ... FROM ...)

Exemple

ProduitsCond			
code	Poids	Volume	codePV
C012	2,3	0,69	P02
C253	1	0,25	P01
C258	2	0,4	P01
C693	2	0,45	PO1

$\Rightarrow$

code	Volume
C012	0.69
C258	0.4
C693	0.45

SELECT code, Volume FROM ProduitCond WHERE volume>ANY(SELECT 1.5 * volume FROM produitCond

Exemple

- Le code du produit conditionné le plus lourd
- Deux solutions

SELECT code FROM produitCond WHERE poids >= ALL (SELECT POIDS FROM produitCond)

SELECT code FROM produitCond WHERE poids = SELECT MAX(Poids) FROM ProduitCond

Sous requêtes et opérateur MINUS

- L'opérateur MINUS est l'opérateur de différence de l'algèbre relationnelle
- A 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

Exemple

- Les codes de produits en vrac qui ne sont jamais utilisé

SELECT codePV FROM produitVrac

MINUS

SELECT codePV FROM produitCond

CodePV
P01
P02
P08
P09

-

CodePV
P01
P01
P01
P02

$\longrightarrow$

CodePV
P06
P09

Exemple...suite

- Utilisation des sous requêtes si l'opérateur MINUS n'est pas implanté

SELECT code FROM ProduitCond WHERE

CodePV NOT IN (SELECT codePV FROM produitCond)

Sous requêtes et HAVING

- 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 ProduitsCond GROUP BY CodePV HAVING COUNT(*)>= ALL (SELECT COUNT(*) FROM ProduitsCond GROUP BY CodePV)

Exercices

- Toutes les informations sur la voiture la plus chère
- Immatriculations des voitures plus chères que la moyenne
- Nom et prénom de la persone qui possède le plus de voitures
- Liste des personnes qui n'ont pas de voitures
- Liste des personnes qui n'ont pas de voitures de plus de 20000 euros

Sous requêtes corrélatives

- Jusqu'à présent les sous requêtes étaient indépendantes de la requête principale
- La sous requête était exécutée et le résultat fourni était envoyé à la requête principale
- Il peut être necessaire de faire dépendre la sous requête de la requête principale
- Dans ce cas, la sous requête ne peut plus s'exécuter indépendament de la requête principale

Sous requêtes corrélatives

```
SELECT ... FROM table1
WHERE ... (SELECT ... FROM table2)
                WHERE ...(table2.col=table1.col...
```

- Les tables internes et externes sont reliés par une condition à l'intérieur de la sous requête
- Il y a corrélation entre les sources de données internes et externes

Exemple

- On rajoute à la table Commande un champ Date
- Lister La commande la plus importante de chaque mois

Commande			
numCom	code	quantité	DateCmde
CD01	C012	25	17-jan-2003
CD02	C693	14	30-dec-2002
CD03	C012	4	12-may-2001
CD04	C012	11	03-jun-2000
CD05	C693	71	14-03-2001

- Dans un premier temps, Recherchons, pour chaque mois la quantité commandée la plus importante

SELECT max(quantité) as qteMax FROM Commande GROUP BY Quantité

Exemple ... suite

- Le regroupement effectué nous empèche d'afficher le numéro de la commande correspondant
- On selectionne donc une ligne si la quantité commandée est la plus importante pour le mois considéré

Exemple ... fin

Commande			
numCom	code	quantité	DateCmde
CD01	C012	25	17-jan-2003
CD02	C693	14	30-dec-2002
CD03	C012	4	12-may-2001
CD04	C012	11	03-jun-2000
CD05	C693	71	14-03-2001

ProduitsVrac	
codePV	Désignation
P01	sucre
P02	poivre
P03	sel

SELECT c1.NumCom, month(c1.dateCmde) as mois, year(c1.dateCmde), c1.quantité

FROM Commandes c1

WHERE quantité= SELECT max(quantité) as qteMax FROM Commande c2 WHERE month(c1.dateCmde) = month(c2.dateCmde) year(c1.dateCmde) = year(c2.dateCmde)and GROUP BY Quantité

Le pédicat EXISTS

- Tester si la requête interne a renvoyé une table vide ou pas
- Recherchons la liste des produits conditionnés utilisés au moins une fois

ProduitsCond			
code	Poids	Volume	codePV
C012	2,3	0,69	P02
C253	1	0,25	P01
C258	2	0,4	P01
C693	2	0,45	P01

ProduitsVrac	
codePV	Désignation
P01	sucre
P02	poivre
P03	sel

```
SELECT CodePV FROM ProduitsVrac PV
WHERE EXISTS(SELECT * FROM ProduitsCond PC
WHERE PV.CodePV= PC.CodePV)
```

Exemple

- A l'inverse, ceux qui ne sont pas utilisés

```
SELECT CodePV FROM ProduitsVrac PV

WHERE NOT EXISTS(

SELECT * FROM ProduitsCond PC WHERE PV.CodePV=
PC.CodePV)
```

Exercices

- Immatriculation de la voiture la plus chère de chaque propriétaire
- Immatriculation des voitures plus chères que la moyenne de leur année
- Nom et prénom des personnes sans voiture
- Liste des personnes qui possèdent plus de deux voitures
- Liste des voitures plus chères qu'une autre plus récente