CH04 - Bases de données

Partie 3

T^ale NSI

20 septembre 2021

Niveau physique
Langage SQL

Le SGBD

Il garantit entre autres

Le SGBD

Il garantit entre autres

- l'indépendance physique de la BDD : l'utilisateur n'a pas à se soucier des aspects matériels ;

Le SGBD

Il garantit entre autres

- l'indépendance physique de la BDD : l'utilisateur n'a pas à se soucier des aspects matériels ;
- l'indépendance logique : les programmes qui utilisent la BDD sont indépendants de sa structure logique ;

Le SGBD

Il garantit entre autres

- l'indépendance physique de la BDD : l'utilisateur n'a pas à se soucier des aspects matériels ;
- l'indépendance logique : les programmes qui utilisent la BDD sont indépendants de sa structure logique ;
- l'accès aux données : il se fait grâce à un langage de manipulation des données (LMD) optimisé pour la rapidité et l'accès simultané multiple en lecture/écriture ;

Le SGBD

Il garantit entre autres

- l'indépendance physique de la BDD : l'utilisateur n'a pas à se soucier des aspects matériels ;
- l'indépendance logique : les programmes qui utilisent la BDD sont indépendants de sa structure logique ;
- l'accès aux données : il se fait grâce à un langage de manipulation des données (LMD) optimisé pour la rapidité et l'accès simultané multiple en lecture/écriture ;
- la centralisation des données pour administration ;

Le SGBD

Il garantit entre autres

- l'indépendance physique de la BDD : l'utilisateur n'a pas à se soucier des aspects matériels ;
- l'indépendance logique : les programmes qui utilisent la BDD sont indépendants de sa structure logique ;
- l'accès aux données : il se fait grâce à un langage de manipulation des données (LMD) optimisé pour la rapidité et l'accès simultané multiple en lecture/écriture ;
- la centralisation des données pour administration ;
- la non-redondance des données ;

Le SGBD

Il garantit entre autres

- l'indépendance physique de la BDD : l'utilisateur n'a pas à se soucier des aspects matériels ;
- l'indépendance logique : les programmes qui utilisent la BDD sont indépendants de sa structure logique ;
- l'accès aux données : il se fait grâce à un langage de manipulation des données (LMD) optimisé pour la rapidité et l'accès simultané multiple en lecture/écriture ;
- la centralisation des données pour administration ;
- la non-redondance des données ;
- la sécurité des données vis-à-vis du piratage mais aussi des pannes physiques.

Principaux SGBD en 2020

Le SQL

- *Structured Query Langage* (langage de requêtes structuré).

Le SQL

- *Structured Query Langage* (langage de requêtes structuré).
- Créé en 1974, normalisé en 1986, dernière version parue en 2011.

Le SQL

- *Structured Query Langage* (langage de requêtes structuré).
- Créé en 1974, normalisé en 1986, dernière version parue en 2011.
- Utilisé par la plupart des SGBD avec de petites différences.

Exemple de requête

```sql
SELECT DISTINCT nom, prenom
FROM Auteur
         JOIN Ecrire ON Ecrire.id_auteur = Auteur.id_auteur
         JOIN Livre ON Livre.num_isbn = Ecrire.num_isbn
WHERE Livre.titre LIKE '%s%';
```

Exemple de requête

```sql
SELECT DISTINCT nom, prenom
FROM Auteur
         JOIN Ecrire ON Ecrire.id_auteur = Auteur.id_auteur
         JOIN Livre ON Livre.num_isbn = Ecrire.num_isbn
WHERE Livre.titre LIKE '%s%';
```

Voici comment obtenir la liste des noms et prénoms des auteurs ayant écrit un livre dont le titre comporte la lettre « s ». Nous expliquerons comment produire de telles requêtes plus tard.

Vocabulaire

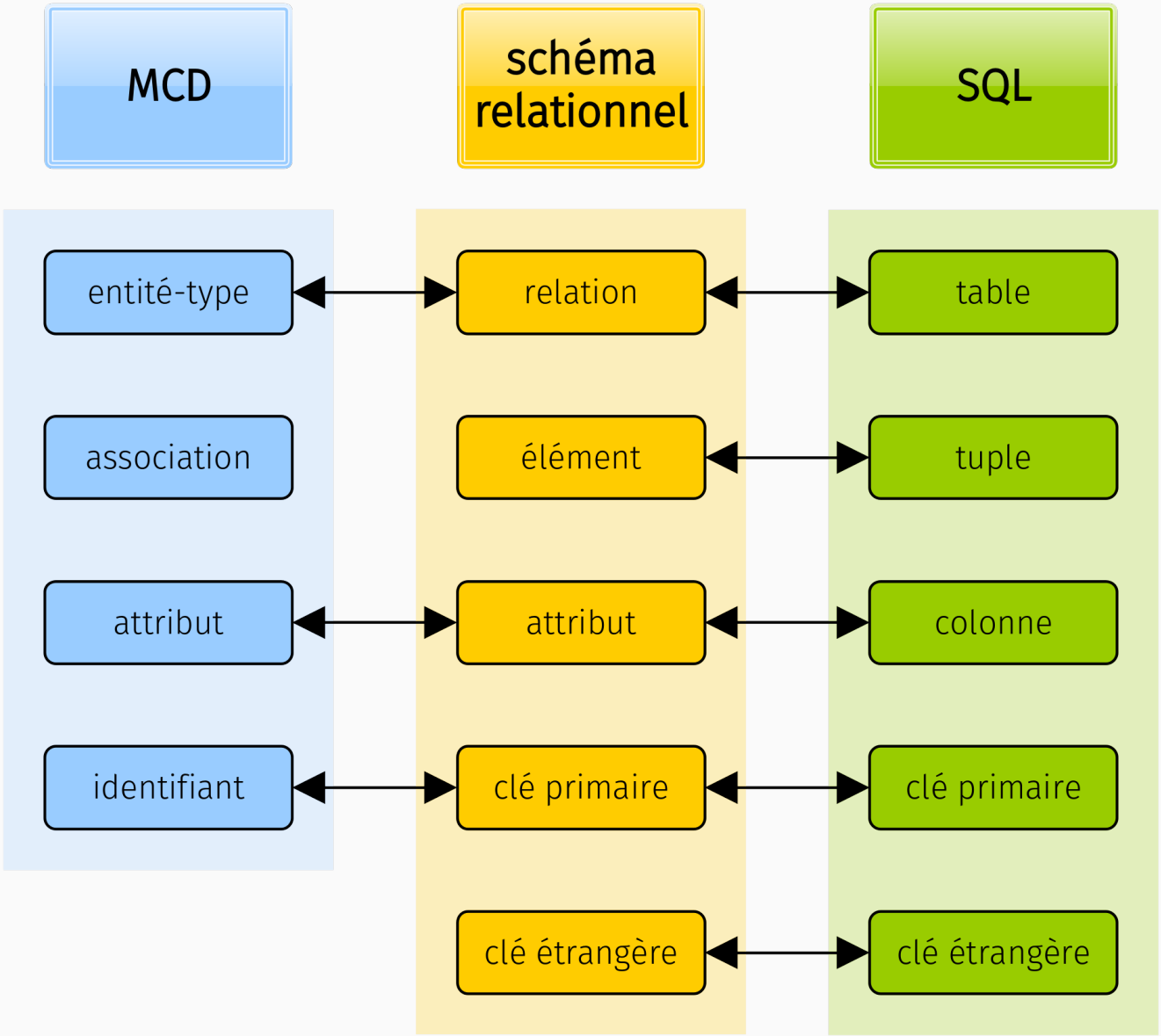
En SQL, les relations s'appellent des tables.

Vocabulaire

En SQL, les relations s'appellent des tables.

Les éléments des tables s'appellent des tuples.

Bilan des termes utilisés



Conventions

On écrira les mots-clés SQL en majuscules.

Conventions

On écrira les mots-clés SQL en majuscules.

On ne met pas d'accents ou d'espaces dans les noms des tables ou des attributs.

Conventions

On écrira les mots-clés SQL en majuscules.

On ne met pas d'accents ou d'espaces dans les noms des tables ou des attributs.

Les espaces et tabulations n'ont qu'un rôle esthétique.

Conventions

On écrira les mots-clés SQL en majuscules.

On ne met pas d'accents ou d'espaces dans les noms des tables ou des attributs.

Les espaces et tabulations n'ont qu'un rôle esthétique.

Les requêtes peuvent prendre plusieurs lignes mais doivent se terminer par un point-virgule.

Conventions

On écrira les mots-clés SQL en majuscules.

On ne met pas d'accents ou d'espaces dans les noms des tables ou des attributs.

Les espaces et tabulations n'ont qu'un rôle esthétique.

Les requêtes peuvent prendre plusieurs lignes mais doivent se terminer par un point-virgule.

On utilisera SQLite car on peut s'en servir avec DB Browser sans installation compliquée.

Création de la BDD

Créer une BDD

```
CREATE DATABASE Bibliotheque;
USE Bibliotheque;
```

Créer une BDD

```
CREATE DATABASE Bibliotheque;
USE Bibliotheque;
```

On n'utilisera pas cette commande : dans DB Browser on peut créer un nouveau fichier de BDD sans passer par SQLite.

Supprimer une BDD

```sql
DROP DATABASE Bibliotheque;
```

Supprimer une BDD

```sql
DROP DATABASE Bibliotheque;
```

On n'utilisera pas cette commande non plus.

Créer une table

Voici comment créer la table **Pays** :

```sql
DROP TABLE IF EXISTS Pays; -- recréer la table de zéro
CREATE TABLE Pays
(
    nom_pays   TEXT,
    population INTEGER,
    superficie INTEGER,
    PRIMARY KEY (nom_pays), -- clé primaire
    CHECK (population > 0), -- contraintes utilisateur
    CHECK (superficie > 0)
);
```

Table Livre

```sql
DROP TABLE IF EXISTS Livre;
CREATE TABLE Livre
(
    num_isbn INTEGER,
    titre    TEXT,
    annee    TEXT,
    PRIMARY KEY (num_isbn),
    CHECK (date(annee) BETWEEN '1900' AND '2100')
);
```

Table Livre

```sql
DROP TABLE IF EXISTS Livre;
CREATE TABLE Livre
(
    num_isbn INTEGER,
    titre    TEXT,
    annee    TEXT,
    PRIMARY KEY (num_isbn),
    CHECK (date(annee) BETWEEN '1900' AND '2100')
);
```

`date(annee) BETWEEN '1900' AND '2100'` est l'équivalent SQL de `'1900' <= date(annee) <= '2100'` en Python.

Table Livre

```sql
DROP TABLE IF EXISTS Livre;
CREATE TABLE Livre
(
    num_isbn INTEGER,
    titre    TEXT,
    annee    TEXT,
    PRIMARY KEY (num_isbn),
    CHECK (date(annee) BETWEEN '1900' AND '2100')
);
```

`date(annee) BETWEEN '1900' AND '2100'` est l'équivalent SQL de `'1900' <= date(annee) <= '2100'` en Python.

Attention

SQLite ne connaît pas le type `DATE`, il faut créer des attributs de type `TEXT` et utiliser la fonction `date`.

Table Livre

```sql
DROP TABLE IF EXISTS Auteur;
CREATE TABLE Auteur
(
    id_auteur      INTEGER,
    nom_pays       TEXT,
    nom            TEXT,
    prenom         TEXT,
    date_naissance TEXT,
    PRIMARY KEY (id_auteur),
    UNIQUE (nom, prenom), -- contrainte d'unicité
    FOREIGN KEY (nom_pays) REFERENCES Pays (nom_pays)
    /*nom_pays est une clé étrangère*/
        ON DELETE CASCADE
        /*si on supprime des tuples dans Pays, automatiquement
        (en cascade) on supprimera les tuples qui y font
        réference dans Auteur*/
        ON UPDATE CASCADE
        /*si on met à jour les attributs nom_pays dans Pays,
        alors le SGBD les mettra à jour aussi dans Auteur*/
```

Ordre des créations des tables

On ne peut pas créer **Auteur** avant d'avoir crée **Pays** car **Auteur** possède une clé étrangère liée à **Pays**.

Table Ecrire

```sql
DROP TABLE IF EXISTS Ecrire;
CREATE TABLE Ecrire
(
    id_auteur    INTEGER,
    num_isbn     INTEGER,
    nb_chapitres INTEGER,
    PRIMARY KEY (id_auteur, num_isbn),
    FOREIGN KEY (id_auteur) REFERENCES Auteur (id_auteur)
        ON DELETE CASCADE
        ON UPDATE CASCADE,
    FOREIGN KEY (num_isbn) REFERENCES Livre (num_isbn)
            ON DELETE CASCADE
            ON UPDATE CASCADE
);
```

Variantes syntaxiques et autres

Écriture plus compacte

On peut signifier qu'un attribut est une clé étrangère dans sa définition même :

Écriture plus compacte

On peut signifier qu'un attribut est une clé étrangère dans sa définition même :

```
DROP TABLE IF EXISTS Ecrire;
CREATE TABLE Ecrire
(
    id_auteur    INTEGER REFERENCES Auteur (id_auteur)
        ON DELETE CASCADE
        ON UPDATE CASCADE,
    num_isbn     INTEGER REFERENCES Livre (num_isbn)
        ON DELETE CASCADE
        ON UPDATE CASCADE,
    nb_chapitres INTEGER,
    PRIMARY KEY (id_auteur, num_isbn)
);
```

Écriture plus compacte (bis)

On peut signifier qu'un attribut est une clé primaire dans sa définition même :

Écriture plus compacte (bis)

On peut signifier qu'un attribut est une clé primaire dans sa définition même :

```sql
DROP TABLE IF EXISTS Livre;
CREATE TABLE Livre
(
    num_isbn INTEGER PRIMARY KEY,
    titre    TEXT,
    annee    TEXT,
    CHECK (date(annee) BETWEEN '1900' AND '2100')
);
```

Insertion des données dans la BDD

Données de Pays

```sql
INSERT INTO Pays
VALUES ('France', 672051, 67064000),
       ('Italie', 301336, 66436000),
       ('Royaume-Uni', 242900, 60317000);
```

Les attributs des tuples sont dans le même ordre que lors de la création.

Et cætera

De même que lors de la création, on ne peut pas insérer de tuples dans **Auteur** avant d'avoir peuplé **Pays** : en effet dans un tuple de **Auteur** tel que

```
(1, 'France', 'Hugo', 'Victor', '1802-02-26')
```

Les contraintes de référence font qu'un tuple « France » doit d'abord exister dans **Pays**.