

SQL 3 : Fonctions d'agrégation et GROUP BY

Quentin Fortier

Fonctions d'agrégations

Fonctions s'appliquant sur un attribut a :

- **MAX**(a) : maximum de a parmi les enregistrements
- **MIN**(a) : minimum de a parmi les enregistrements
- **SUM**(a) : somme de a parmi les enregistrements
- **AVG**(a) : moyenne de a parmi les enregistrements
- **COUNT**(*) : nombre total d'enregistrements

Exemples

- ❶ livre (titre : CHAR(50), auteur : CHAR(50), pages : INT)
- ❷ emprunteur (id : INT, nom : CHAR(50))
- ❸ emprunt (id_emprunteur : INT, titre_livre : CHAR(50))

Question

Comment obtenir le nombre moyen de pages d'un livre?

```
SELECT AVG(pages) FROM livre;
```

Exemples

- ❶ livre (titre : CHAR(50), auteur : CHAR(50), pages : INT)
- ❷ emprunteur (id : INT, nom : CHAR(50))
- ❸ emprunt (id_emprunteur : INT, titre_livre : CHAR(50))

Question

Comment obtenir le nombre de livres empruntés par M. Machin?

```
SELECT COUNT(*) FROM emprunt  
JOIN emprunteur ON id = id_emprunteur  
WHERE nom = 'Machin';
```

Exemples

Dans eleve (nom, classe, note, ...), comment obtenir la note maximum dans le classe de PC?

```
SELECT MAX(note)
FROM eleve
WHERE classe = 'PC';
```

Exemples

Étant données les tables

`planete(nom, rayon, poids, nom_etoile)` et

`etoile(nom, galaxie)` comment obtenir la somme des poids des planètes de la Voie lactée?

Question 13 On dispose d'une table de données dont le schéma de relation est le suivant :

candidats(identifiant, nom, prénom, rang, moyenne, age, adresse),

dont les attributs sont respectivement l'identifiant des candidats, leur nom, leur prénom, leur rang au concours, leur moyenne, leur âge et leur adresse postale. La requête SQL suivante :

```
SELECT COUNT(*) FROM candidats WHERE moyenne>10
```

- ☒ permet de lister tous les candidats de la table candidats.
- ☒ renvoie le nombre de candidats dont la moyenne est supérieure ou égale à 10.
- ☒ permet de regrouper les candidats ayant la même moyenne.
- ☒ provoque une erreur.

```
SELECT ... FROM ... GROUP BY attribut;
```

a pour effet de grouper les résultats par même attribut.

Il y a un résultat affiché pour chaque valeur possible de attribut.

Les fonctions d'agrégations dans le **SELECT** s'appliquent alors à chaque groupe.

Examples

```
SELECT Continent, SUM(Population)
FROM Country
GROUP BY Continent;
```

Continent	SUM(Population)
Asia	3705025700
Europe	730074600
North America	482993000
Africa	784475000
Oceania	30401150
Antarctica	0
South America	345780000

Remarques

Attention : Avec un **GROUP BY** a, il ne faut pas afficher d'attribut, à part a.

```
SELECT Continent, Name, SUM(Population)
FROM Country
GROUP BY Continent;
```

Continent	Name	SUM(Population)
Asia	Afghanistan	3705025700
Europe	Albania	730074600
North America	Aruba	482993000
Africa	Angola	784475000
Oceania	American Samoa	30401150
Antarctica	Antarctica	0
South America	Argentina	345780000

Exemples

Comment afficher la densité de population de chaque continent?

```
SELECT Continent, SUM(Population) / SUM(Surface)
FROM Country
GROUP BY Continent;
```

Exemples

Comment afficher chaque continent trié par ordre décroissant de densité de population?

```
SELECT Continent, SUM(Population)/SUM(Surface) AS densite  
FROM Country  
GROUP BY Continent  
ORDER BY densite DESC;
```

Exemples

Dans eleve (nom, classe, ...), comment afficher chaque classe avec son nombre d'élèves?

```
SELECT classe, COUNT(*) FROM eleve GROUP BY classe;
```

+-----+-----+	
classe	COUNT(*)
+-----+-----+	
MPSI1	41
MPSI2	38
PCSI1	40
PCSI2	37
+-----+-----+	

Exemples

Dans la table `eleve(nom, classe, note, ...)`, comment afficher la moyenne, note maximum et note minimum de chaque classe?

```
SELECT classe, AVG(note), MAX(note), MIN(note)
FROM eleve
GROUP BY classe;
```

Exemples

Étant données les tables

`planete(nom, rayon, poids, nom_etoile)` et

`etoile(nom, galaxie)` comment obtenir, pour chaque étoile, le nombre de planètes tournant autour?

HAVING

Lorsque l'on groupe des enregistrements avec **GROUP BY**, on peut afficher seulement les groupes vérifiant une condition avec **HAVING**.

WHERE sert à établir une condition sur les **enregistrements** affichés.

HAVING sert à établir une condition sur les **groupes** affichés.

HAVING ne peut être utilisé qu'à la suite d'un **GROUP BY**.

Exemples

Dans la table `eleve(nom, classe, ...)`, comment afficher que les classes avec au moins 40 élèves?

```
SELECT classe, COUNT(*)  
FROM eleve  
GROUP BY classe  
HAVING COUNT(*) >= 40;
```

Exemples

Dans eleve (nom, classe, note, ...), comment afficher que les classes dont la moyenne est ≥ 12 ?

```
SELECT classe
FROM eleve
GROUP BY classe
HAVING AVG(note) >= 12;
```

Exemples

- ❶ livre (titre CHAR(50), auteur CHAR(50), pages INT)
- ❷ emprunteur (id INT, nom CHAR(50))
- ❸ emprunt (id_emprunteur INT, titre_livre CHAR(50))

Comment afficher les noms des personnes ayant emprunté au moins 5 livres?

```
SELECT nom FROM emprunteur
JOIN emprunt ON id = id_emprunteur
GROUP BY nom
HAVING COUNT(*) >= 5;
```