Les Requêtes SQL de Type SELECT

Les requêtes SQL de type `SELECT` permettent d'extraire des données d'une base de données. Elles constituent le cœur de la manipulation des données relationnelles. Ce cours détaille les principales fonctionnalités de `SELECT` avec des exemples concrets en SQLite et MySQL, tout en mettant en lumière les différences de syntaxe lorsque c'est pertinent.

Structure générale d'une requête SELECT

La structure de base d'une requête `SELECT` est la suivante :

```sql
SELECT colonne(s)
FROM table
[WHERE condition]
[GROUP BY colonne(s)]
[HAVING condition]
[ORDER BY colonne(s) [ASC|DESC]]
[LIMIT n OFFSET m];
```

Sélection explicite de colonnes et SELECT *

`SELECT` permet de choisir les colonnes que vous souhaitez récupérer. Il est possible de sélectionner toutes les colonnes avec `SELECT *` ou de spécifier uniquement les colonnes nécessaires.

Exemple de sélection explicite

```sql
SELECT nom, age
FROM utilisateurs;
```

Ce type de requête permet d'optimiser les performances en ne récupérant que les colonnes nécessaires, contrairement à :

Exemple avec SELECT *

```sql
SELECT *
FROM utilisateurs;
```

Cette requête récupère toutes les colonnes de la table, ce qui peut être lourd si la table contient de nombreuses colonnes ou si seules certaines données sont nécessaires.

Résultat attendu pour SELECT *

id	nom	age	email
1	Alice	25	alice@example.com
2	Bob	30	bob@example.com

Bonnes pratiques

Même si `SELECT *` est pratique pour des tests rapides, privilégiez une sélection explicite dans un environnement de production pour éviter de manipuler des données inutiles.

Alias de colonnes

Un alias est un nom temporaire donné à une colonne ou une valeur calculée dans une requête. Cela permet d'améliorer la lisibilité et la compréhension des résultats.

Syntaxe

Un alias s'applique en utilisant le mot-clé `AS` ou simplement un espace après le nom ou l'expression.

```sql
SELECT colonne AS alias
FROM table;
```

Exemple avec alias pour une colonne

```sql
SELECT nom AS utilisateur, age AS âge
FROM utilisateurs;
```

Résultat attendu :

utilisateur	âge
Alice	25
Bob	30

Exemple avec alias pour une valeur calculée

Les alias sont particulièrement utiles pour nommer des colonnes calculées, comme les résultats des fonctions d'agrégation.

```sql
SELECT SUM(age) AS total_age, AVG(age) AS moyenne_age
FROM utilisateurs;
```

Résultat attendu :

total_age	moyenne_age
55	27.5

Exemple sans `AS`

Dans MySQL et SQLite, l'utilisation de `AS` est facultative. L'alias peut être appliqué en plaçant un espace après la colonne ou l'expression.

```sql
SELECT nom utilisateur, age âge
FROM utilisateurs;
```

Cependant, pour des raisons de clarté et de cohérence, l'utilisation explicite de `AS` est recommandée.

La clause WHERE

La clause `WHERE` permet de spécifier des conditions pour filtrer les lignes retournées.

Exemple

```sql
SELECT nom, age
FROM utilisateurs
WHERE age > 25;
```

Résultat attendu :

nom	age
Bob	30

Les opérateurs de comparaison

Les opérateurs permettent de définir des conditions dans la clause `WHERE`.

Opérateur	Description	Exemple
=	Égal	`age = 30`
!= ou <>	Différent	`age != 30`
>	Supérieur	`age > 25`
<	Inférieur	`age < 30`
>=	Supérieur ou égal	`age >= 25`
<=	Inférieur ou égal	`age <= 30`

Combiner des conditions : AND et OR

- `AND` : Toutes les conditions doivent être vraies.
- `OR` : Une seule condition doit être vraie.

Exemple

```sql
SELECT nom, email
FROM utilisateurs
WHERE age > 25 AND email LIKE '%example.com';
```

Résultat attendu :

nom	email
Bob	bob@example.com

Rechercher des modèles : LIKE

`LIKE` est utilisé pour rechercher des motifs dans des chaînes de caractères.

Modèle	Description
`%`	Remplace 0 ou plusieurs caractères.
`_`	Remplace un seul caractère.

Exemple

```sql
SELECT nom
FROM utilisateurs
WHERE email LIKE '%@example.com';
```

Résultat attendu :

nom
Alice
Bob

Trier les résultats : ORDER BY

`ORDER BY` permet de trier les résultats en ordre croissant (`ASC`) ou décroissant (`DESC`).

Exemple

```sql
SELECT nom, age
FROM utilisateurs
ORDER BY age DESC;
```

Résultat attendu :

nom	age
Bob	30
Alice	25

Calculer des totaux : SUM, AVG, MIN, MAX, COUNT

Les fonctions d'agrégation permettent de réaliser des calculs sur des ensembles de données.

Fonction	Description
`SUM`	Somme des valeurs
`AVG`	Moyenne des valeurs
`MIN`	Valeur minimale

Fonction	Description
MAX	Valeur maximale
COUNT	Nombre de lignes correspondantes

Exemple avec alias

```sql
SELECT COUNT(*) AS total_utilisateurs, AVG(age) AS moyenne_age
FROM utilisateurs;
```

Résultat attendu :

total_utilisateurs	moyenne_age
2	27.5

Exemples de requêtes simples sur la base des élections

1. Afficher les codes et noms des États.

```sql
SELECT code AS code_etat, nom AS nom_etat
FROM Etats;
```

2. Trouver les États ayant plus de 10 grands électeurs.

```sql
SELECT code AS code_etat, nom AS nom_etat
FROM Etats
WHERE grd > 10;
```

3. Calculer la somme des grands électeurs avec un alias.

```sql
SELECT SUM(grd) AS total_grands_electeurs
FROM Etats;
```

4. Afficher les noms des États triés par grands électeurs décroissants.

```sql
SELECT nom AS nom_etat, grd AS grands_electeurs
FROM Etats
ORDER BY grd DESC;
```

5. Afficher les États où le vainqueur est défini.

```sql
SELECT code AS code_etat, nom AS nom_etat
FROM Etats
WHERE vainqueur IS NOT NULL;
```

6. Compter le nombre total d'États.

```sql
SELECT COUNT(*) AS total_etats
FROM Etats;
```

7. Trouver les États avec un vainqueur différent en 2020.

```sql
SELECT code AS code_etat, nom AS nom_etat
FROM Etats
WHERE vainqueur != vainqueur_2020;
```

8. Afficher les États ayant un champ libre défini.

```sql
SELECT code AS code_etat, nom AS nom_etat, champlibre AS description
FROM Etats
WHERE champlibre IS NOT NULL;
```

9. Calculer la moyenne des grands électeurs par État.

```sql
SELECT AVG(grd) AS moyenne_grands_electeurs
FROM Etats;
```

10. Afficher les États ayant exactement 3 grands électeurs.

```sql
SELECT code AS code_etat, nom AS nom_etat
FROM Etats
WHERE grd = 3;
```

Ces exemples montrent l'utilité des alias pour rendre les colonnes ou les résultats calculés plus lisibles dans les résultats des requêtes.