المملكة المغربية
ⵜⴰⴳⵍⴷⵉⵜ ⵏ ⵍⵎⵖⵔⵉⴱ
وزارة التربية الوطنية
والتعليم الأولي والرياضة
ⵜⴰⵎⴰⵡⴰⵙⵜ ⵏ ⵜⵣⵔⴼⵜ ⴰⵏⴰⵎⵓ
ⴰ ⵜⵓⵓⵔⵔⴰ ⴰⵎⵙⵡⴰⵔⴰ ⴰ ⵜⵜⵍⵜ

Base de Données (1)

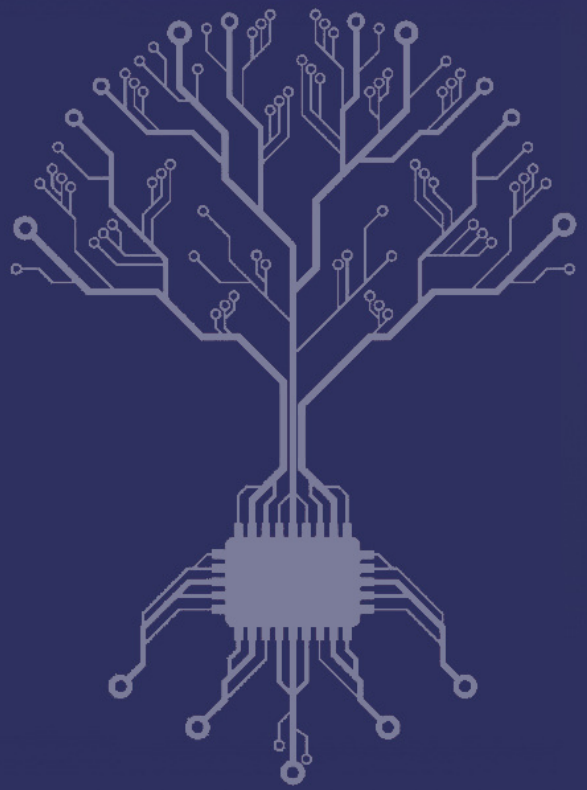
Youssef Daghouj

30 Octobre 2025

1 Introduction

2 SQL
- Langage de Définition de Données (LDD)
- Langage de Manipulation de Données (LMD)

Introduction



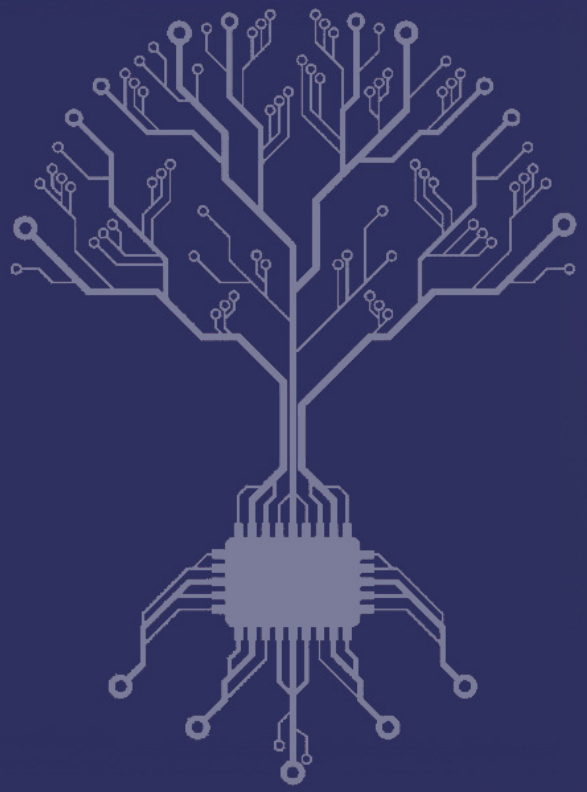
Après avoir établi le MCD et le MLD, il faut adapter ce dernier à un SGBD pour créer le MPD en utilisant un langage appelé SQL (Structured Query Language).

Système de Gestion de Base de Données

Un SGBD est un logiciel (*e.g.* MySQL, Oracle, SQLite, ...) qui permet de gérer une BD pour définir et manipuler les données, tout en garantissant leur cohérence et intégrité.

Le SGBD utilisé dans la suite est SQLite.

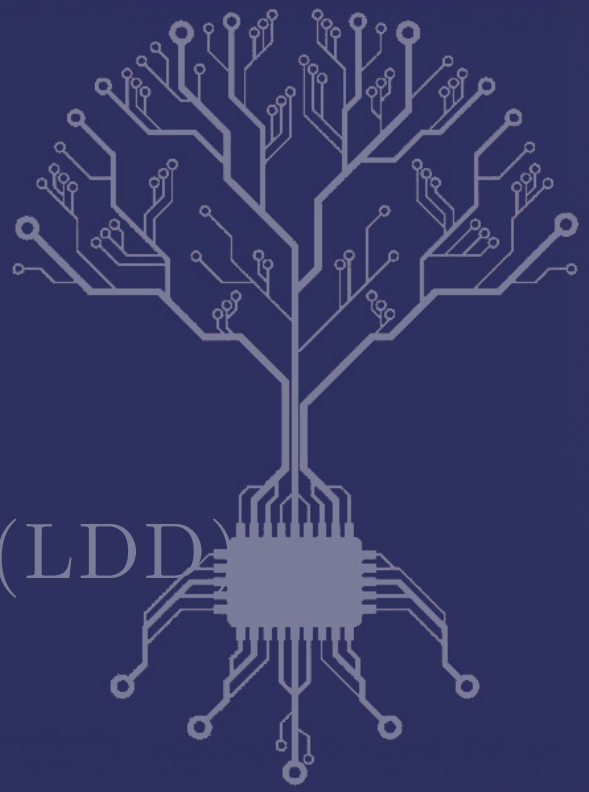
SQL



SQL est un langage informatique normalisé servant à exploiter des bases de données relationnelles. Il comporte plusieurs sous langages comme :

- Langage de Définition de Données (LDD)
- Langage de Manipulation de Données (LMD)

SQL :
Langage de Définition de Données (LDD)



LDD

Le LDD permet de créer, modifier, ou supprimer une table.

Création

La création d'une table consiste à définir :

- Son nom ;
- Ses colonnes (leurs noms, types et contraintes d'intégrité) ;
- Ses contraintes d'intégrité.

Création

Syntaxe

```
CREATE TABLE [IF NOT EXISTS] nomTable (

nomColonne1 type contrainteColonne1

nomColonne2 type contrainteColonne2

...

contrainteTable1,

contrainteTable2,

...

);
```

Création : Types de données

- TEXT : Chaîne de caractères
- INTEGER : Nombre entier
- REAL : Nombre réel

Création : Contraintes d'intégrité

Définition

Une contrainte d'intégrité est une règle imposée à la base de données afin de garantir son intégrité.

Création : Contraintes de colonne

- UNIQUE : Toutes les valeurs de la colonne doivent être différentes ou NULL (non spécifiée).
- NOT NULL ou NULL : Interdit (NOT NULL) ou autorise (NULL) l'insertion de valeur NULL pour cet attribut.
- PRIMARY KEY : Désigne l'attribut comme clé primaire de la table
- DEFAULT valeur : Permet de spécifier la valeur par défaut de la colonne

Création : Contraintes de colonne

Exemple

Soit la table school avec l'attribut nom de type chaîne de caractère qui représente la clé primaire de la table et l'attribut adresse de type chaîne de caractères. Créer la table en indiquant qu'on ne peut pas trouver deux écoles à la même adresse :

```
CREATE TABLE school (
name TEXT PRIMARY KEY,
address TEXT UNIQUE
);
```

Création : Contraintes de colonne

- CHECK (condition) : Vérifie lors de l'insertion des enregistrements (lignes) que l'attribut vérifie la condition.

  Les opérateurs de comparaison :

  - $=, !=, <, >, <=, >=$
  - BETWEEN value1 AND value2
  - IN ou NOT IN (value1, value2, ...)
  - LIKE : permet d'utiliser des jokers (% pour une chaîne de caractères de longueur quelconque et _ pour un seul caractère)

Exemple

```
CREATE TABLE student (
idstudent INTEGER PRIMARY KEY,
firstname TEXT NOT NULL ,
lastname TEXT NOT NULL ,
field TEXT NOT NULL ,
average REAL NOT NULL CHECK (average BETWEEN 0 AND 20),
);
```

Création : Contraintes de table

Les contraintes de tables portent sur plusieurs attributs de la table sur laquelle elles sont définies :

- PRIMARY KEY (colonne1, colonne2, ...) : Désigne la concaténation des attributs cités comme clé primaire de la table.
- FOREIGN KEY (colonne) REFERENCES table (colonne) : Identifie une ou plusieurs colonnes comme étant clé étrangère.

Création : Contraintes de table

Exemple

Soit les schémas :

- professor(idprof, firstname, lastname, mail)
- course(idcourse, name, field)
- session(#idprof, #idcourse, date)

Création : Contraintes de table

Exemple

```
CREATE TABLE professor(
idprof INTEGER PRIMARY KEY,
firstname TEXT NOT NULL ,
lastname TEXT NOT NULL ,
mail TEXT NOT NULL CHECK (mail LIKE '%@%.%')
);
```

Création : Contraintes de table

Exemple

```
CREATE TABLE course (
idcourse INTEGER PRIMARY KEY,
name TEXT NOT NULL ,
field TEXT NOT NULL
);
```

Création : Contraintes de table

Exemple

```
CREATE TABLE session(
idprof INTEGER,
idcourse INTEGER,
date TEXT,
PRIMARY KEY (idprof, idcourse),
FOREIGN KEY (idprof) REFERENCES professor (idprof),
FOREIGN KEY (idcourse) REFERENCES course (idcourse)
);
```

Modification

- Ajout d'une colonne
  - Syntaxe : ALTER TABLE nomTable ADD (nomColonne type contraintes) ;
  - Exemple : ALTER TABLE student ADD (age INTEGER) ;
- Renommage d'une colonne
  - Syntaxe : ALTER TABLE nomTable RENAME ancienNom TO nouveauNom
- Suppression d'une colonne
  - Syntaxe : ALTER TABLE nomTable DROP nomColonne
  - Exemple : ALTER TABLE student DROP age

Suppression

DROP TABLE nomTable

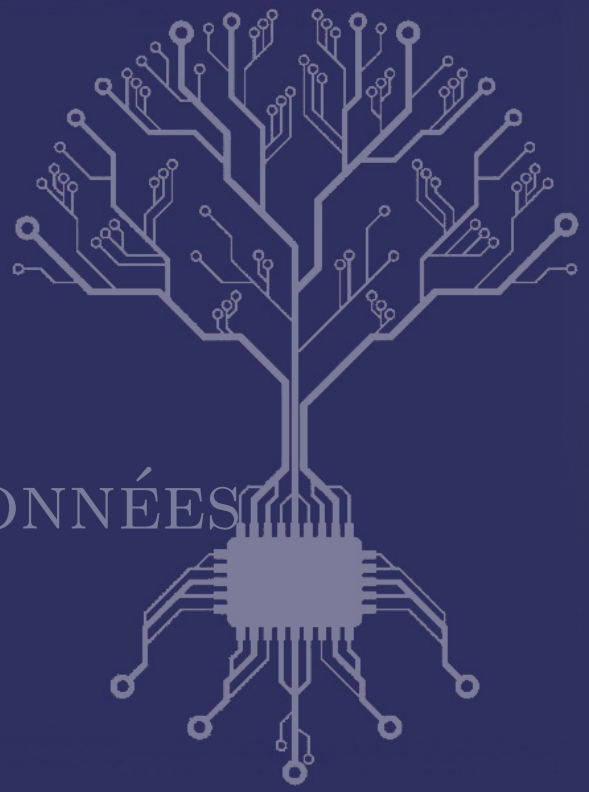
Modification et Suppression

La modification ou suppression d'une ligne dans une table pourra être impossible s'il existe des lignes dans d'autres tables référençant la clé primaire de cette ligne. Pour éviter ce genre de problème, on peut ajouter à la contrainte FOREIGN KEY des options :

Modification et Suppression

- En cas de modification :
  - ON UPDATE CASCADE : modifier les valeurs des clés étrangères
  - ON UPDATE SET NULL : mettre à NULL des valeurs des clés étrangères
- En cas de suppression :
  - ON DELETE CASCADE : supprimer les lignes concernées
  - ON DELETE SET NULL : mettre à NULL des valeurs des clés étrangères

SQL : Langage de Manipulation de Données (LMD)



C'est l'ensemble des commandes qui permettent l'ajout, la modification et la suppression de lignes.

Insertion de lignes

La commande INSERT permet d'insérer une ligne dans une table en spécifiant les valeurs à insérer.

Syntaxe

```
INSERT INTO nomTable VALUES
(valeur11, valeur12, . . . ),
(valeur21, valeur22, . . . ),
... ;
```

Insertion de lignes

Exemple

```
INSERT INTO student VALUES
(251423561, "ahmad", "bachir", "economics", 15, CP1170),
(251677830, "fatima", "mohammadi", "mathematics", 19, CP1133);
```

Modification de lignes

La commande UPDATE permet de modifier les valeurs d'une ou plusieurs colonnes, dans une ou plusieurs lignes, selon une condition.

Syntaxe

```
UPDATE nomTable SET nomColonne = expression, ... WHERE condition ;
```

En l'absence d'une clause WHERE, toutes les lignes sont mises à jour.

Modification de lignes

Exemple

Soit la table employee dont les colonnes sont : idemployee, name, salary, departement. Augmenter de 10% le salaire des employés du département commercial :

```
UPDATE employee
SET salary = salary * 1.1
WHERE departement = "commerce" ;
```

Suppression de lignes

Syntaxe

DELETE FROM nomTable WHERE condition ;

Exemple

DELETE FROM employee WHERE departement = ”commerce” ;