Éléments médiatiques et bases de données

7. Bases du langage SQL

Les 4 requêtes de base

- Le langage SQL est relativement complexe et possède beaucoup de requêtes et de subtilités, mais à la base, on utilise principalement 4 types de requêtes.
  - Les requêtes de sélection (SELECT)
  - Les insertions (INSERT)
  - Les suppressions (DELETE)
  - Les mises à jour (UPDATE)

La commande SELECT

- La commande SELECT constitue, à elle-seule, le langage permettant d'effectuer une *requête*, une interrogation vers la base de données.
- Elle permet de :
  - sélectionner certaines colonnes d'une table ;
  - sélectionner certaines rangées dans une table en fonction de leur contenu ;
  - combiner des informations venant de plusieurs tables (requêtes avec jointures, à voir plus tard) ;
  - combiner toutes ces différentes opérations entre elles.
- Une REQUÊTE (ou interrogation) est donc une combinaison d'opérations portant sur des tables, et dont le résultat est lui-même une table temporaire (qui n'existe que le temps de la requête).

Syntaxe simplifiée de SELECT

- SELECT nom_colonne FROM nom_table WHERE condition
- Elle est constituée de 3 clauses. La clause SELECT, qui permet de spécifier quelles colonnes on veut obtenir.
- La clause FROM, qui permet de spécifier la table sur laquelle notre requête va porter.
- La clause WHERE, qui énonce une condition que doivent respecter les rangées sélectionnées.

La clause SELECT

- Elle permet de spécifier les attributs que l'on désire voir apparaître dans le résultat de la requête.
  - Pour préciser explicitement les colonnes (attributs) que l'on désire conserver, il faut les lister en les séparant par une virgule.
    - Ex : SELECT nomAuteur, prenomAuteur FROM Auteurs
  - Pour spécifier que l'on veut obtenir TOUTES les colonnes, on peut utiliser l'opérateur *.
    - Ex : SELECT * FROM Auteurs
- L'opérateur DISTINCT permet de spécifier que l'on ne veut qu'une seule version de la rangée si des doublons se présentent.
  - Ex : SELECT DISTINCT prenomAuteur from Auteurs

La clause SELECT (suite)

- Il est possible d'utiliser les opérations mathématique de base dans la clause SELECT.
  - Pour avoir le salaire annuel d'un employé dont le salaire mensuel est stocké dans la base de données, on pourrait donc faire :

    SELECT nomEmploye, prenomEmploye, salaireEmploye * 12 from Employes
- Il est possible de renommer une colonne en utilisant l'opérateur AS pour qu'elle s'affiche sous un autre nom que celui de la table originale. Ceci aura d'autres impacts que nous verrons plus tard.
  - SELECT salaireMensuel * 12 AS salaireAnnuel from Employes
- Il est possible de faire une concaténation de chaînes avec la fonction CONCAT.
  - SELECT CONCAT(nomEmploye, ' ' , prenomEmploye) AS nomComplet from Employes
- Une des opérations un peu plus complexe qui est très utile consiste à appliquer la fonction COUNT sur les colonnes que vous voulez tester pour obtenir le nombre de rangées retournées. Parfois, c'est tout ce qui vous intéresse.
  - SELECT COUNT(*) from EMPLOYES

La clause FROM

- La clause FROM spécifie la ou les tables sur lesquelles porte la requête. Elle devient surtout complexe lorsque les requêtes se font avec des jointures. Nous étudierons cette clause plus en détail dans un cours futur. Pour l'instant, nous l'utiliserons pour spécifier le nom de la table dans laquelle nous allons chercher les informations.

La clause WHERE

- La clause WHERE permet de filtrer les résultats en imposant une condition à remplir pour qu'ils soient présents dans le résultat de la requête.
- La syntaxe est : WHERE expression, tout simplement.
- Le filtrage le plus simple que l'on peut effectuer est constitué d'une expression simple.
  - Les expressions simples utilisent les opérateurs suivant pour effectuer une comparaison :
    - =
    - != (différent)
    - <, >, <=, >= (comparaisons)
    - IN (ensemble de valeurs) et NOT IN
    - BETWEEN
    - LIKE
    - IS NULL, IS NOT NULL

Expression composée

- Il existe aussi des expressions composée, lorsque l'on veut effectuer plusieurs comparaisons, ou modifier le résultat d'une comparaison.
- Les opérateurs pour créer ces opérations composées sont :
  - AND
  - OR
  - NOT
- Ex : SELECT nomEmploye WHERE salaire * 12 > 1 000 000 AND nomEmploye != 'Harvey'

Une autre clause : la clause ORDER BY

- Il est possible d'utiliser la clause ORDER BY pour ordonner nos résultats. Cette clause va **toujours** à la fin de la requête, même lorsque nous aurons des groupes (voir plus tard).
- On l'utilise comme suit :
  - ORDER BY expression, suivi de ASC pour ascendant ou DESC pour descendant.
- Ex : SELECT prenomEmploye, nomEmploye, salaireEmploye from Employes order by salaireEmploye ASC;
  - Retourne le prenom de tous les employés, leurs noms et leurs salaires, le tout trié par ordre ascendant de salaire, le plus petit salaire en premier jusqu'au plus haut salaire en dernier.

Insertions SQL

- Pour insérer des données dans la base de données, on utilise le INSERT STATEMENT.
- Syntaxe : INSERT INTO table_name
  VALUES (value1, value2, value3,...)
- Il est possible d'utiliser le STATEMENT sans spécifier *toutes les colonnes* lorsque des valeurs par défaut et des colonnes qui permettent l'ajout de NULL sont présentes.
- Syntaxe : INSERT INTO table_name (colonne1, colonne2, colonne3, …)
  VALUES (value1, value2, value3,...)

Modifications sur les données

- Pour effectuer des modifications sur des rangées existantes, on utilise le UPDATE STATEMENT.
- Syntaxe : UPDATE table_name
  SET column1=value, column2=value2,...
  WHERE some_column=some_value
- Dans la plupart des cas, pour les updates, on utilise la clause WHERE avec la colonne de la clé primaire. Cependant, nous utiliserons parfois la clause WHERE de la même façon que dans les SELECT.

Suppression de données

- Pour supprimer des rangées, on utilise le DELETE STATEMENT.
- Syntaxe : DELETE FROM table_name WHERE some_column=some_value
- Pour la clause WHERE, même chose que pour les updates et les select.