

Résumé SQL

Le langage SQL (Structured Query Language) permet de manipuler et interroger des bases de données relationnelles. Voici un résumé des principales commandes et concepts de base.

1. La clause **SELECT**

La commande SELECT permet de sélectionner des données depuis une table.

Exemple :

- `SELECT nom, age FROM utilisateurs;`

2. La clause **WHERE**

WHERE permet de filtrer les résultats selon une condition.

Exemple :

- `SELECT * FROM utilisateurs WHERE age > 18;`

3. La clause **ORDER BY**

ORDER BY permet de trier les résultats d'une requête dans un ordre croissant, si vous voulez mettre l'ordre décroissant, vous ajoutez DESC

Exemple :

- `SELECT * FROM utilisateurs ORDER BY age DESC;`

4. La clause **GROUP BY**

GROUP BY regroupe les résultats selon une ou plusieurs colonnes, souvent utilisée avec des fonctions d'agrégation comme COUNT().

Exemple :

- `SELECT ville, COUNT(*) FROM utilisateurs GROUP BY ville;`

5. Fonctions d'agrégation : **COUNT, SUM, AVG**

- `COUNT()` : compte le nombre d'éléments retourné par la requête.
- `SUM()` : calcule la somme d'une colonne numérique.
- `AVG()` : calcule la moyenne d'une colonne numérique.

Exemples :

- `SELECT COUNT(*) FROM utilisateurs;`

- SELECT SUM(salaire) FROM employes;
- SELECT AVG(age) FROM utilisateurs;

6. Les Jointures

Les jointures permettent de relier plusieurs tables entre elles grâce à **des colonnes communes**.

- **INNER JOIN** : retourne **uniquement** les enregistrements qui ont une **correspondance** dans les **deux** tables A et B (comme $A \cap B$).

Exemple :

- SELECT utilisateurs.nom, commandes.id_commande FROM utilisateurs INNER JOIN commandes ON utilisateurs.id = commandes.id_utilisateur;
- **LEFT JOIN** : renvoie **tous** les enregistrements de la table de **gauche**, ainsi que les enregistrements correspondants de la table de droite lorsqu'ils existent. S'il n'y a aucune correspondance dans la table de droite, les valeurs de celle-ci seront remplacées par **NULL**.

Exemple :

- SELECT utilisateurs.nom, commandes.id_commande FROM utilisateurs LEFT JOIN commandes ON utilisateurs.id = commandes.id_utilisateur;

7. Création d'une table avec clé primaire

Exemple :

- CREATE TABLE utilisateurs (
id INT **PRIMARY KEY**,
nom TEXT,
age INT
);
- Ou
CREATE TABLE utilisateurs (
id INT,
nom TEXT ,
age INT,
PRIMARY KEY(id)
);

NB : si la table est déjà créée sans une indication de clé primaire, on **ajoute une contrainte de clé primaire** sur la table :

- **ALTER TABLE** utilisateurs
ADD CONSTRAINT pk_utilisateur
PRIMARY KEY (id) ;

8. Ajout d'une colonne dans une table en tant qu'une clé étrangère

Deux cas possible :

Cas1 (plus simple) : On ajoute la colonne dans la table et on dit qu'elle est une clé étrangère qui fait référence à numéro_téléphonique dans la table utilisateur :

- **ALTER TABLE** offre **ADD COLUMN** num_tel TEXT **REFERENCES** utilisateur(numéro_téléphonique);
- Cas2 : on **ajoute la colonne** sans dire qu'elle est une clé étrangère, puis on **ajoute une contrainte** pour dire que la colonne ajoutée est une clé étrangère :

- **ALTER TABLE** offre **ADD COLUMN** num_tel TEXT ;
- **ALTER TABLE** offre
ADD CONSTRAINT fk_utilisateur
FOREIGN KEY (id_utilisateur)
REFERENCES utilisateurs(id);

Conclusion

Ces commandes constituent les bases essentielles du SQL. En les maîtrisant, vous pourrez interroger, modifier et relier efficacement les données dans vos bases de données relationnelles.