

Bases de données

Souhila KACI

Partie 3

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

Exemple

- Afficher pour chaque codePV, le nombre de références de produits conditionnés qui lui correspondent.
- Il faut utiliser l'opérateur d'agrégat COUNT.

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'

⇒

codePV	NbP
'P01'	3
'P02'	1

Comment ne pas faire

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'

⇒

codePV	NbP
'P01'	3
'P02'	1

~~SELECT DISTINCT codePV,COUNT(*) FROM ProduitCond;~~

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

- Une requête interrogative SQL générale :
SELECT ⟨liste d'attributs (et fonctions d'agrégat)⟩
FROM ⟨liste de tables⟩
WHERE ⟨condition (de lignes/de tuples)⟩
GROUP BY ⟨attributs de regroupement⟩
HAVING ⟨condition (de groupes)⟩
ORDER BY ⟨attributs de tri (chaque attribut suivi de DESC ou ASC)⟩
- Les clauses SELECT/FROM obligatoires.
- Si clause HAVING alors clause GROUP BY.

```
SELECT codePV, count(*) AS nbProduit FROM  
ProduitCond GROUP BY codePV;
```

- Les lignes sont regroupées par codePV identiques.

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'

Fonctionnement : 2eme étape

- Chaque ensemble de lignes est regroupé en une seule.
- Les attributs codePC, Poids, Volume ne peuvent pas être conservés.

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'

⇒

codePV
'P01'
'P02'

Fonctionnement : 3eme étape

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

SELECT codePV, count(*) **AS** nbProduit **FROM**
ProduitCond **GROUP BY** 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'

⇒

codePV	NbP
'P01'	3
'P02'	1

- 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 : Pas possible de mettre dans le SELECT des attributs (seuls, sans utilisation de fonctions d'agrégation) non participant à la clause GROUP BY
SELECT **codePV**, count(*) **AS** nbProduit, **poids**
FROM ProduitCond
GROUP BY **codePV**;
- Les fonctions d'agrégation agissent sur les sous-tables générées par le regroupement.

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

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

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

⇒

code	NbC	Qte
'C012'	3	40
'C693'	2	85

- La clause de sélection WHERE permet de ne conserver que les lignes qui correspondent au critère énoncé. Cette sélection agit **AVANT** le regroupement.
- Les lignes sont sélectionnées puis regroupées.

- 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(quantite) as Qte  
FROM Commande WHERE quantite>12 GROUP BY code;
```

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

⇒

code	NbC	Qte
'C012'	1	25
'C693'	2	85

Un petit exercice (1)

Voiture	<u>Immatriculation</u>	Marque	Annee	Prix	<u>IdProprio</u>
	'1111AA01'	'Toyota'	1997	16 000	'Id01'
	'2222BB02'	'Peugeot'	2000	31 200	'Id01'
	'3333CC03'	'Fiat'	1997	2 000	'Id03'
	'4444DD13'	'Fiat'	1995	30 300	'Id02'
	'5555EE62'	'Renault'	1997	21 000	'Id02'
	'6666FF59'	'Opel'	1999	2 900	'Id01'
	'7777ZZ75'	'Ford'	1998	22 222	'Id03'

Personnes	<u>IdProprio</u>	Nom	Prenom	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

Exercice (2)

- Afficher le nombre de voitures par identifiant de propriétaire (afficher son identifiant).
- Afficher le capital de chaque propriétaire (afficher son identifiant).
- Afficher le prix moyen par marque.
- Afficher le nombre de voitures de plus de 20000 euros pour chaque propriétaire (afficher son identifiant).
- Afficher le nombre de voitures mises en circulation après 1996 pour chaque propriétaire (afficher son identifiant).

- Avec la clause WHERE on sélectionne **d'abord** et on regroupe **après**.
- Comment faire si on veut **sélectionner après le regroupement**.

Exemple

Liste des produits conditionnés qui apparaissent dans au moins 3 commandes ?

Dans ce cas il faut utiliser le mot clé **HAVING**.

- 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.

Première étape : Regrouper

- On regroupe les lignes qui possèdent le même code.

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

- On compte le nombre de lignes regroupées.

'C012'	3
'C693'	2

- On supprime celles pour lesquelles le compte est inférieur à 2.

'C012'	3
--------	---

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

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

⇒

code	NbC
'C012'	3

- Les produits conditionnés dont la quantité totale des commandes dépasse 60.

SELECT code, sum(quantite) as NbC **FROM** Commande
GROUP BY code **HAVING** sum(quantite) > 60;

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

⇒

code	NbC
'C693'	85
'C012'	40



Regrouper plusieurs colonnes

- Les regroupements peuvent concerner plusieurs colonnes en même temps.
- Nombre de produits par client et par code.

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

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

⇒

code	CodeClient	Qte
'C012'	'CL1'	25
'C012'	'CL2'	15
'C693'	'CL1'	71
'C693'	'CL2'	14

Un petit exercice (1)

Voiture	<u>Immatriculation</u>	Marque	Annee	Prix	<u>IdProprio</u>
	'1111AA01'	'Toyota'	1997	16 000	'Id01'
	'2222BB02'	'Peugeot'	2000	31 200	'Id01'
	'3333CC03'	'Fiat'	1997	2 000	'Id03'
	'4444DD13'	'Fiat'	1995	30 300	'Id02'
	'5555EE62'	'Renault'	1997	21 000	'Id02'
	'6666FF59'	'Opel'	1999	2 900	'Id01'
	'7777ZZ75'	'Ford'	1998	22 222	'Id03'

Personnes	<u>IdProprio</u>	Nom	Prenom	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

- 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.
- Valeur moyenne des voitures par année et par marque.
- Identifiant des propriétaires de plus de 2 voitures qui coûtent plus de 20 000 euros.

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 et l'heure du système
 - `date(now())` : Date du jour
 - ...

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

Commande			
numCom	code	quantite	Date
'CD01'	'C012'	25	17/01/2003
'CD02'	'C693'	14	30/12/2002
'CD03'	'C012'	4	12/05/2001
'CD04'	'C012'	11	03/06/2000
'CD05'	'C693'	71	14/03/2001

date	jour	mois	année	diff
17/01/2003	17	01	2003	376
30/12/2002	30	12	2002	28
12/05/2001	12	05	2001	600
03/06/2000	03	06	2000	1195
14/03/2001	14	03	2001	540

Quelques compléments

Prédicat Between

- Crée un raccourci qui permet d'exprimer les conditions \leq et \geq .
- $a \text{ BETWEEN } 0 \text{ AND } 3 \Leftrightarrow a \geq 0 \text{ AND } a \leq 3$
- On peut bien sûr prendre la négation de cette condition :
NOT (a BETWEEN 0 AND 3)
- On peut l'utiliser sur des chaînes de caractères.

Quelques compléments

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 caractères quelconque

Exemple

- **LIKE** 'R__' : 3 caractères dont le premier est un R
- **LIKE** '%phone' : chaîne qui se termine par phone

Quelques compléments

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))

Quelques compléments

Opérateurs ensemblistes

- Les opérateurs classiques de l'algèbre relationnelle ont leur équivalent en langage SQL
 - UNION, INTERSECT, DIFF
- **Sur certains SGBDR, ces opérateurs ne sont pas implantés.**
- Comment simuler l'opérateur DIFF ??
- Attention, les attributs des deux tables doivent être identiques

```
SELECT ... FROM ... WHERE ...  
UNION  
SELECT ... FROM ... WHERE ...
```

Exemple

- Les produits conditionnés dont le poids est strictement inférieur à 2 ou le volume inférieur ou égal à 0.4.

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

⇒

codePC
C253
C258

SELECT codePC FROM ProduitCond WHERE Poids < 2

UNION

SELECT codePC FROM ProduitCond WHERE Volume <= 0.4;

Qu'est ce qui est préférable ?

- Lorsque l'on traite des tables contenant 100000 enregistrements, l'optimisation des requêtes devient importante !!
- Qu'est ce qui est plus rapide

```
SELECT codePC FROM ProduitCond WHERE Poids < 2  
UNION
```

```
SELECT codePC FROM ProduitCond WHERE Volume <= 0.4;
```

- Ou alors

```
SELECT codePC FROM ProduitCond WHERE Poids <2 OR  
Volume <=0.4;
```