

# Atelier Pratique SGBD MySQL

**Sous l'encadrement de :** Pr. BENNANI Anas

**Réalisé par:** SACHA Ilyas

**Date :** 05/15/2024

**Apogee:** 19009409

# TABLE DE MATIERE

TABLE DE MATIERE .....	1
INTRODUCTION GENERALE .....	3
ATELIER 1 : DEVELOPPER UNE BASE DE DONNEES AVEC MySQL .....	4
1. Création de la base de données et des tables principales .....	4
1.1. Création de la base de données .....	4
1.2. Définition des tables.....	4
2. Insertion et gestion des donnée .....	5
3. Création et manipulation de la table des commandes .....	6
3.1. Création de la table des commandes .....	6
3.2. Manipulation de la table des commandes .....	7
4. Requêtes avancées .....	7
ATELIER 2 : DEVELOPPER LA SHEMA D'UNE BASE DE DONNEES AVEC MySQL .....	9
1. Configuration de la base de données et des tables .....	9
1.1. Création et configuration de la base Biblio_sacha.....	9
1.2. Création des tables Etudiant, Livre, Auteur, Editeur, et Thème.....	9
2. Modification des structures de tables .....	13
2.1. Ajout de contraintes pour assurer l'intégrité des données .....	13
3. Modification des structures de tables .....	15
3.1. Création de la Table Prêt .....	15
3.2. Ajout de Contraintes Uniques .....	16
3.3. Ajout de Contraintes de Validation .....	17
3.4. Insertion de Données dans les tables crée.....	17
Conclusion : .....	19
ATELIER 3 : INTERROGER LA BASE DE DONNEES .....	20
1. Insertion de Données .....	20
2. Requêtes SQL pour la Gestion et l'Analyse des Données .....	24
2.1. Sélection des étudiants ayant plus de 21 ans .....	24
2.2. Sélection des livres empruntés non rendus .....	25
3. Opérations sur la Table Professeur .....	27
3.1. Création de la Table Professeur.....	27

## TABLE DE MATIERE

3.2.	Ajout d'une Contrainte Unique -- RequêteLDD3 .....	27
3.3.	Ajout et Modification de la Colonne email -- RequêteLDD4/DD5.....	27
3.4.	Suppression de la Colonne email -- RequêteLDD6.....	28
3.5.	Suppression de la Table <b>Professeur</b> -- RequêteLDD7 .....	28
4.	Manipulation des Données dans la Table Etudiant.....	28
4.1.	Insertion de Nouvelles Données .....	28
4.2.	Mise à jour de la ville.....	29
4.3.	Incrémentation de l'Âge .....	29
4.4.	Incrémentation de l'Âge .....	29
4.5.	Requêtes de Sélection .....	29
	Conclusion .....	45
	CONCLUSION GENERALE .....	46

# INTRODUCTION GENERALE

Ce rapport présente un atelier sur la gestion de bases de données relationnelles à travers la création et la manipulation de bases de données en utilisant SQL. Il se divise en trois chapitres principaux, chacun abordant un aspect distinct de la gestion de bases de données. Le premier chapitre se concentre sur la gestion de bases de données d'entreprise avec MySQL, incluant la création de tables pour les employés, les clients et les commandes, ainsi que des requêtes avancées pour interroger les données. Le deuxième chapitre traite de l'administration d'une bibliothèque universitaire, avec la création de tables pour les étudiants, les livres, les auteurs, les éditeurs et les prêts, tout en ajoutant des contraintes pour assurer l'intégrité des données. Enfin, le troisième chapitre approfondit les techniques avancées de gestion et d'analyse des données dans le contexte de la bibliothèque universitaire, en introduisant des requêtes complexes et des opérations de reporting.

# ATELIER 1 : DEVELOPPER UNE BASE DE DONNEES AVEC MySQL

Dans cette partie, nous explorons la création et la gestion d'une base de données pour une entreprise fictive, en utilisant le langage SQL. Ce chapitre détaille le processus de définition de la structure de la base de données, y compris la création des tables nécessaires pour stocker les informations sur les employés, les clients, et les commandes. Nous aborderons également les techniques pour insérer, modifier, et interroger les données, ce qui est essentiel pour la gestion quotidienne des informations d'entreprise.

## 1. Création de la base de données et des tables principales

### 1.1. Création de la base de données

On a commencé par la création d'une nouvelle base de données nommée « TP\_SGBD\_sacha ». C'est l'étape initiale pour séparer et organiser les données pour des applications ou des objectifs spécifiques.

En utilisant cette commande :

```
CREATE DATABASE TP_SGBD_sacha ;
```

Et après on a spécifié que les opérations suivantes seront effectuées dans la base de données **TP\_SGBD\_sacha**.

```
USE TP_SGBD_sacha ;
```

### 1.2. Définition des tables

On a créé une table nommée **Employés** avec quatre colonnes : **num\_employe**, **nom**, **prenom**, et **tel**. Le champ **num\_employe** est configuré pour s'incrémenter automatiquement et **num\_employe** est la clé primaire. Voici la requête :

```
CREATE TABLE Employes (  
    num_employe INT AUTO_INCREMENT PRIMARY KEY,  
    nom VARCHAR(35),  
    prenom VARCHAR(35),  
    tel VARCHAR(13) )
```

Après on va créer une autre table appelé **Clients**. avec des détails tels que la référence du client, le nom de la société, la ville, et le code postal. **Refclient** est la clé primaire.

```
CREATE TABLE clients (  
    Refclient VARCHAR(20) PRIMARY KEY ,  
    nom_ste VARCHAR(50),  
    ville VARCHAR(50),  
    code_postal VARCHAR(10)) ;
```

## 2. Insertion et gestion des donnée

On va insérer deux enregistrements dans la table **Employes**. Chaque enregistrement inclut un numéro d'employé, un nom, un prénom, et un numéro de téléphone.

```
INSERT INTO Employes (num_employe, nom, prenom, tel)  
VALUES
```

```
(1112, "son", "goku", "0689754125"),  
(1113, "aimane", "achhab", "0656560565") ;
```

On va maintenant insérer des données sur les clients dans la table **clients**.

## INSERT INTO

Clients (Refclient, nom\_ste, ville, code\_postal)

## VALUES

```
("C1", "Acom", "Tanger", "90000"),  
("C2", "B2C", "CASA", "40000"),  
("C3", "Tcom", "Rabat", "40000");
```

### 3. Création et manipulation de la table des commandes

#### 3.1. Création de la table des commandes

On va créer une table **Commandes** pour stocker les commandes avec des champs pour la référence de commande, le nom de la société, la date, le montant total, et la valeur de la taxe.

## CREATE TABLE Commandes (

```
Refcom VARCHAR(50) PRIMARY KEY,  
ste VARCHAR(50),  
date DATE,  
somme DECIMAL(10, 2),  
TVA DECIMAL(5, 2));
```

### 3.2. Manipulation de la table des commandes

Dans cela on va ajouter une nouvelle colonne **Refclient** à la table **Commandes** et établit une relation de clé étrangère avec la table **clients**.

Cela nous permet de nous assurer que chaque commande référencée dans la table **Commandes** correspond à un client existant dans la table **clients**. Cette connexion aide à garder les informations organisées et correctes, facilitant la recherche et l'analyse des données sur les commandes et les clients.

**ALTER TABLE** Commandes

**ADD COLUMN** Refclient **CHAR**,

**ADD FOREIGN KEY** (Refclient) **REFERENCES** clients (Refclient) ;

Pour démontrer comment gérer les contraintes de base de données on va ajouter puis supprimer immédiatement une contrainte de clé étrangère

**ALTER TABLE** Commandes

**ADD CONSTRAINT** fk\_clt\_cmd **FOREIGN KEY** (Refclient)

**REFERENCES** clients (Refclient);

**ALTER TABLE** Commandes **DROP FOREIGN KEY** fk\_clt\_cmd;

## 4. Requêtes avancées

En fin pour tester notre data base on va exécuter une requête **SELECT** qui joint les tables clients et commandes sur le champ Refclient. Les résultats sont triés par le nom de la société et la date. Mais pour voir des résultats et non une table vide, on a besoin d'insérer des données dans la table **Commandes**.

**INSERT INTO** commandes

(Refcom, ste, date, somme, TVA, Refclient)

**VALUES**



```
( 'CMD001', 'Acom', '2023-05-01', 1000.00, 20.00, 'C1'),
( 'CMD002', 'B2C', '2023-05-02', 1500.50, 30.00, 'C2'),
( 'CMD003', 'Acom', '2023-05-03', 750.75, 15.00, 'C1'),
( 'CMD004', 'Tcom', '2023-05-04', 2000.00, 40.00, 'C3'),
( 'CMD005', 'B2C', '2023-05-05', 300.00, 6.00, 'C2');
```

Maintenant on va afficher les résultats de la Query.

## SELECT

```
c.nom_ste,
o.Refcom,
o.ste,
o.date,
o.somme,
o.tva
```

FROM clients c

LEFT JOIN commandes o ON c.Refclient = o.Refclient

ORDER BY c.nom\_ste, o.date;

Resultat:

	nom_ste	Refcom	ste	date	somme	tva
►	Acom	CMD001	Acom	2023-05-01	1000.00	20.00
	Acom	CMD003	Acom	2023-05-03	750.75	15.00
	B2C	CMD002	B2C	2023-05-02	1500.50	30.00
	B2C	CMD005	B2C	2023-05-05	300.00	6.00
	Tcom	CMD004	Tcom	2023-05-04	2000.00	40.00

# ATELIER 2 : DEVELOPPER LA SCHEMA D'UNE BASE DE DONNEES AVEC MySQL

Ce deuxième chapitre se concentre sur la mise en place et l'administration d'une base de données pour une bibliothèque universitaire, couvrant la gestion des étudiants, des livres, et des prêts. Nous examinerons la structure des tables, la modification des attributs de table pour répondre aux besoins spécifiques de la bibliothèque, et l'ajout de contraintes pour maintenir l'intégrité des données. En outre, nous illustrerons comment les opérations de base de données peuvent faciliter la gestion quotidienne des prêts de livres et la maintenance des enregistrements d'étudiants et de ressources bibliographiques. Ce chapitre est conçu pour démontrer l'utilisation pratique des bases de données dans un contexte éducatif et organisationnel

## 1. Configuration de la base de données et des tables

### 1.1. Création et configuration de la base Biblio\_sacha

Pour commencer, nous allons créer une base de données appelée Biblio\_sacha et y accéder pour effectuer nos opérations.

On va créer une nouvelle base de données nommée **Biblio\_sacha**.

```
CREATE DATABASE Biblio_sacha ;
```

On va sélectionner la base de données **Biblio\_sacha** pour y effectuer les opérations sur la base de données sélectionnée.

```
USE Biblio_sacha ;
```

### 1.2. Création des tables Etudiant, Livre, Auteur, Editeur, et Thème

La table Etudiant contient les informations sur les étudiants de la bibliothèque.

La commande suivante va créer une table **Etudiant** avec les colonnes suivantes :

- **num\_etudiant** : un entier représentant le numéro de l'étudiant.
- **nom** : une chaîne de caractères de 30 caractères maximum pour le nom de l'étudiant, obligatoire.
- **prenom** : une chaîne de caractères de 30 caractères maximum pour le prénom de l'étudiant, obligatoire.
- **age** : un entier représentant l'âge de l'étudiant.
- **ville** : une chaîne de caractères de 20 caractères maximum pour la ville de l'étudiant.
- **tel** : une chaîne de caractères de 20 caractères maximum pour le numéro de téléphone de l'étudiant.

```
CREATE TABLE Etudiant(  
    num_etudiant INT,  
    nom varchar(30) NOT NULL,  
    prenom VARCHAR(30) NOT NULL,  
    age INT,  
    ville VARCHAR(20),  
    tel VARCHAR(20)) ;
```

Et après on va modifier la colonne **ville** de la table **Etudiant** pour augmenter la taille maximale à 30 caractères et la rendre obligatoire.

```
ALTER TABLE etudiant  
MODIFY ville VARCHAR(30) NOT NULL;
```

On va ajouter une contrainte de clé primaire à la colonne **num\_etudiant** pour garantir l'unicité de chaque étudiant

## ALTER TABLE etudiant ADD CONSTRAINT pk PRIMARY KEY (num\_etudiant);

La table **Livre** contient les informations sur les livres disponibles dans la bibliothèque. La commande suivante va créer une table **Livre** avec les colonnes suivantes :

- num\_livre : une chaîne de caractères de 10 caractères maximum représentant le numéro du livre, clé primaire.
- titre : une chaîne de caractères de 50 caractères maximum pour le titre du livre, obligatoire.
- num\_auteur : une chaîne de caractères de 10 caractères maximum représentant le numéro de l'auteur, obligatoire.
- num\_editeur : une chaîne de caractères de 10 caractères maximum représentant le numéro de l'éditeur, obligatoire.
- num\_theme : une chaîne de caractères de 10 caractères maximum représentant le numéro du thème, obligatoire.
- date\_edition : une date représentant la date d'édition du livre, obligatoire.

```
CREATE TABLE Livre(  
    num_livre VARCHAR(10) PRIMARY KEY,  
    titre VARCHAR(50) NOT NULL,  
    num_auteur VARCHAR(10) NOT NULL,  
    num_editeur VARCHAR(10) NOT NULL,  
    num_theme VARCHAR(10) NOT NULL,  
    date_edition DATE NOT NULL);
```

La table **Auteur** contient les informations sur les auteurs des livres. La commande suivante va créer une table **Auteur** avec les colonnes suivantes :

- num\_auteur : une chaîne de caractères de 10 caractères maximum représentant le numéro de l'auteur, clé primaire.

- nom : une chaîne de caractères de 30 caractères maximum pour le nom de l'auteur, obligatoire.
- adresse : une chaîne de caractères de 50 caractères maximum pour l'adresse de l'auteur.

```
CREATE TABLE Auteur(  
    num_auteur VARCHAR(10) PRIMARY KEY,  
    nom VARCHAR(30) NOT NULL,  
    adresse VARCHAR(50));
```

La table **Editeur** contient les informations sur les éditeurs des livres. La commande suivante va créer une table **Editeur** avec les colonnes suivantes :

- num\_editeur : une chaîne de caractères de 10 caractères maximum représentant le numéro de l'éditeur, clé primaire.
- nom : une chaîne de caractères de 30 caractères maximum pour le nom de l'éditeur, obligatoire.
- adresse : une chaîne de caractères de 50 caractères maximum pour l'adresse de l'éditeur.

```
CREATE TABLE Editeur(  
    num_editeur VARCHAR(10) PRIMARY KEY,  
    nom VARCHAR(30) NOT NULL,  
    adresse VARCHAR(50));
```

La table **Theme** contient les informations sur les thèmes des livres. La commande suivante va créer une table **Theme** avec les colonnes suivantes :

- num\_theme : une chaîne de caractères de 10 caractères maximum représentant le numéro du thème, clé primaire.
- intitule\_theme : une chaîne de caractères de 20 caractères maximum pour l'intitulé du thème, obligatoire.

```
CREATE TABLE Theme(  
    num_theme VARCHAR(10) PRIMARY KEY,  
    intitule_theme VARCHAR(20) NOT NULL);
```

## 2. Modification des structures de tables

### 2.1. Ajout de contraintes pour assurer l'intégrité des données

#### Table Livre.

Maintenant on va ajouter de contraintes de clé étrangère pour les tables **Livre**. Cette commande va ajouter une contrainte de clé étrangère à la colonne num\_auteur de la table **Livre**, référencée par la colonne num\_auteur de la table **Auteur**. Si un auteur est supprimé ou mis à jour, les modifications seront propagées aux livres associés grâce aux options ON DELETE CASCADE et ON UPDATE CASCADE. C'est-à-dire que la suppression ou la mise à jour d'un auteur entraînera automatiquement la suppression ou la mise à jour des livres associés.

```
ALTER TABLE Livre
```

```
ADD CONSTRAINT FK_Livre_Auteur
```

```
FOREIGN KEY (num_auteur)
```

```
REFERENCES Auteur (num_auteur)
```

```
ON DELETE CASCADE
```

```
ON UPDATE CASCADE;
```

On va faire la même chose dans la commande suivante, la suppression ou la mise à jour d'un éditeur entraînera automatiquement la suppression ou la mise à jour des livres associés.

```
ALTER TABLE Livre
```

```

ADD CONSTRAINT FK_Livre_Editeur
FOREIGN KEY (num_editeur)
REFERENCES Editeur (num_editeur)
ON DELETE CASCADE
ON UPDATE CASCADE ;

```

Pour afficher les détails des contraintes de clé étrangère pour la table "**Livre**" dans la base de données "**Biblio\_sacha**", y compris les colonnes et les tables référencées. On va utiliser la commande SQL suivante qui sélectionne des informations sur les contraintes de clé étrangère dans la table "**Livre**" de la base de données "**Biblio\_sacha**"

```

SELECT

```

```

    table_name,
    constraint_name,
    column_name,
    referenced_table_name,
    referenced_column_name

```

```

FROM information_schema.key_column_usage

```

```

WHERE table_name = 'Livre'




```

```

AND table_schema = 'Biblio_sacha';

```

Resultat:

Result Grid    Filter Rows: <input type="text"/>   Export:    Wrap Cell Content: 					
	TABLE_NAME	CONSTRAINT_NAME	COLUMN_NAME	REFERENCED_TABLE_NAME	REFERENCED_COLUMN_NAME
▶	livre	PRIMARY	num_livre	<b>NULL</b>	<b>NULL</b>
	livre	FK_Livre_Auteur	num_auteur	auteur	num_auteur
	livre	FK_Livre_Editeur	num_editeur	editeur	num_editeur

### 3. Modification des structures de tables

#### 3.1. Création de la Table Prêt

Comme les exemples précédents on va commencer par la création de la table **Prêt** de colonnes suivantes :

- num\_etudiant: Entier représentant le numéro de l'étudiant.
- num\_livre: Chaîne de caractères représentant le numéro du livre.
- date\_pret: Date du prêt du livre.
- rendu: Booléen indiquant si le livre a été rendu. Valeur par défaut: FALSE.
- date\_retour: Date de retour du livre.

La clé primaire est une combinaison des colonnes num\_etudiant, num\_livre, et date\_pret, ce qui garantit l'unicité de chaque prêt.

```
CREATE TABLE Pret(  
    num_etudiant INT,  
    num_livre VARCHAR(10),  
    date_pret DATE,  
    rendu BOOLEAN DEFAULT FALSE,  
    date_retour DATE,  
    PRIMARY KEY(num_etudiant,num_livre,date_pret),  
    CONSTRAINT fk_pret_etudiant  
    FOREIGN KEY (num_etudiant)  
    REFERENCES Etudiant (num_etudiant),  
    CONSTRAINT fk_pret_livre  
    FOREIGN KEY (num_livre)
```



## REFERENCES Livre (num\_livre));

Pour assurer que l'âge des étudiants se situe entre 15 et 30 ans, On va utiliser la commande SQL suivante pour ajouter une contrainte CHECK à la table Etudiant.

```
ALTER TABLE Etudiant
```

```
ADD CONSTRAINT chq_age
```

```
CHECK (age BETWEEN 15 AND 30);
```

### 3.2. Ajout de Contraintes Uniques

Maintenant on va ajouter cette contrainte unique qui garantit qu'un livre ne peut être prêté qu'une seule fois à une date donnée.

```
ALTER TABLE pret
```

```
ADD CONSTRAINT uq_pret_uniquedate
```

```
UNIQUE (num_livre, date_pret);
```

On va ajouter une autre contrainte unique qui assure qu'il n'y a pas deux enregistrements de prêt dans la table **Pret** où le même livre (num\_livre) a été rendu (rendu) à la même date de retour (date\_retour). Cela permet de prévenir les doublons et de maintenir l'intégrité des données.

```
ALTER TABLE pret
```

```
ADD CONSTRAINT uq_numlivre_rendu_date_retour
```

```
UNIQUE (num_livre, rendu, date_retour);
```

### 3.3. Ajout de Contraintes de Validation

On va ajouter une contrainte CHECK qui s'assure que la date de retour (date\_retour) respecte l'une des deux conditions suivantes :

1. La date de retour est nulle (le livre n'est pas encore rendu).
2. Si la date de retour n'est pas nulle :
  - Elle doit être après ou égale à la date de prêt (date\_pret).
  - Le livre doit être marqué comme rendu (rendu).

Cela garantit que les dates de prêt et de retour des livres dans la table **Pret** sont cohérentes.

**ALTER TABLE Pret**

**ADD CONSTRAINT CHK\_Pret\_DateRetour**

**CHECK (date\_retour is null**

**OR (date\_retour >= date\_pret and rendu));**

La commande suivante affiche les index de la table Pret, y compris les clés primaires et uniques.

**SHOW INDEX FROM pret;**

Resultat :

Table	Non_unique	Key_name	Seq_in_index	Column_name	Collation	Cardinality	Sub_part	Packed	Null	Index_type	Comment	Index_comment	Visible	Expressio
pret	0	PRIMARY	1	num_etudiant	A	0				BTREE			YES	
pret	0	PRIMARY	2	num_livre	A	0				BTREE			YES	
pret	0	PRIMARY	3	date_pret	A	0				BTREE			YES	
pret	0	uq_pret_uniquedate	1	num_livre	A	0				BTREE			YES	
pret	0	uq_pret_uniquedate	2	date_pret	A	0				BTREE			YES	
pret	0	uq_numlivre_rendu_date_retour	1	num_livre	A	0				BTREE			YES	
pret	0	uq_numlivre_rendu_date_retour	2	rendu	A	0			YES	BTREE			YES	
pret	0	uq_numlivre_rendu_date_retour	3	date_retour	A	0			YES	BTREE			YES	

### 3.4. Insertion de Données dans les tables crée

Pour simuler des opérations sur les tables crée, on peut insérer des enregistrements comme exemples. En utilisant les requêtes suivantes.

**Editeur :**

```
INSERT INTO Editeur(num_editeur, nom, adresse)
VALUES ("E001", "jiji", "");
```

**Auteur :**

```
INSERT INTO Auteur(num_auteur, nom, adresse)
VALUES ("A001", "AKIRA TORIYAMA", "");
```

**Theme:**

```
INSERT INTO Theme(num_theme, intitule_theme)
VALUES ("T001", "shonen");
```

**Livre:**

```
INSERT INTO livre (num_livre, titre, num_auteur,
num_editeur, num_theme, date_edition)
VALUES ('L001', "Dragon ball", "A001", "E001", "T001",
'2023-04-30');
```

**Etudiant :**

```
INSERT INTO Etudiant (num_etudiant, nom, prenom,
age, tel, ville)
VALUES (1, 'SACHA', 'ILYAS', 18, '1234567890', 'Paris');
```

**Prêt :**

```
INSERT INTO Pret (num_etudiant, num_livre,
date_pret, rendu, date_retour)
```

```
VALUES (1, 'L001', '2024-03-01', true, '2024-04-30');
```

On Peut vérifier que les données sont bien insérées en utilisant la requête suivante, tu peux changer le nom de la table pour vérifier les insertions sur les autres tables.

```
SELECT * FROM Pret;
```

Résultat :

Result Grid					
Filter Rows:					
	num_etudiant	num_livre	date_pret	rendu	date_retour
▶	1	L001	2024-03-01	1	2024-04-30

### Conclusion :

Dans ce chapitre ou cet atelier, nous avons appris à créer et modifier une base de données Biblio\_sacha. Nous avons créé des tables pour les étudiants, les livres, les auteurs, les éditeurs, les thèmes et les prêts, en définissant des clés primaires et des contraintes d'intégrité pour garantir la cohérence des données. Nous avons ajouté des contraintes de clé étrangère pour maintenir les relations entre les tables, ainsi que des contraintes uniques et de vérification pour assurer l'unicité et la validité des enregistrements. Enfin, nous avons inséré des données pour tester notre structure de base de données.

# ATELIER 3 : INTERROGER LA BASE DE DONNEES

Dans ce chapitre on va compléter les concepts introduits dans le chapitre précédent en se focalisant sur des requêtes plus complexes et le reporting pour la bibliothèque universitaire. Nous aborderons des techniques avancées pour analyser les données, telles que les requêtes multidimensionnelles qui permettent de dégager des tendances et des modèles d'utilisation des ressources de la bibliothèque.

## 1. Insertion de Données

On va toujours commencer par la sélectionne la base de données **Biblio\_sacha** pour s'assurer que toutes les commandes sont exécutées dans cette base de données spécifique. Cela garantit que les modifications et opérations s'appliquent à la bonne base de données, évitant ainsi des erreurs et la confusion avec d'autres bases de données présentes sur le même serveur.

**USE Biblio\_sacha;**

Comme déjà vu on va insérer des auteurs dans la table **Auteur** avec leurs numéros, noms et adresses.

**INSERT INTO** Auteur (num\_auteur, nom, adresse)

**VALUES**

('H', 'Hakim', 'Rue 123'),  
( 'M', 'Moujarrib', 'Rue 456'),  
( 'N', 'Neferiti', 'Rue 789'),  
( 'R', 'Ramsis', 'Rue 101'),  
( 'T', 'Tafih', 'Rue 102');

On va aussi insérer des thèmes dans la table **Theme** avec leurs numéros et intitulés.

## INSERT INTO

Theme (num\_theme, intitule\_theme)

## VALUES

('Eco', 'Economie'),  
('Info', 'Informatique'),  
('Math', 'Mathématiques'),  
('Div', 'Divers');

On va insérer des éditeurs dans la table **Editeur** avec leurs numéros, noms et adresses.

## INSERT INTO

Editeur(num\_editeur, nom, adresse)

## VALUES

('DAO', 'Dar Al-Oujoum', 'rue abc'),  
('NP', 'Nul Part', 'Av DEF'),  
('PLB', 'Pour les Bêtes', 'RUE GHI');

Et après on va insérer des étudiants dans la table Etudiant avec leurs numéros, noms, prénoms, âges, téléphones et villes en utilisant la commande suivante.

## INSERT INTO

```
etudiant(num_etudiant,nom,prenom, age, tel, ville)
```

## VALUES

```
(50, 'Kaslani', 'Kassoul', 28, '065555555', 'Tanger'),
(51, 'Kaslani', 'Kassoula', 27, '065555556', 'Tanger'),
(100, 'Abbassi', 'Abbass', 23, '070000607', 'Tanger'),
(101, 'Kaddouri', 'Kaddour', 24, '077777700', 'Chefchaouen'),
(102, 'Jallouli', 'Jalloul', 23, '066666660', 'Tétouan'),
(103, 'Ayyachi', 'Aicha', 22, NULL, 'Tétouan'),
(113, 'Slaoui', 'Salwa', 21, '060000001', 'Tanger'),
(202, 'Khaldouni', 'Khalid', 22, '060000002', 'Tanger'),
(309, 'Karimi', 'Karim', 20, '066600005', 'Casa'),
(310, 'Karimi', 'Karima', 20, NULL, 'Casa'),
(567, 'Moussaoui', 'Moussa', 21, '050070070', 'Tanger'),
(580, 'Moussi', 'Moussa', 22, NULL, 'Casa'),
(998, 'Moujtahida', 'Moujidda', 21, NULL, 'Tanger'),
(999, 'Moujtahid', 'Moujidd', 21, NULL, 'Tanger');
```

La commande suivante insère des livres dans la table **Livre** avec leurs numéros, titres, numéros d'auteur, numéros d'éditeur, numéros de thème et dates d'édition.

## INSERT INTO Livre

```
(num_livre, titre, num_auteur, num_editeur, num_theme,
date_edition)
```

## VALUES

```
('BD1', 'Comment avoir 20 en BD', 'R', 'NP', 'Info', '2015-01-01'),
('BD2', 'Tout sur les BD', 'N', 'NP', 'Info', '2014-12-01'),
('BD3', 'Maitriser les BD', 'R', 'NP', 'Info', '2014-07-07'),
('BD4', 'SGBD Relationnels', 'R', 'DAO', 'Info', '2014-01-01'),
('BD5', 'SI et BD', 'N', 'DAO', 'Info', '2003-02-04'),
('BD6', 'Les BD : Pour les nuls', 'R', 'NP', 'Info', '2014-01-01'),
('ECO1', 'L"économie du Maroc en l'an 3050', 'M', 'DAO', 'Eco',
'2015-04-01'),
('Math1', 'Algèbre', 'H', 'NP', 'Math', '2014-09-02'),
('Math2', 'Analyse', 'H', 'NP', 'Math', '2014-08-02'),
('Math3', 'Algèbre linéaire', 'H', 'DAO', 'Math', '2015-08-02'),
('Math4', 'Aimer les Maths', 'M', 'NP', 'Math', '2014-08-04'),
('SE1', 'Systèmes d\'exploitation', 'R', 'NP', 'Info', '2003-08-06'),
('SE2', 'Maitriser UNIX', 'R', 'DAO', 'Info', '2002-10-02'),
('SE3', 'Tout sur les SE', 'N', 'NP', 'Info', '2001-08-07'),
('TW1', 'Histoire', 'T', 'PLB', 'Div', NULL),
('TW2', 'Personnes fameuses', 'T', 'PLB', 'Div', NULL),
('TW3', 'Comment devenir un bon joueur en 5 jours et sans
coach', 'T', 'PLB', 'Div', NULL);
```



Mais il y a un petit problème : dans le deuxième chapitre, nous avons déterminé que la colonne `date_edition` n'acceptera pas de valeurs nulles. Or, dans notre requête, il y a des valeurs nulles. Nous allons donc modifier les caractéristiques de cette colonne en utilisant la commande suivante et ré-exécuter la commande précédente.

```
ALTER TABLE Livre
```

```
MODIFY date_edition DATE;
```

```
ALTER TABLE Livre
```

```
MODIFY titre VARCHAR(100) NOT NULL;
```

## 2. Requêtes SQL pour la Gestion et l'Analyse des Données

### 2.1. Sélection des étudiants ayant plus de 21 ans

Cette commande sélectionne et affiche les noms, prénoms, âges et villes des étudiants ayant plus de 21 ans, triés par nom ascendant et âge descendant.

```
SELECT
```

```
    nom,
```

```
    prenom,
```

```
    age,
```

```
    ville
```

```
FROM Etudiant
```

```
WHERE age > 21
```

```
ORDER BY nom ASC, age DESC;
```

Result Grid		Filter Rows:		Export:		Wrap Cell Content:	
	nom	prenom	age	ville			
▶	Abbassi	Abbass	23	Tanger			
	Ayyachi	Aicha	22	Tétouan			
	Jallouli	Jalloul	23	Tétouan			
	Kaddouri	Kaddour	24	Chefchaouen			
	Kaslani	Kassoul	28	Tanger			
	Kaslani	Kassoula	27	Tanger			
	Khaldouni	Khalid	22	Tanger			
	Moussi	Moussa	22	Casablanca			

## 2.2. Sélection des livres empruntés non rendus

Alors que notre table **Prêt** est vide, nous devons la remplir pour pouvoir effectuer des opérations avec cette table.

### INSERT INTO Pret

(num\_etudiant, num\_livre, date\_pret, rendu,  
date\_retour)

### VALUES

(50, 'BD1', '2024-01-01', false, NULL),  
 (51, 'BD2', '2024-01-05', false, NULL),  
 (100, 'Math1', '2024-02-01', false, NULL),  
 (101, 'ECO1', '2024-02-15', false, NULL),  
 (102, 'Math2', '2024-03-01', false, NULL),  
 (103, 'SE1', '2024-03-05', false, NULL),  
 (113, 'SE2', '2024-04-01', true, '2024-04-15'),

Maintenant pour afficher les noms des auteurs, titres des livres et dates de prêt des livres non rendus, triés par nom d'auteur ascendant et date de prêt. On va utiliser la requête suivante.

## Select

A.nom As AuteurNom,

L.titre As TitreLivre,

P.date\_pret

## From

Auteur A

## Join

Livre L On A.num\_auteur = L.num\_auteur

## Join

Pret P On L.num\_livre = P.num\_livre

## Where

P.rendu = False

## Order By

A.nom Asc, P.date\_pret;

Result Grid			
Filter Rows:		Export:	Wrap Cell Content:
	AuteurNom	TitreLivre	date_pret
▶	Hakim	Algèbre	2024-02-01
	Hakim	Analyse	2024-03-01
	Hakim	Algèbre linéaire	2024-05-01
	Moujarrib	L'économie du Maroc en l'an 3050	2024-02-15
	Neferiti	Tout sur les BD	2024-01-05
	Ramsis	Comment avoir 20 en BD	2024-01-01
	Ramsis	Systèmes d'exploitation	2024-03-05
	Ramsis	Maitriser les BD	2024-04-10

### 3. Opérations sur la Table Professeur

#### 3.1. Création de la Table Professeur

Cette commande crée la table **Professeur** avec les colonnes cin, nom, prenom et adresse.

```
CREATE TABLE Professeur (  
    cin NUMERIC PRIMARY KEY,  
    nom VARCHAR(20) NOT NULL,  
    prenom VARCHAR(20) NOT NULL,  
    adresse TEXT);
```

#### 3.2. Ajout d'une Contrainte Unique -- RequêteLDD3

```
ALTER table Professeur  
ADD CONSTRAINT unique_nom_prenom  
UNIQUE (nom, prenom);
```

#### 3.3. Ajout et Modification de la Colonne email -- RequêteLDD4/DD5

```
ALTER TABLE Professeur  
ADD email VARCHAR(10);  
  
ALTER TABLE Professeur  
MODIFY email VARCHAR(50);
```

### 3.4. Suppression de la Colonne email -- RequêteLDD6

**ALTER TABLE** Professeur

**DROP COLUMN** email;

### 3.5. Suppression de la Table **Professeur** -- RequêteLDD7

**DROP TABLE** Professeur;

## 4. Manipulation des Données dans la Table Etudiant

### 4.1. Insertion de Nouvelles Données

**INSERT INTO**

Etudiant (num\_etudiant,nom,prenom,age,tel, ville)

**VALUES**

(1001,'Hammadi','Hamada',25,'061111111','Casa'),

(1002, 'Tahiri', 'Tahir', 24, '066666600', 'Tanger');

On va maintenant insérer les étudiants suivants dont le tel n'est pas disponible

**INSERT INTO**

Etudiant (num\_etudiant, nom, prenom, age, ville)

**VALUES**

(1003, 'Sallami', 'Salma', 26, 'Tanger'),

(1004, 'Mimouni', 'Mimoun', 23, 'Casa');

## 4.2. Mise à jour de la ville

La requête suivante -- Requête LMD3 met à jour la ville des étudiants de Casa à Casablanca.

**UPDATE** Etudiant e

**JOIN** (

**SELECT** num\_etudiant

**FROM** Etudiant

**WHERE** ville = 'Casa'

) **AS** v **ON** e.num\_etudiant = v.num\_etudiant

**SET** e.ville = 'Casablanca';

## 4.3. Incrémentation de l'Âge

**UPDATE** Etudiant

**SET** age = age + 1

**WHERE** num\_etudiant > 1000;

## 4.4. Incrémentation de l'Âge

**DELETE FROM** Etudiant

**WHERE** num\_etudiant > 1000;

## 4.5. Requetes de Sélection

Maintenant, on va explorer diverses requêtes SQL pour interroger et analyser les données de la base de données de la bibliothèque universitaire. Ces requêtes incluent des sélections basiques, des jointures, des agrégations et des filtres conditionnels pour extraire des informations spécifiques sur les étudiants, les livres, les auteurs, les éditeurs et les prêts. Nous allons utiliser ces requêtes pour répondre à des questions précises et obtenir des insights précieux sur la gestion de la bibliothèque.

- **Requête 7: Sélection par Nom**

```
SELECT nom, prenom, ville AS adresse
```

```
FROM Etudiant
```

```
WHERE nom = 'Moujtahid';
```

<div> <div>Result Grid</div> <div></div> <div></div> <div>Filter Rows:</div> </div>			
	nom	prenom	adresse
▶	Moujtahid	Moujidd	Tanger

- **Requête 8: Sélection des Livres par Auteur**

```
SELECT titre
```

```
FROM Livre
```

```
WHERE num_auteur = 'R';
```

Result Grid		Filter Rows:
	titre	
▶	Comment avoir 20 en BD	
	Maitriser les BD	
	SGBD Relationnels	
	Les BD : Pour les nuls	
	Systèmes d'exploitation	
	Maitriser UNIX	

- Requête 9: Sélection des Livres par Nom d'Auteur

**SELECT** L.titre

**FROM** Livre L

**JOIN** Auteur A **ON** L.num\_auteur = A.num\_auteur

**WHERE** A.nom = 'Ramsis';

Result Grid		Filter Rows:
	titre	
▶	Comment avoir 20 en BD	
	Maitriser les BD	
	SGBD Relationnels	
	Les BD : Pour les nuls	
	Systèmes d'exploitation	
	Maitriser UNIX	



- Requête 10: Sélection du Numéro d'Auteur par Titre de Livre

```
SELECT num_auteur
```

```
FROM Livre
```

```
WHERE titre = 'Comment avoir 20 en BD';
```

Result Grid			Filter Rows:
	num_auteur		
▶	R		

- Requête 11: Sélection des Informations de l'Auteur par Titre de Livre

```
SELECT A.nom, A.adresse
```

```
FROM Livre L
```

```
JOIN Auteur A ON L.num_auteur = A.num_auteur
```

```
WHERE L.titre = 'Comment avoir 20 en BD';
```

Result Grid				Filter Rows:
	nom	adresse		
▶	Ramsis	Rue 101		

- Requête 12: Sélection des Livres par Editeur et Auteur

```
SELECT L.titre
```

```

FROM Livre L
JOIN Editeur E ON L.num_editeur = E.num_editeur
WHERE E.nom = 'Nul Part'
AND L.num_auteur = (SELECT num_auteur
                     FROM Auteur
                     WHERE nom = 'Ramsis');

```

Result Grid		Filter Rows:
	titre	
▶	Comment avoir 20 en BD	
	Maitriser les BD	
	Les BD : Pour les nuls	
	Systèmes d'exploitation	

- Requête 13: Sélection des Livres par Liste d'Auteurs

```

SELECT L.titre
FROM Livre L
JOIN Auteur A ON L.num_auteur = A.num_auteur
WHERE A.nom IN ('Ramsis', 'Neferiti');

```

Result Grid		Filter Rows:	Export:
	titre		
▶	Tout sur les BD		
	SI et BD		
	Tout sur les SE		
	Comment avoir 20 en BD		
	Maitriser les BD		
	SGBD Relationnels		
	Les BD : Pour les nuls		
	Systèmes d'exploitation		
	Maitriser UNIX		

- Requête 14: Sélection des Thèmes par Editeur

```
SELECT DISTINCT T.intitule_theme
FROM Livre L
JOIN Theme T ON L.num_theme = T.num_theme
JOIN Editeur E ON L.num_editeur = E.num_editeur
WHERE E.nom = 'Nul Part';
```

Result Grid		Filter Rows:	Export:
	intitule_theme		
▶	Informatique		
	Mathématiques		

- Requête 15: Sélection des Thèmes et Editeurs

```
SELECT T.intitule_theme, E.nom AS editeur
```

FROM Livre L

JOIN Theme T ON L.num\_theme = T.num\_theme

JOIN Editeur E ON L.num\_editeur = E.num\_editeur;

Result Grid		
	intitule_theme	editeur
▶	Informatique	Dar Al-Oujoum
	Informatique	Dar Al-Oujoum
	Economie	Dar Al-Oujoum
	Mathématiques	Dar Al-Oujoum
	Informatique	Dar Al-Oujoum
	shonen	jiji
	Informatique	Nul Part
	Informatique	Nul Part
	Informatique	Nul Part
	Informatique	Nul Part
	Mathématiques	Nul Part

- Requête 16: Sélection des Livres Spécifique Empruntés par un Etudiant Spécifique

SELECT L.TITRE

FROM Livre L

JOIN Pret P ON L.num\_livre = P.num\_livre

WHERE L.num\_auteur = (SELECT num\_auteur FROM  
Auteur WHERE nom = 'Ramsis')

AND P.num\_etudiant = (SELECT num\_etudiant FROM  
Etudiant WHERE nom = 'Moujtahid');

Result Grid	
	titre

On peut dire que Moujtahid n'a emprunté aucun livre de l'auteur Ramsis.

- **Requête 17: Sélection des Livres Jamais Empruntés**

```
SELECT L.titre
FROM Livre L
LEFT JOIN Pret P ON L.num_livre = P.num_livre
WHERE P.num_livre IS NULL;
```

Result Grid		Filter Rows:	Export:	Wrap Cell Content:
	titre			
▶	SI et BD			
	Les BD : Pour les nuls			
	Aimer les Maths			
	Tout sur les SE			
	Histoire			
	Personnes fameuses			
	Comment devenir un bon joueur en 5 jours et s...			

- **Requête 18: Sélection des Livres d'un Auteur Spécifique Jamais Empruntés**

```
SELECT L.titre
FROM Livre L
LEFT JOIN Pret P ON L.num_livre = P.num_livre
WHERE L.num_auteur = (SELECT num_auteur FROM
Auteur WHERE nom = 'Ramsis');
```

Result Grid	Filter Rows:	Export:	Wrap Cell Content:
titre			
Comment avoir 20 en BD			
Maitriser les BD			
SGBD Relationnels			
Les BD : Pour les nuls			
Systèmes d'exploitation			
Maitriser UNIX			

- **Requête 19: Sélection des Livres et Thèmes Empruntés par un Etudiant Spécifique**

```
SELECT L.titre, T.intitule_theme
FROM Livre L
JOIN Pret P ON L.num_livre = P.num_livre
JOIN Theme T ON L.num_theme = T.num_theme
WHERE P.num_etudiant IN (SELECT num_etudiant
FROM Etudiant
WHERE nom LIKE '%Moujtahid%');
```

Result Grid	Filter Rows:	Export:	Wrap Cell Content:
titre	intitule_theme		

- **Requête 20: Sélection des Livres, Thèmes, et Editeurs Empruntés par un Etudiant Spécifique**

```
SELECT L.titre, T.intitule_theme, E.nom AS EditeurNom
FROM Livre L
```

```

JOIN Pret P ON L.num_livre = P.num_livre
JOIN Theme T ON L.num_theme = T.num_theme
JOIN Editeur E ON L.num_editeur = E.num_editeur
WHERE P.num_etudiant IN (
    SELECT num_etudiant
    FROM Etudiant
    WHERE nom LIKE '%Moujtahid%');

```

Result Grid			Filter Rows:	<input type="text"/>	Export:		Wrap Cell Content:	
	titre	intitule_theme	EditeurNom					

- **Requête 21: Sélection des Thèmes Non Empruntés par un Etudiant Spécifique**

```

SELECT DISTINCT T.intitule_theme
FROM Theme T
WHERE NOT EXISTS (
    SELECT 1
    FROM Pret P
    JOIN Livre L ON P.num_livre = L.num_livre
    WHERE L.num_theme = T.num_theme
    AND P.num_etudiant = 999);

```

Result Grid	Filter Rows:	Export:	Wrap Cell Content:
intitule_theme			
Divers			
Economie			
Informatique			
Mathématiques			
shonen			

- **Requête 22: Sélection des Livres Empruntés par un Etudiant Spécifique**

```
SELECT L.titre, P.date_pret
FROM Pret P
JOIN Livre L ON P.num_livre = L.num_livre
JOIN Etudiant E ON P.num_etudiant = E.num_etudiant
WHERE E.nom LIKE '%Kaslani%';
```

Result Grid	Filter Rows:	Export:	Wrap Cell Content:
titre	date_pret		
Comment avoir 20 en BD	2024-01-01		
Tout sur les BD	2024-01-05		

- **Requête 23: Sélection du Livre Non Rendu le Plus Ancien**

```
SELECT L.titre, E.nom AS Emprunteur, P.date_pret
FROM Pret P
JOIN Livre L ON P.num_livre = L.num_livre
JOIN Etudiant E ON P.num_etudiant = E.num_etudiant
WHERE P.rendu = FALSE
```



ORDER BY P.date\_pret ASC

LIMIT 1;

Result Grid	Filter Rows:	Export:	Wrap Cell Content:
titre	Emprunteur	date_pret	
Comment avoir 20 en BD	Kaslani	2024-01-01	

- Requête 24: Sélection des Livres, Auteurs, Editeurs et Thèmes

SELECT

L.titre,

A.nom AS AuteurNom,

E.nom AS EditeurNom, T.intitule\_theme

FROM Livre L

JOIN Auteur A ON L.num\_auteur = A.num\_auteur

JOIN Editeur E ON L.num\_editeur = E.num\_editeur

JOIN Theme T ON L.num\_theme = T.num\_theme;

Result Grid	Filter Rows:	Export:	Wrap Cell Content:
titre	AuteurNom	EditeurNom	intitule_theme
Comment avoir 20 en BD	Ramsis	Nul Part	Informatique
Tout sur les BD	Neferiti	Nul Part	Informatique
Maitriser les BD	Ramsis	Nul Part	Informatique
SGBD Relationnels	Ramsis	Dar Al-Oujoum	Informatique
SI et BD	Neferiti	Dar Al-Oujoum	Informatique
Les BD : Pour les nuls	Ramsis	Nul Part	Informatique
L'économie du Maroc en l'an 3050	Moujarrib	Dar Al-Oujoum	Economie
Dragon ball	AKIRA TORIYAMA	jiji	shonen
Algèbre	Hakim	Nul Part	Mathématiques
Analyse	Hakim	Nul Part	Mathématiques
Al-Bh...	Hakim	Dar Al-Oujoum	Mathématiques

Result 21 x

- **Requête 25: Sélection des Livres Empruntés dans une Période Spécifique**

```
SELECT L.titre
FROM Pret P
JOIN Livre L ON P.num_livre = L.num_livre
WHERE P.date_pret
BETWEEN '2024-01-01' AND '2024-05-31';
```

Result Grid		Filter Rows:	Export:	Wrap Cell Content:
	titre			
▶	Comment avoir 20 en BD			
	Tout sur les BD			
	Maitriser les BD			
	SGBD Relationnels			
	L'économie du Maroc en l'an 3050			
	Dragon ball			
	Algèbre			
	Analyse			
	Algèbre linéaire			
	Systèmes d'exploitation			



- **Requête 26: Sélection des Livres Contenant un Mot Spécifique dans le Titre**

```
SELECT titre
FROM Livre
WHERE titre LIKE '%BD%';
```

Result Grid		Filter Rows:	Export:	Wrap Cell Content:
	titre			
▶	Comment avoir 20 en BD			
	Tout sur les BD			
	Maitriser les BD			
	SGBD Relationnels			
	SI et BD			
	Les BD : Pour les nuls			



- **Requête 27: Nombre de Livres par Auteur**

```
SELECT A.nom, COUNT(L.num_livre) AS NombreLivres
FROM Auteur A
JOIN Livre L ON A.num_auteur = L.num_auteur
GROUP BY A.nom;
```

Result Grid		
Filter Rows: <input type="text"/>		
Export:  Wrap Cell Content: 		
	nom	NombreLivres
▶	AKIRA TORIYAMA	1
	Hakim	3
	Moujarrib	2
	Neferiti	3
	Ramsis	6
	Tafih	3

- **Requête 28: Âge Moyen des Étudiants par Ville**

```
SELECT ville, AVG(age) AS AgeMoyen
FROM Etudiant
GROUP by ville;
```

Result Grid		
Filter Rows: <input type="text"/>		
Export:  Wrap Cell Content: 		
	ville	AgeMoyen
▶	Paris	18.0000
	Tanger	23.4000
	Chefchaouen	24.0000
	Tétouan	22.5000
	Casablanca	20.6667
	Casa	24.0000

- **Requête 29: Livres Empruntés Plus de Trois Fois**

```
SELECT L.titre, COUNT(*) AS NombreEmprunts
FROM Pret P
```

```
JOIN Livre L ON P.num_livre = L.num_livre
GROUP BY L.titre
HAVING COUNT(*) > 3;
```

Result Grid			Filter Rows: <input type="text"/>	Export:	Wrap Cell Content:
	titre	NombreEmprunts			

- **Requête 30: Livres Non Rendus Empruntés Plus de Trois Fois**

```
SELECT L.titre, COUNT(*) AS NombreEmprunts
FROM Pret P
JOIN Livre L on P.num_livre = L.num_livre
WHERE P.rendu = false
GROUP BY L.Titre
HAVING COUNT(*) > 3;
```

Result Grid			Filter Rows: <input type="text"/>	Export:	Wrap Cell Content:
	titre	NombreEmprunts			

- **Requête 31: Livres d'un Auteur Spécifique Empruntés Plus de Quatre Fois**

```
SELECT L.titre, COUNT(*) AS NombreEmprunts
FROM Pret P
JOIN Livre L ON P.num_livre = L.num_livre
```

WHERE L.num\_auteur = (SELECT num\_auteur FROM  
Auteur WHERE nom = 'Ramsis')

GROUP BY L.titre

HAVING COUNT(\*) > 4;

Result Grid		Filter Rows:		Export:		Wrap Cell Content:	
	titre	NombreEmprunts					

- Requête 32: Nombre de Livres Empruntés par Etudiant

SELECT

E.num\_etudiant,

E.nom,

COUNT(DISTINCT P.num\_livre) AS NombreLivres

FROM Pret P

JOIN Etudiant E ON P.num\_etudiant = E.num\_etudiant

WHERE YEAR(P.date\_pret) = 2024 AND E.age < 23

GROUP BY E.num\_etudiant, E.nom

HAVING COUNT(DISTINCT P.num\_livre);

Result Grid		Filter Rows:		Export:		Wrap Cell Content:	
	num_etudiant	nom	NombreLivres				
▶	103	Ayyachi	1				
	113	Slaoui	1				
	202	Khaldouni	1				
	309	Karimi	1				
	310	Karimi	1				

## Conclusion

En travaillant sur cet atelier SGBD, nous avons appris à créer et gérer une base de données pour une bibliothèque universitaire en utilisant MySQL. Nous avons effectué des insertions de données dans différentes tables, modifié des structures de tables et ajouté des contraintes pour assurer l'intégrité des données. En outre, nous avons exploré des requêtes SQL complexes pour interroger et analyser les données, notamment pour sélectionner des étudiants et des livres en fonction de critères spécifiques. Ce processus nous a permis de renforcer notre compréhension des concepts de gestion de bases de données relationnelles et d'optimiser les pratiques de manipulation de données.

# CONCLUSION GENERALE

À travers cet atelier, nous avons acquis une compréhension approfondie de la gestion de bases de données relationnelles en utilisant MySQL. Nous avons appris à créer et configurer des bases de données, à définir des tables avec des clés primaires et des contraintes d'intégrité, et à effectuer des insertions, des modifications et des suppressions de données. En outre, nous avons exploré des requêtes SQL avancées pour interroger et analyser les données, ce qui nous a permis d'optimiser la gestion des informations dans différents contextes, tels que l'administration d'une entreprise fictive et d'une bibliothèque universitaire. Ces compétences sont essentielles pour toute gestion efficace de bases de données, facilitant la prise de décisions basée sur des données précises et bien structurées.