BASE DE DONNÉES – MANIPULATION DE DONNÉES

Data Manipulation

Les principales commandes de manipulation de données sont les suivantes :

INSERT => Ajouter des données dans une table

SELECT => Consulter des données dans une table

UPDATE => Modifier des données dans une table

DELETE => Supprimer des données dans une table

Data Manipulation

Insérer des données dans une table :

INSERT [INTO] table_name [(column_name, ...)] VALUES ({expr | DEFAULT}, ...)

Les nombres de column_name et d'expr doivent correspondre.

Pour les champs ayant l'option AUTO_INCREMENT, il est possible soit de ne pas préciser le champ dans la liste soit de lui passer la valeur NULL pour laisser à la charge du système d'attribuer automatiquement la valeur requise au champ.

Le fait de préciser les champs est optionnel. Si ceux-ci sont préciser, il est cependant nécessaire de placer les valeurs dans le même ordre. Si ils ne sont pas précisés, il faut donner une valeur pour chaque champ dans l'ordre de la table.

Data Manipulation

Lire des données dans une table :

SELECT [DISTINCT] select_expression, … FROM table_name
[WHERE where_expression]
[ORDER BY { nom_colonne } [ASC | DESC] , …]
[LIMIT [offset,] lignes]

Le « select_expression » indique la ou les colonnes à lire, ou même une valeur calculée. Par exemple : concat(prenom, ' ', nom) AS fullname va récupérer un champ virtuel « fullname » composé du prénom et du nom séparés par un espace.

Si deux champs ont le même nom (age) dans deux tables différentes (table1, table2) on pourra les distinguer au sein d'une même requête comme étant table1.age et table2.age

Data Manipulation

Le DISTINCT permet de ne lire que les valeurs distinctes.

Le FROM permet de sélectionner les tables qui doivent être lues pour exécuter la requête

Le ORDER BY permet de trier le résultat de la requête, de l'ordonner de manière croissante (ASC) ou décroissante (DESC)

Le WHERE permet de préciser les critères de recherche et d'associer les tables entre elles

Tous les opérateurs de comparaisons de base sont supportés (=,!=, ≤, =>, <>, IN, NOT IN, IS NULL, IS NOT NULL, ...)

Data Manipulation

Il est également possible de donner des critères « approximatifs » sur les chaînes de caractères avec le mot clé LIKE :

prenom LIKE '%a%'

Cette expression signifie « n'importe quoi suivi d'un a, suivi de n'importe quoi ».

Par exemple pour sortir uniquement les personnes qui s'appellent Jean :

```
SELECT * FROM personnes \
        WHERE prenom='jean' ;
```

```
+--------+---------------+
| Field  | Type          |
+--------+---------------+
| id     | int(11)       |
| prenom | varchar(100)  |
| nom    | varchar(100)  |
| mail   | varchar(200)  |
| AGE    | decimal(10,0) |
+--------+---------------+
```

Data Manipulation

Sur la même base, il est possible de ne sortir que certains champs (nom et prénom) concaténés en les séparant par un espace :

SELECT concat(prenom, ‘ ‘, nom) as FullName FROM personnes ;

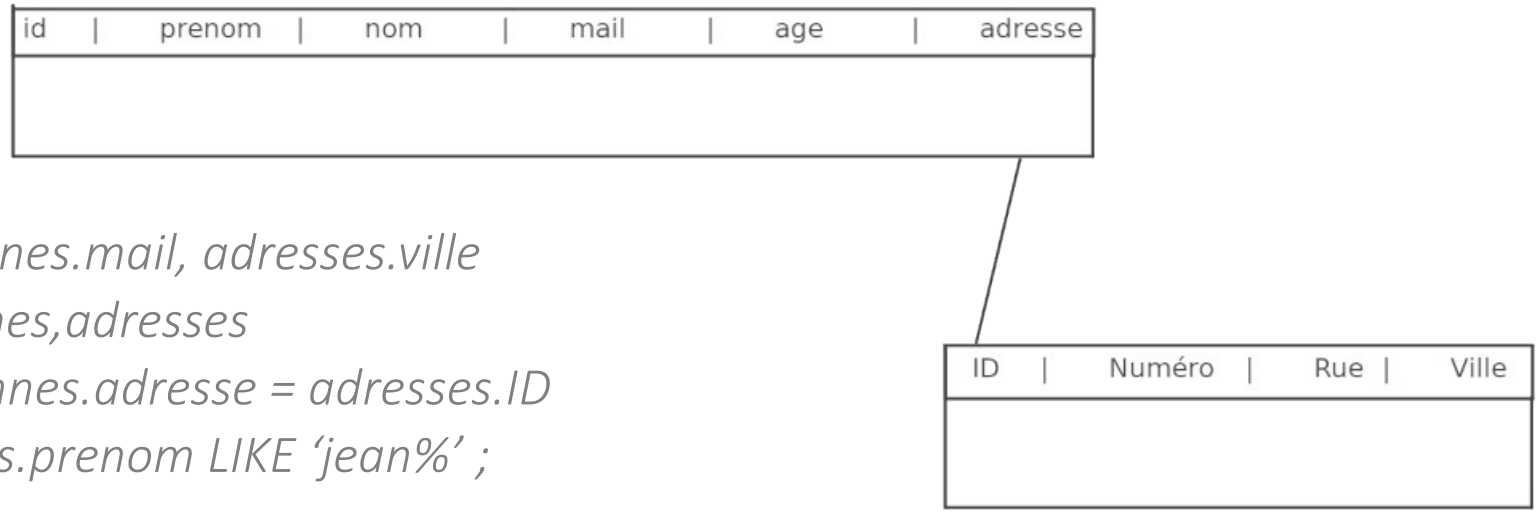
Il est ainsi possible de sélectionner les critères conditionnels de récupération des données ainsi que le format de sortie de la requête pour obtenir des éléments exploitables en l'état.

SELECT id FROM personnes WHERE age ≥ 30 ;

```
+--------+---------------+
| Field  | Type          |
+--------+---------------+
| id     | int(11)       |
| prenom | varchar(100)  |
| nom    | varchar(100)  |
| mail   | varchar(200)  |
| AGE    | decimal(10,0) |
+--------+---------------+
```

Data Manipulation

Un des principes de notre base de données relationnelle est de créer des liens entre nos différentes tables.




```
SELECT personnes.mail, adresses.ville
FROM personnes,adresses
WHERE personnes.adresse = adresses.ID
AND personnes.prenom LIKE 'jean%' ;
```

Et ainsi pouvoir être en mesure de construire une requête corrélant les deux tables.

Data Manipulation

Modifier les données dans une table :

UPDATE table_name SET column_name=expr1 [, column2_name=expr2 ...]
[WHERE where_expression] [LIMIT row_count] ;

Le SET permet d'attribuer une nouvelle valeur à un champ, il est possible de mettre à jour plusieurs champs en même temps.

Le WHERE permet de préciser quels tuples doivent être mis à jour, le système de conditionnement est le même que pour l'instruction SELECT.

Sans condition WHERE, l'intégralité de la table sera mise à jour. Il est possible de définir une LIMIT fixant le nombre de lignes pouvant être modifiées par la requête.

Data Manipulation

Liaison de requête :

Il est possible d'insérer dans une table, des données issues d'une autre requête.

INSERT [INTO] table_name [(col_name1, ...)]
SELECT ...

Et également d'opérer une liaison similaire sur un UPDATE (C'est toujours la donnée dans la table collé au mot clé UPDATE qui est mise à jour).

UPDATE table_name [, table_name2 ...]
SET column_name=expr1 [, column_name2=expr2 ...]
[WHERE where_expression]

Data Manipulation

Suppression des données d'une table :

DELETE FROM table_name
[WHERE where_expression] [LIMIT row_count]

Comme pour les autres types de requêtes, le WHERE sert à conditionner et à préciser les lignes qui doivent être supprimées.

Si aucun WHERE n'est défini, toutes les données de la table seront effacées. Pour procéder à une telle suppression, on préférera utiliser la commande TRUNCATE.

TRUNCATE TABLE ;

TP – SQL STATISTICS

FIN !

Merci pour votre participation !