V- Fonctions d'agrégations :

Les fonctions d'agrégation permettent de répondre aux requêtes du type :

- **"Quel est la note max dans le premier contrôle"**
- **"Quel est la note moyen des étudiants"**

L'agrégation va servir (par exemple) à regrouper les étudiants d'une même classe (ce groupe de valeurs est appelé un *agrégat*) et à effectuer une opération sur chacun des agrégats

Exemples d'opérations : min, max, somme, moyenne, comptage

Exemple : **Soit la relation Etudiant de schéma S = (Nom, Filière, note) :**

Nom	Filière	note
Yasser	DSI	12
Ilyass	BTP	15
Walid	BTP	14
Reda	DSI	17
Imane	BTP	11

$$\text{Filiere}\ \Upsilon_{\text{moyenne(note)}}(\text{Etudiant})$$

Filière	note
DSI	14.5
BTP	13.33

EN SQL, les fonctions d'agrégation s'appliquent à une colonne, en général de type numérique. Ces fonctions sont :

1. **COUNT qui compte le nombre de valeurs non nulles.**
2. **MAX et MIN.**
3. **AVG qui calcule la moyenne des valeurs de la colonne.**
4. **SUM qui effectue le cumul (somme).**

En SQL: $A1,\ldots, An\ \gamma\ f1(B1),\ldots, fm(Bm)\ (R)$ sera traduit par :

SELECT $A_1, A_2 \ldots, A_n, f_1(B_1), f_2(B_2), \ldots, f_p(B_p)$

FROM R

GROUP BY $A_1, A_2, \ldots, A_n$;

Exemples :

Filiere $\gamma_{moyenne(note)}$(Etudiant)

SELECT Filiere, **AVG(**note**)** **FROM** Etudiant **GROUP BY** Filiere;

- **Filiere $\gamma_{Max(note)}$(Etudiant)**

SELECT Filiere, **MAX(**note**)** **FROM** Etudiant **GROUP BY** Filiere;

Pour réaliser l'opération d'agrégation (regroupement des lignes d'une table par valeurs contenues dans une colonne) avec SQL, on utilise le mot clé GROUP BY suivi du nom de la colonne sur laquelle s'effectue l'agrégat.

Application :

Releve

Filière	Numéro	Elève	Note
DSI	1	Meyer	17,5
BTP	2	Martin	7,75
DSI	1	Bernard	9,25
BTP	1	Robert	14,0
BTP	2	Dubois	11,5
DSI	1	Lemaire	7,25
BTP	1	Albert	13,0
BTP	1	Garcia	16,5
BTP	2	Richard	12,5
DSI	2	Petit	15,5
BTP	1	Simon	10,5

Soit la table Releve suivante .Traduire les requêtes suivantes en opérations de l'algèbre relationnelle puis en SQL.

1 Calculer la moyenne des BTP et celle des DSI.

2 Calculer la moyenne de chaque classe.

3 Calculer la moyenne de la BTP 2 et celle de la DSI 2.

4 Sélectionner les classes dont la moyenne est supérieure ou égale à douze.

Solution

1

$_{\text{Filiere}}\Upsilon_{\text{moyenne(note)}}(\text{releve})$

Filière	moyenne(Note)
DSI	12.375
BTP	12.25

2

$_{\text{Filiere,Numero}}\Upsilon_{\text{moyenne(note)}}(\text{releve})$

Filière	Numéro	moyenne(Note)
DSI	1	11.33
DSI	2	15.5
BTP	1	13.5
BTP	2	10.6

3

$\sigma_{\textbf{\textit{Numero}}=2}(_{\text{Filiere,Numero}}\Upsilon_{\text{moyenne(note)}}(\text{releve}))$

Filière	Numéro	moyenne(Note)
DSI	2	15.5
BTP	2	10.6

4

$\sigma_{\text{moyenne(note)}\textbf{\textit{>=12}}}(_{\text{Filiere,Numero}}\Upsilon_{\text{moyenne(note)}}(\text{releve}))$

Filière	Numéro	moyenne(Note)
DSI	2	15.5
BTP	1	13.5

Requete SQL :

1-

SELECT filiere, **avg**(note) **FROM** releve **GROUP BY** filiere;

2 –

SELECT filiere, numero, **avg**(note)
FROM releve **GROUP BY** filiere, numero;

3 –

SELECT filiere, numero, **avg**(note)
FROM releve **WHERE** numero = 2 **AND** (Filiere='BTP' OR
Filiere='DSI') **GROUP BY** filiere, numero;

4 -

SELECT filiere, numero, **avg**(note) **FROM** releve
GROUP BY filiere, numero **HAVING avg**(note) >= 12;

Opérateur IN:

*L'opérateur logique **IN** dans SQL s'utilise avec la commande WHERE pour vérifier si une colonne est égale à une des valeurs comprise dans set de valeurs déterminées*

SELECT nom_colonne **FROM** nom_table **WHERE** nom_colonne **IN** (valeur1, valeur2, valeur3, …)

Exemple :

SELECT * **FROM** releve **WHERE** filiere **IN** ('DSI','BTP')

Opérateur BETWEEN :

*L'opérateur **BETWEEN** est utilisé dans une requête SQL pour sélectionner un intervalle de données dans une requête utilisant WHERE. L'intervalle peut être constitué de chaînes de caractères, de nombres ou de dates*

SELECT * **FROM** nom_table **WHERE** nom_colonne **BETWEEN** 'valeur1' **AND** 'valeur2'

Exemple :

SELECT * **FROM** releve **WHERE** note **BETWEEN** 15 **AND** 10

Opérateur LIKE :

*L'opérateur **LIKE** est utilisé dans la clause WHERE des requêtes SQL. Ce mot-clé permet d'effectuer une recherche sur un modèle particulier. Il est par exemple possible de rechercher les enregistrements dont la valeur d'une colonne commence par telle ou telle lettre. Les modèles de recherches sont multiples.*

SELECT * **FROM** nom_table **WHERE** colonne **LIKE** modele

Exemple :

- **LIKE** *'%a'* : rechercher toutes les chaines de caractère qui se termine par un «a»
- **LIKE** *'a%'*: rechercher toutes les lignes de « colonne » qui commence par un «a».
- **LIKE** *'%a%'*: rechercher tous les enregistrements qui utilisent le caractère «a».

Opérateur ORDER BY:

La commande ORDER BY permet de trier les lignes dans un résultat d'une requête SQL. Il est possible de trier les données sur une ou plusieurs colonnes, par ordre ascendant ou descendant.

SELECT * **FROM** nom_table **WHERE** nom_colonne **ORDER BY** Colonne

Exemple :

SELECT * **FROM** releve **ORDER BY** note

SELECT * **FROM** releve **ORDER BY** note **DESC**

SELECT * **FROM** releve **ORDER BY** note,eleve

Opérateur LIMIT :

La clause LIMIT est à utiliser dans une requête SQL pour spécifier le nombre maximum de résultats que l'ont souhaite obtenir. Cette clause est souvent associé à un OFFSET, c'est-à-dire effectuer un décalage sur le jeu de résultat

Exemple :

 # récupérer les 10 premier tuples

SELECT * **FROM** releve **LMIT** 10

 # récupérer les résultats de 6 à 15

SELECT * **FROM** releve **LIMIT 10 OFFSET 5**

 # On peut combiner ORDER BY et LIMIT

SELECT * **FROM** releve **ORDER BY** note **LIMIT** 10