INFORMATIQUE Développement d'applications
BLOC 1
UE05 Bases de données

Chapitre 3.1 : SQL - DDL

Vincent Reip

Octobre 2025

Objectif

- Au terme de ce chapitre, l'étudiant sera capable de :
  - Créer et modifier les tables d'un schéma relationnel

SQL

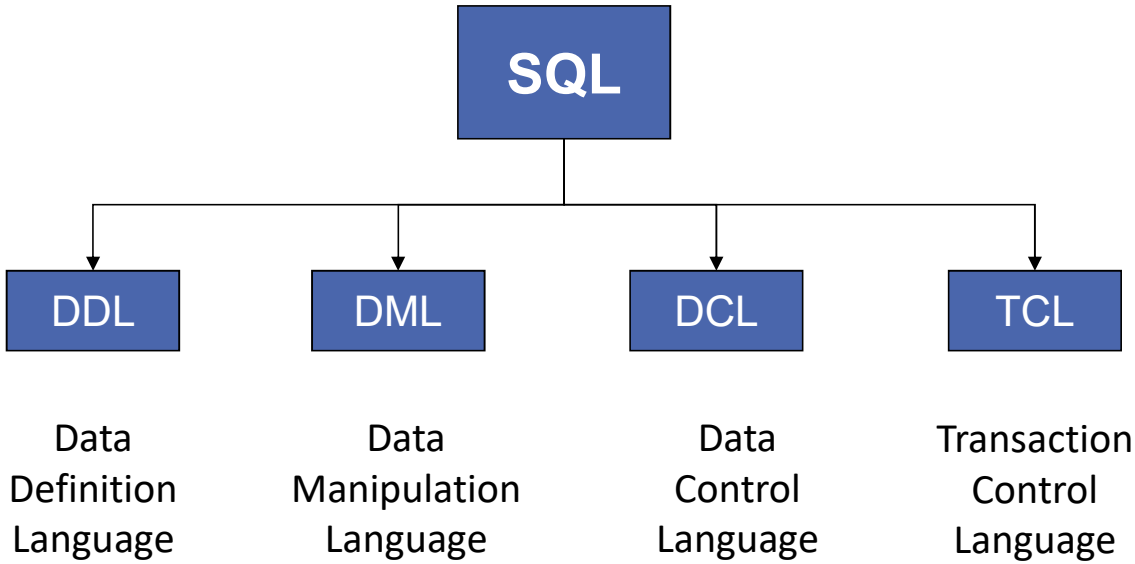
- Structured Query Language (SQL)
  - Langage apparu dans les années 80
  - Langage le plus répandu pour la gestion des bases de données relationnelles
  - SQL se divise en
    - Un langage de définition de données (DDL)
    - Un langage de manipulation de données (DML)
    - Un langage de contrôle d'accès (DCL)
    - Un langage de contrôle des transactions (TCL)

Evolution de SQL

- SEQUEL (Début 70)
  - Manipulation et recherche de données (IBM Research Labs)
- SQL 1 (1986) – 1ère standardisation
  - Création et suppression de tables
  - Sélection, mise à jour,… de données
  - Gestion de transactions
- SQL 2 (1992)
  - Création de type de données
  - Meilleur support des contraintes d'intégrité
  - Modification des schémas (ALTER TABLE)
- SQL 3 (1999)
  - Extension au relationnel-objet
- SQL:2003 (2003)
  - Support de XML
- SQL:2008 (2008)
  - Adaptations « mineures », curseurs, auto-incrémentation
- SQL:2011 (2011)
  - Tables « temporelles » (historisation)
- SQL:2016 (2016)
  - Support de JSON

Malgré les normes, il existe beaucoup de différences de syntaxe et de fonctionnalités entre les différents SGBD

SQL


SQL
DDL
DML
DCL
TCL
Data Definition Language
Data Manipulation Language
Data Control Language
Transaction Control Language

DDL

•DDL (Data Definition Language) : Permet de gérer le schéma d'une base de données

- Ajouter de nouvelles tables
  - CREATE TABLE
- Supprimer des tables existantes (et leurs données)
  - DROP TABLE
- Modifier le schéma de tables existantes
  - ALTER TABLE
- Créer des vues
  - CREATE VIEW
- Créer des index
  - CREATE INDEX

DML

- DML (Data Manipulation Language) : Permet de gérer les fonctions primitives CRUD sur les données d'une base de données
  - Créer (CREATE) de nouveaux tuples dans une table
    - INSERT INTO
  - Lire (READ) des données dans les tables
    - SELECT … FROM …
  - Modifier (UPDATE) des données dans les tables
    - UPDATE
  - Supprimer (DELETE) des tuples dans une table
    - DELETE FROM

DCL

- DCL (Data Control Language) : Permet de gérer le contrôle de l'accès aux données par les utilisateurs
  - Permettre à un utilisateur d'effectuer certaines opérations sur certaines données
    - GRANT
  - Supprimer les permissions données à un utilisateur
    - REVOKE

```
GRANT INSERT, UPDATE, DELETE
ON TABLE VOITURE
TO IN140121
```

```
GRANT UPDATE (kilometrage, couleur)
ON TABLE VOITURE
TO IN140121
```

```
GRANT SELECT
ON TABLE VOITURE
TO PUBLIC
```

```
GRANT SELECT (modele, kilometrage, couleur)
ON TABLE VOITURE
TO IN150548
```

```
GRANT ALL PRIVILEGES
ON TABLE VOITURE
TO PUBLIC
```

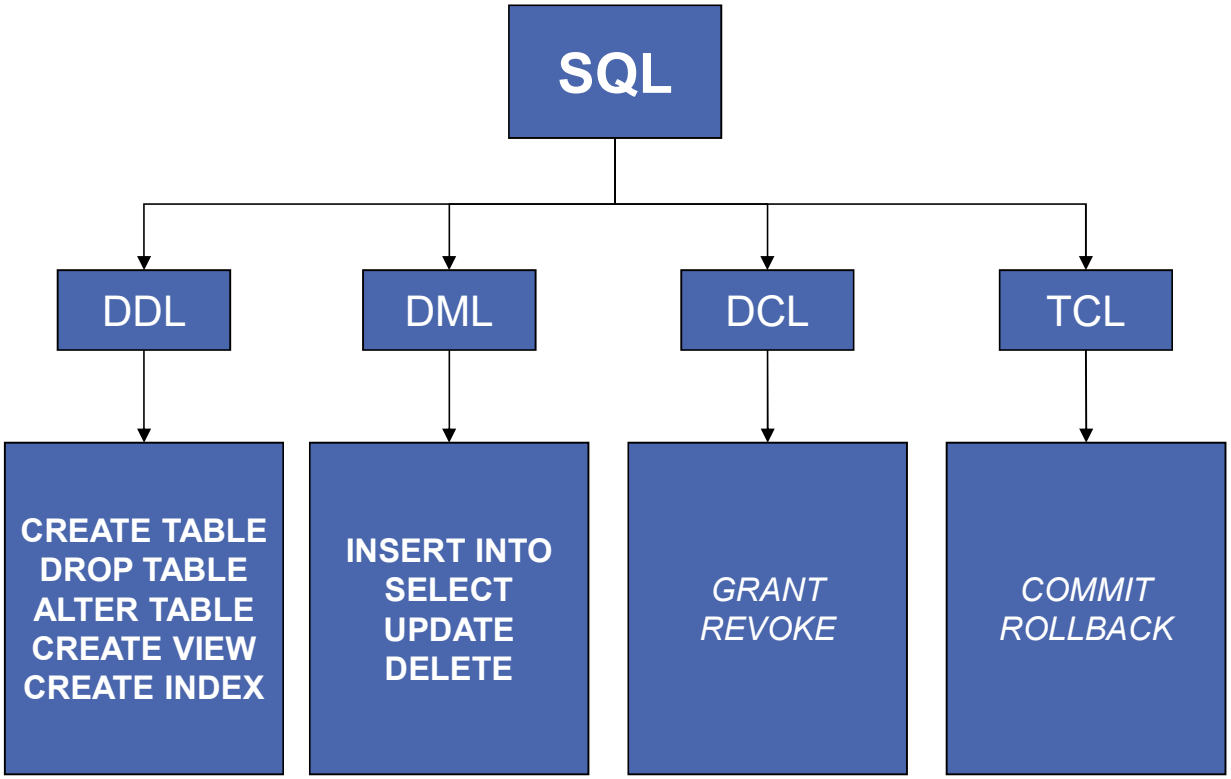
```
REVOKE UPDATE
ON TABLE VOITURE
TO IN140121
```

TCL

- TCL (Transaction Control Language) : Permet de gérer l'exécution de transactions[1] au sein de la base de données
  - Confirmer les modifications provoquées par une transaction
    - COMMIT
  - Annuler les modifications provoquées par une transaction
    - ROLLBACK

(1) une transaction est un événement au cours duquel une base de données passe d'un état A à un état B à la suite d'une multitude d'opérations (source : wikipedia).

SQL


SQL
DDL
DML
DCL
TCL
CREATE TABLE
DROP TABLE
ALTER TABLE
CREATE VIEW
CREATE INDEX
INSERT INTO
SELECT
UPDATE
DELETE
GRANT
REVOKE
COMMIT
ROLLBACK

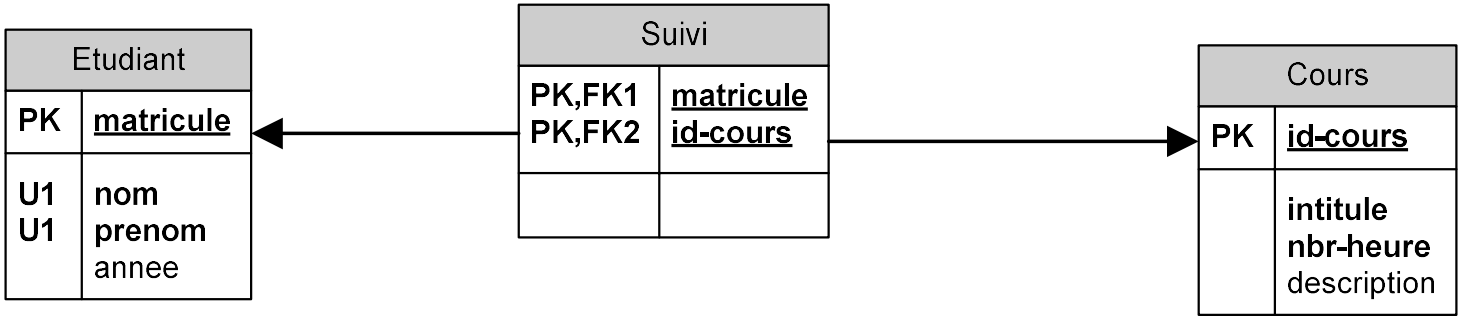
DDL : CREATE TABLE

```
CREATE TABLE nom_table (
attribut1 type_attr1 [valeur_par_defaut] [contrainte_attribut],
attribut2 type_attr2 [valeur_par_defaut] [contrainte_attribut],
….
[contrainte_de_table]
)
```

- Une nouvelle relation (table) est créée en spécifiant :
  - le nom de la table
  - les attributs de la table
    - le nom
    - le type
    - une éventuelle valeur par défaut
    - optionnellement une ou plusieurs contraintes sur l'attribut
  - les éventuelles contraintes sur la table

Parmi les contraintes possibles, on retrouve : PRIMARY KEY, UNIQUE, FOREIGN KEY, NOT NULL, CHECK…

CREATE TABLE : Exemple



CREATE TABLE : Exemple

```
CREATE TABLE Etudiant (
 matricule INTEGER PRIMARY KEY,
 nom VARCHAR(60) NOT NULL,
 prenom VARCHAR(60) NOT NULL,
 annee INTEGER DEFAULT 1,
 UNIQUE(nom, prenom )
);
```

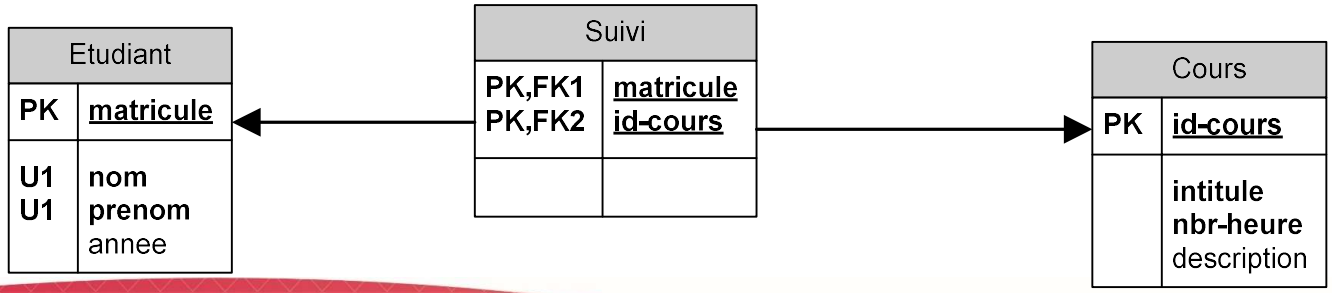
Etudiant	
PK	**matricule**
U1 **U1**	**nom** **prenom** annee

```
CREATE TABLE Cours (
 id-cours INTEGER PRIMARY KEY,
 intitule VARCHAR(50) NOT NULL,
 nbr-heure NUMERIC(3,0) NOT NULL,
 description TEXT
);
```

Cours	
PK	**id-cours**
	intitule **nbr-heure** description

CREATE TABLE : Exemple

```
CREATE TABLE Suivi (
 matricule INTEGER,
 id-cours INTEGER,
 PRIMARY KEY( matricule, id-cours),
 FOREIGN KEY (matricule) REFERENCES Etudiant(matricule),
 FOREIGN KEY (id-cours) REFERENCES Cours(id-cours)
);
```



CREATE TABLE : Remarques

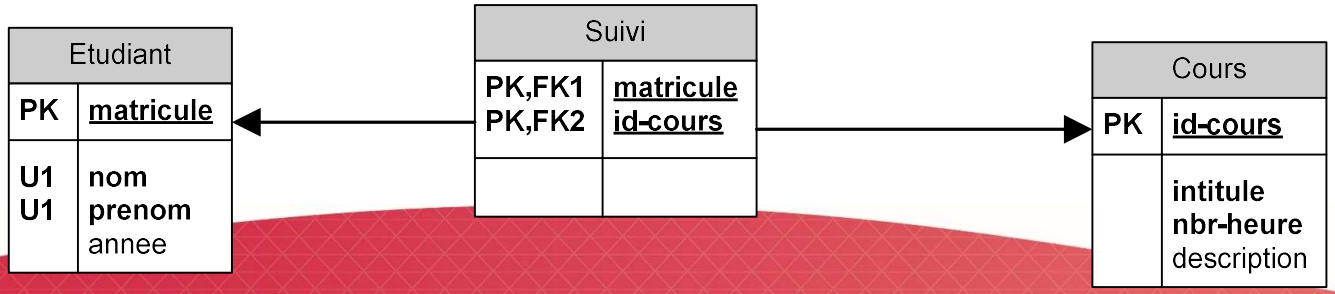
- Un attribut ne peut être déclaré comme participant à la fois à une clé primaire (PRIMARY KEY) et à une clé candidate (UNIQUE)
- Une clé composée de plusieurs attributs doit être déclarée comme contrainte de table
- Des attributs spécifiés comme étant uniques peuvent admettre des valeurs NULL
- Par défaut, une clé étrangère correspond à la clé primaire de la table référencée. Elle peut également correspondre à une clé candidate
- L'ordre de création des tables est important : on ne peut référencer une table avant qu'elle ne soit créée…

Gestion des clés étrangères

- La déclaration d'une contrainte d'intégrité référentielle (clé étrangère) induit que :
  - un tuple référencé ne peut être supprimé
  - la clé primaire d'un tuple référencé ne peut être modifiée
- Ce comportement peut être modifié en gérant les événements de suppression (ON DELETE) et de modification (ON UPDATE). On peut leur associer 3 comportements :
  - SET NULL : clé étrangère mise à NULL
  - SET DEFAULT : clé étrangère prend une valeur par défaut (NULL si cette valeur n'existe pas dans la table référencée)
  - CASCADE : suppression/modification des tuples dont la valeur de la clé étrangère correspond à la valeur de la clé primaire (candidate) du tuple supprimé/modifié.

Gestion des clés étrangères

```
CREATE TABLE Suivi (
 matricule INTEGER,
 id-cours INTEGER,
 PRIMARY KEY( matricule, id-cours),
 FOREIGN KEY (matricule) REFERENCES Etudiant(matricule)
                                          ON DELETE CASCADE,
 FOREIGN KEY (id-cours) REFERENCES Cours(id-cours)
);
```



Lorsqu'un tuple X est supprimé dans la table ETUDIANT, les tuples de la table SUIVI référençant ce tuple X sont automatiquement supprimés.

Contraintes

- Les contraintes permettent de préserver la cohérence des données.
  - PRIMARY KEY(*) et UNIQUE : contrainte d'unicité
  - NOT NULL : valeur obligatoire
  - FOREIGN KEY : clé étrangère (intégrité référentielle)
  - CHECK : permet de vérifier les propriétés d'un ou plusieurs attributs de la table grâce à une expression booléenne

```
CONSTRAINT nom_contrainte CHECK ( expression_booleenne )
```

(*) la contrainte PRIMARY KEY implique le caractère obligatoire (NOT NULL) du/des attribut(s)

Contraintes : exemples

- CREATE TABLE professeur (
  matricule INTEGER **PRIMARY KEY**,
  nom VARCHAR(60) **NOT NULL**,
  prenom VARCHAR(60) **NOT NULL**,
  salaire NUMERIC (10,2) **NOT NULL**,
  CONSTRAINT GrosSalaire CHECK (salaire > 2500));

- CREATE TABLE voiture (
  numSerie INTEGER **PRIMARY KEY**,
  marque VARCHAR(60) **NOT NULL**,
  modele VARCHAR(60) **NOT NULL UNIQUE**,
  rejetCo2 INTEGER **NOT NULL**,
  conso NUMERIC (4,2) **NOT NULL**,
  CONSTRAINT Eco CHECK (rejetCo2 < 120 AND conso < 5.2));

Les contraintes de type « CHECK » peuvent uniquement faire intervenir des attributs de la table concernée.

DROP TABLE

```
DROP TABLE nom_table [CASCADE CONSTRAINT]
```

- Permet de supprimer une table ainsi que tous les tuples qu'elle contient
- Si la clause « CASCADE CONSTRAINT » est utilisée, toutes les contraintes d'intégrité référentielle (clé étrangère) référençant cette table seront-elles aussi supprimées
  - Sinon, le SGBD refuse de supprimer une table référencée

ALTER TABLE

- La commande ALTER TABLE permet de modifier le schéma d'une table :
  - Renommer la table
    - ALTER TABLE nom_table RENAME TO nouveau_nom_table
  - Ajouter un / des attributs
    - ALTER TABLE nom_table ADD attribut type_attr [valeur_par_defaut] [contrainte_attribut]
  - Modifier un attribut existant
    - ALTER TABLE nom_table MODIFY attribut type_attr [valeur_par_defaut] [contrainte_attribut]
  - Supprimer un attribut existant
    - ALTER TABLE nom_table DROP COLUMN attribut [CASCADE CONSTRAINT]
  - Renommer un attribut existant
    - ALTER TABLE nom_table RENAME COLUMN attribut TO nouveau_nom_attribut