

Mémento sur les bases de SQL

SQL, ou *Structured Query Language* (*langage de requête structuré*), est un langage qui permet de communiquer avec les bases de données. Il permet de sélectionner des données spécifiques et de construire des rapports complexes. Aujourd'hui, SQL est un langage universel de données. Il est utilisé dans pratiquement toutes les technologies qui traitent des données.

ÉCHANTILLON DE DONNÉES

PAYS				
id	nom	population	superficie	
1	France	66600000	640680	
2	Allemagne	80700000	357000	
...

VILLE				
id	nom	pays_id	population	note
1	Paris	1	2243000	5
2	Berlin	2	3460000	3
...

INTERROGATION D'UNE TABLE UNIQUE

Récupérer toutes les colonnes de la table pays :
SELECT *
FROM pays;

Récupérer les colonnes id et nom de la table ville :
SELECT id, nom
FROM ville;

Récupérer les noms de villes triés par la colonne note dans l'ordre ASCending par défaut :
SELECT nom
FROM ville
ORDER BY note **[ASC]**;

Récupérer les noms de villes triés par la colonne note dans l'ordre DESCendant :
SELECT nom
FROM ville
ORDER BY note **DESC**;

ALIAS

COLONNES

SELECT nom **AS** nom_ville
FROM ville;

TABLEAUX

SELECT pa.nom, vi.nom
FROM ville **AS** vi
JOIN pays **AS** pa
ON vi.pays_id = pa.id;

FILTRAGE DES RÉSULTATS

OPÉRATEURS DE COMPARAISON

Récupérer les noms des villes dont l'évaluation est supérieure à 3 :
SELECT nom
FROM ville
WHERE note > 3;

Récupérer les noms des villes qui ne sont ni Berlin ni Madrid :
SELECT nom
FROM ville
WHERE nom != 'Berlin'
AND nom != 'Madrid';

OPÉRATEURS DE TEXTE

Recherche des noms de villes qui commencent par un "P" ou se terminent par un "s" :
SELECT nom
FROM ville
WHERE nom **LIKE** 'P%'
OR nom **LIKE** '%s';

Recherche des noms de villes qui commencent par une lettre quelconque suivie de "ublin" (comme Dublin en Irlande ou Lublin en Pologne) :
SELECT nom
FROM ville
WHERE nom **LIKE** '_ublin';

AUTRES OPÉRATEURS

Récupérer les noms des villes dont la population est comprise entre 500 000 et 5 millions :
SELECT nom
FROM ville
WHERE population **BETWEEN** 500000 **AND** 5000000;

Récupérer les noms des villes qui n'ont pas de valeur d'évaluation :
SELECT nom
FROM ville
WHERE note **IS NOT NULL**;

Récupérer les noms des villes situées dans des pays dont l'identifiant est 1,4,7 ou 8 :
SELECT nom
FROM ville
WHERE pays_id **IN** (1, 4, 7, 8);

INTERROGATION DE PLUSIEURS TABLES

INNER JOIN

JOIN (ou explicitement **INNER JOIN**) renvoie les lignes dont les valeurs correspondent dans les deux tables.

SELECT ville.nom, pays.nom
FROM ville
[INNER] JOIN pays
ON ville.pays_id = pays.id;

VILLE		PAYS		
id	nom	pays_id	id	nom
1	Paris	1	1	France
2	Berlin	2	2	Allemagne
3	Varsovie	4	3	Islande

LEFT JOIN

LEFT JOIN renvoie toutes les lignes de la table de gauche avec les lignes correspondantes de la table de droite. S'il n'y a pas de ligne correspondante, les valeurs renvoyées de la deuxième table sont **NULL**.

SELECT ville.nom, pays.nom
FROM ville
LEFT JOIN pays
ON ville.pays_id = pays.id;

VILLE		PAYS		
id	nom	pays_id	id	nom
1	Paris	1	1	France
2	Berlin	2	2	Allemagne
3	Varsovie	4	NULL	NULL

RIGHT JOIN

RIGHT JOIN renvoie toutes les lignes de la table de droite avec les lignes correspondantes de la table de gauche. S'il n'y a pas de ligne correspondante, les valeurs renvoyées de la deuxième table sont **NULL**.

SELECT ville.nom, pays.nom
FROM ville
RIGHT JOIN pays
ON ville.pays_id = pays.id;

VILLE		PAYS		
id	nom	pays_id	id	nom
1	Paris	1	1	France
2	Berlin	2	2	Allemagne
NULL	NULL	NULL	3	Islande

AGRÉGATION ET REGROUPEMENT

GROUP BY **regroupe** les lignes qui ont les mêmes valeurs dans les colonnes spécifiées. Il génère des résumés (agrégats) pour chaque combinaison unique de valeurs.

VILLE		
id	nom	pays_id
1	Paris	1
101	Marseille	1
102	Lyon	1
2	Berlin	2
103	Hambourg	2
104	Munich	2
3	Varsovie	4
105	Cracovie	4



VILLE	
pays_id	count
1	3
2	3
4	2

FONCTIONS D'AGRÉGATION

- avg (expr) – valeur moyenne des lignes du groupe
- count (expr) – nombre de valeurs pour les lignes du groupe
- max (expr) – valeur maximale dans le groupe
- min (expr) – valeur minimale dans le groupe
- sum (expr) – somme des valeurs du groupe

EXEMPLES DE REQUÊTES

Trouver le nombre de villes :
SELECT **COUNT**(*)
FROM ville;

Trouver le nombre de villes dont l'évaluation n'est pas nulle :
SELECT **COUNT**(note)
FROM ville;

Déterminer le nombre de valeurs distinctes pour les pays :
SELECT **COUNT**(**DISTINCT** pays_id)
FROM ville;

Trouver les pays avec la plus petite et la plus grande population :
SELECT **MIN**(population), **MAX**(population)
FROM pays;

Déterminer la population totale des villes dans les pays respectifs :
SELECT pays_id, **SUM**(population)
FROM ville
GROUP BY pays_id;

Déterminez l'évaluation moyenne des villes dans les pays respectifs si la moyenne est supérieure à 3,0 :
SELECT pays_id, **AVG**(note)
FROM ville
GROUP BY pays_id
HAVING **AVG**(note) > 3.0;

SOUS-REQUÊTES

Une sous-requête est une requête imbriquée dans une autre requête ou dans une autre sous-requête. Il existe différents types de sous-requêtes.

VALEUR UNIQUE

La sous-requête la plus simple renvoie exactement une colonne et une ligne. Elle peut être utilisée avec les opérateurs de comparaison =, <, <=, > ou >=.

Cette requête permet de trouver les villes ayant la même évaluation que Paris :

```
SELECT nom
FROM ville
WHERE note = (
    SELECT note
    FROM ville
    WHERE nom = 'Paris'
);
```

VALEURS MULTIPLES

Une sous-requête peut également renvoyer plusieurs colonnes ou plusieurs lignes. Ces sous-requêtes peuvent être utilisées avec les opérateurs IN, EXISTS, ALL, ou ANY.

Cette requête permet de trouver des villes dans des pays dont la population est supérieure à 20 millions d'habitants :

```
SELECT nom
FROM ville
WHERE pays_id IN (
    SELECT pays_id
    FROM pays
    WHERE population > 20000000
);
```

SOUS-REQUÊTE CORRÉLÉE

Une sous-requête corrélée fait référence aux tables introduites dans la requête externe. Une sous-requête corrélée dépend de la requête externe. Elle ne peut pas être exécutée indépendamment de cette dernière.

Cette requête recherche les villes dont la population est supérieure à la population moyenne du pays :

```
SELECT *
FROM ville ville_principale
WHERE population > (
    SELECT AVG(population)
    FROM ville ville_moyenne
    WHERE ville_moyenne.pays_id = ville_principale.pays_id
);
```

Cette requête recherche les pays qui ont au moins une ville :

```
SELECT nom
FROM pays
WHERE EXISTS (
    SELECT *
    FROM ville
    WHERE pays_id = pays.id
);
```

OPÉRATIONS D'ENSEMBLE

Les opérations ensemblistes sont utilisées pour combiner les résultats de deux ou plusieurs requêtes en un seul résultat. Les requêtes combinées doivent renvoyer le même nombre de colonnes et des types de données compatibles. Les noms des colonnes correspondantes peuvent être différents.

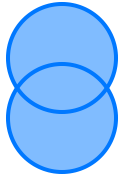
CYCLISME			PATINAGE		
id	nom	pays	id	nom	pays
1	YK	DE	1	YK	DE
2	ZG	DE	2	DF	DE
3	WT	PL	3	AK	PL
...

UNION

UNION combine les résultats de deux ensembles de résultats et supprime les doublons. **UNION ALL** ne supprime pas les lignes en double.

Cette requête affiche les cyclistes allemands et les patineurs allemands :

```
SELECT nom
FROM cyclisme
WHERE pays = 'DE'
UNION / UNION ALL
SELECT nom
FROM patinage
WHERE pays = 'DE';
```

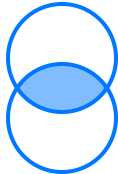


INTERSECT

INTERSECT ne renvoie que les lignes qui apparaissent dans les deux ensembles de résultats.

Cette requête affiche les cyclistes allemands qui sont également des patineurs allemands :

```
SELECT nom
FROM cyclisme
WHERE pays = 'DE'
INTERSECT
SELECT nom
FROM patinage
WHERE pays = 'DE';
```



EXCEPT

EXCEPT renvoie uniquement les lignes qui apparaissent dans le premier ensemble de résultats mais qui n'apparaissent pas dans le deuxième ensemble de résultats.

Cette requête affiche les cyclistes allemands sauf s'ils sont également patineurs allemands :

```
SELECT nom
FROM cyclisme
WHERE pays = 'DE'
EXCEPT / MINUS
SELECT nom
FROM patinage
WHERE pays = 'DE';
```

