



ib  
cegos



ib  
cegos

**FORMATION SQL**





# PLAN

- Introduction
- Règle de passage diagramme de classe UML au modèle relationnel
- Conception d'une base de données relationnelle
- Manipulation d'une base de données relationnelle
- Opérations conditionnelles
- Les fonctions
- Les jointures
- Transaction
- Trigger
- Terminal et gestion des privilèges

# INSTALLATION

Pour le SGBDR nous utiliserons Laragon (version complète), c'est un logiciel gratuit, local, intégrant les serveurs : MySQL, PHP, Apache et NodeJS.

Lien de téléchargement : <https://laragon.org/download/index.html>

Pour Mac on peut utiliser XAMPP :

Lien de téléchargement : <https://www.apachefriends.org/fr/index.html>

Pour visualiser Les BDD avec XAMPP : <http://localhost/phpmyadmin>

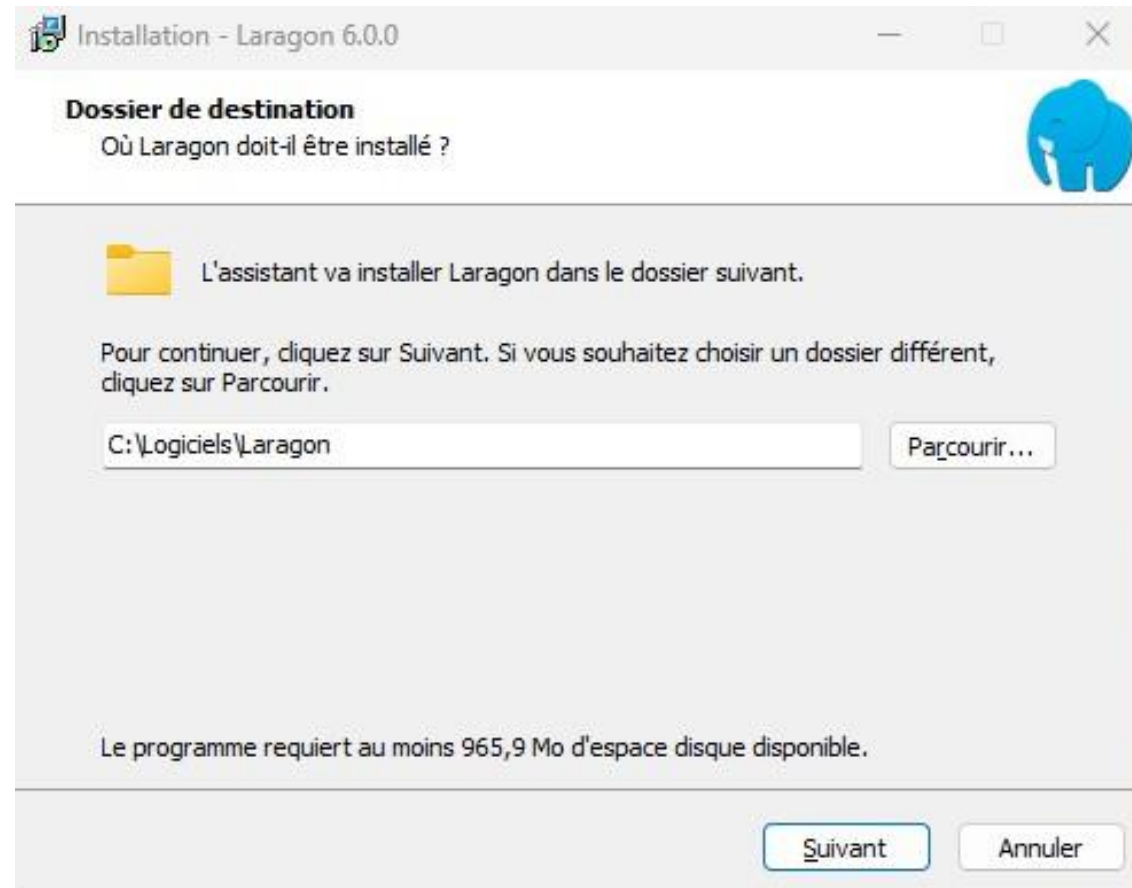
# INSTALLATION

Pour rajouter une autre version de MySQL Community Server à Laragon, il suffit de télécharger la version zipper voulue à l'adresse suivante :

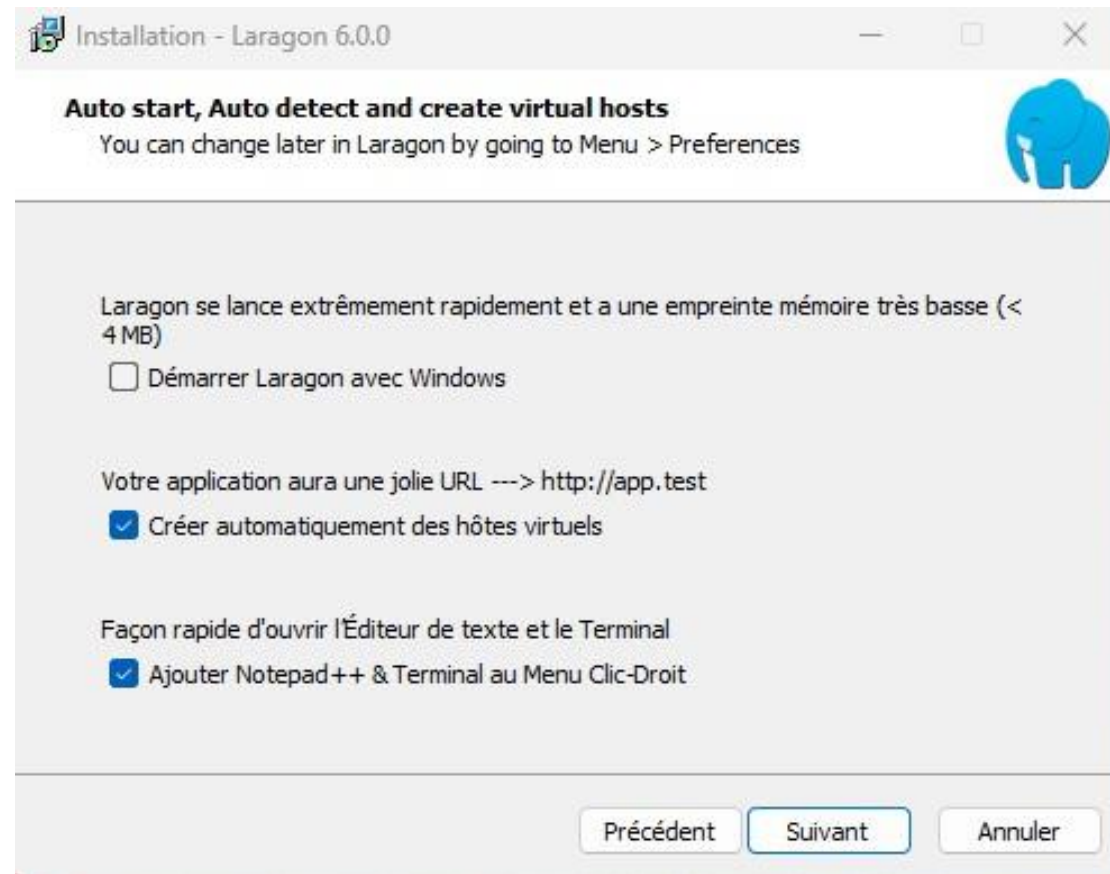
<https://dev.mysql.com/downloads/mysql/>

Puis dézipper le fichier téléchargé et copier le dossier obtenu dans le sous-dossier \bin\mysql du dossier d'installation de Laragon.

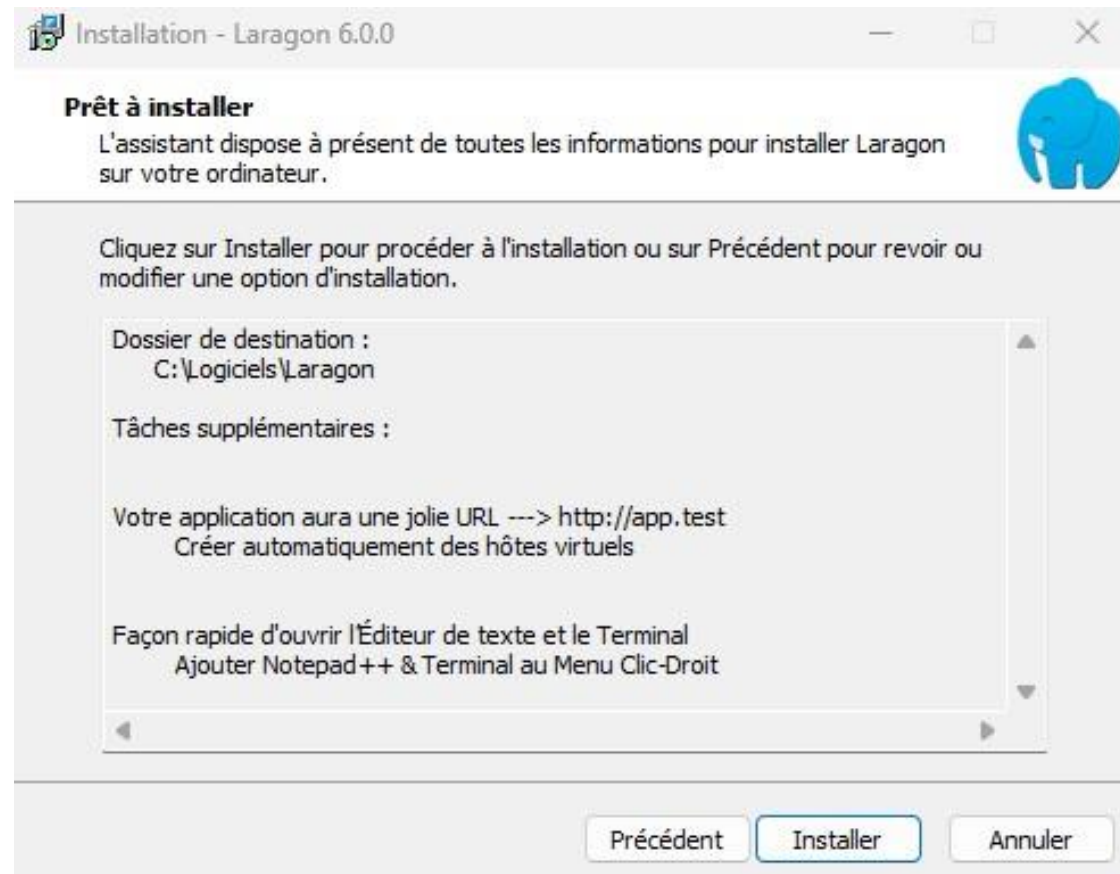
# INSTALLATION



# INSTALLATION



# INSTALLATION





# INSTALLATION

Après l'installation de Laragon, on doit l'exécuter en tant qu'administrateur. Pour cela faire un clique droit sur l'icone de Laragon puis choisir « propriété ». Sélectionner l'onglet compatibilité et cocher la case « exécuter en tant qu'administrateur ».

# INSTALLATION

Pour rajouter une autre version de MySQL Community Server à Laragon, il suffit de télécharger la version zipper voulue à l'adresse suivante :

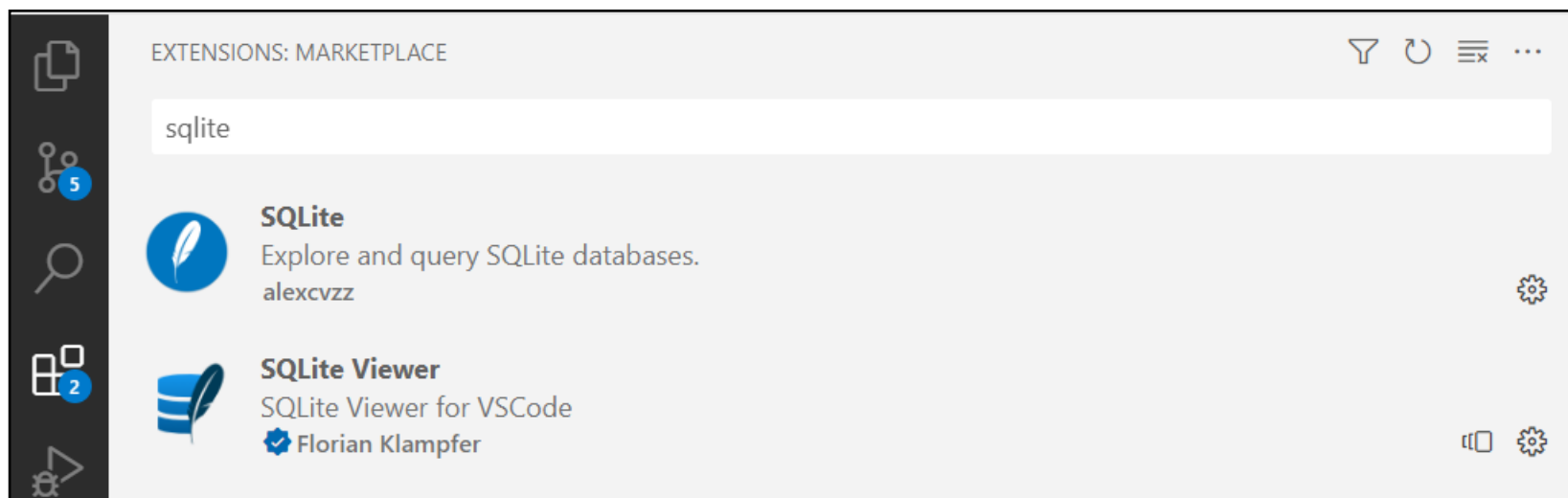
<https://dev.mysql.com/downloads/mysql/>

Puis dézipper le fichier téléchargé et copier le dossier obtenu dans le sous-dossier \bin\mysql du dossier d'installation de Laragon.

# INSTALLATION

Il est possible d'utiliser SQL en installant des extensions :

- Extension Draw.io sur Visual Studio Code pour les dessins UML
- Extensions SQLite





ib  
cegos

# 1. INTRODUCTION





# QU'EST-CE QU'UNE BASE DE DONNEES ?

## – Une base de données

- ▶ Un ensemble cohérent des données qui sont stockées dans un serveur local ou distant
- ▶ Les données sont organisées en tables
- ▶ Les tables sont reliées entre elles

# QU'EST-CE QUE SQL ?

## – Structured Query Language (SQL)

- ▶ Langage de requête structuré.
- ▶ Standard pour communiquer avec des bases des données relationnelles.
- ▶ Intermédiaire entre un client demandeur de service (par exemple une application) et un serveur de base de données (fournisseur des données).

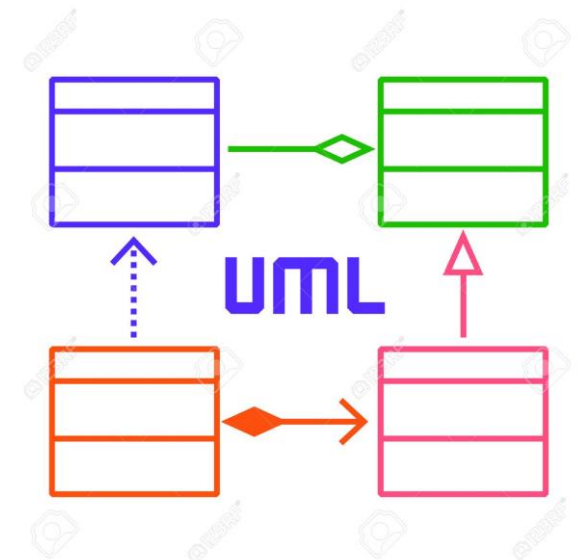
# QU'EST-CE QUE SQL ?

- Le langage SQL permet d'effectuer des requêtes sur une base de données depuis :
  - ▶ Un terminal
  - ▶ Un logiciel
  - ▶ Un langage de programmation
- SQL est composé de 4 sous-langages (appelés aussi sous-ensembles) que sont LDD, LMD, LCT et LCD.

# QUE FAIRE AVEC UNE BASE DE DONNEES ?

## — Faire des requêtes sur le modèle Client / Serveur

- ▶ Récupérer des données
- ▶ Créer des données
- ▶ Mettre à jour des données
- ▶ Supprimer des données
- ▶ Administrer des données
- ▶ Gérer les utilisateurs qui ont accès aux données





# LES DIFFERENTES BDD RELATIONNELLES

## — Histoire

- ▶ Première apparition dans les années 60 (IBM)
- ▶ Commercialisation dans les années 80 (Oracle)
- ▶ Popularisation dans les années 90

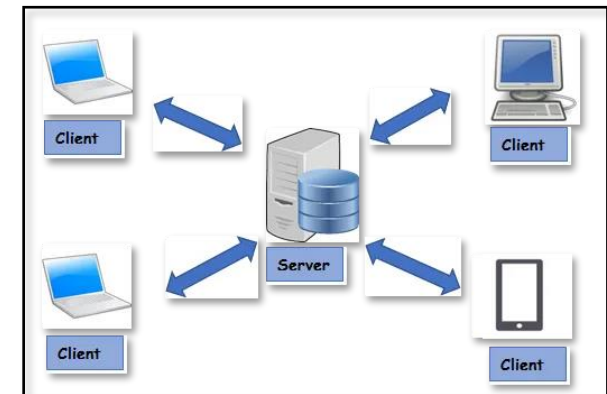
## — SGBDR

- ▶ Système de Gestion de Base de Données Relationnelle
- ▶ En anglais RDBMS (Relational DataBase Management System)



# INTERET D'UN SGBDR ?

- Stocker des informations de manière persistante dans un endroit centralisé et sécurisé
- Accéder aux informations depuis plusieurs supports
- Partager des données avec d'autres applications (APIs)
- Éviter la répétition des données (redondance) et les incohérences
- Maintien l'intégrité des données

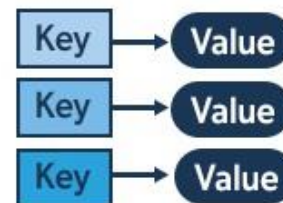


# LES BASES DE DONNEES NOSQL

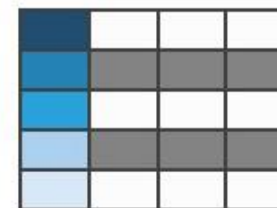
- Not Only SQL
- Pas de structure déterminée
- Pas d'utilisation du langage SQL
- Clés-valeurs
  - ▶ Redis
  - ▶ Memcached
  - ▶ SimpleDB
- Stockage orienté colonne
  - ▶ Elasticsearch
  - ▶ Spark SQL
- Orienté document
  - ▶ MongoDB
  - ▶ Cassandra
- Graph
  - ▶ FlockDB
  - ▶ OrientDB

## NoSQL

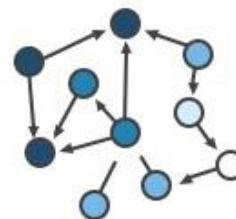
### Key-Value



### Column-Family



### Graph



### Document



# SOUS LANGAGE DE SQL

## – LDD (Langage de Définition de Données, en anglais DDL)

- ▶ Le premier sous-langage de SQL, le LDD, est utilisé pour créer et modifier la structure d'une base de données. Il comprend des commandes telles que CREATE, ALTER et DROP.
- ▶ CREATE est utilisé pour créer de nouveaux objets de base de données tels que des tables, des vues, des index et des déclencheurs.
- ▶ ALTER est utilisé pour modifier la structure des objets existants.
- ▶ DROP est utilisé pour supprimer des objets de la base de données.

# SOUS LANGAGE DE SQL

## – **LMD** (Langage de Manipulation de Données, en anglais DML)

- ▶ Le deuxième sous-langage, LMD, est utilisé pour manipuler les données de la base de données. Il comprend des commandes telles que SELECT, INSERT, UPDATE et DELETE.
- ▶ SELECT est utilisé pour extraire des données d'une ou de plusieurs tables.
- ▶ INSERT est utilisé pour ajouter de nouvelles données à une table.
- ▶ UPDATE est utilisé pour modifier des données existantes.
- ▶ DELETE est utilisé pour supprimer des données d'une table.

# SOUS LANGAGE DE SQL

## – LCT (Langage de Contrôle des Transactions, en anglais TCL)

- ▶ Le troisième sous-langage, LCT, est utilisé pour gérer les transactions au sein d'une base de données. Il comprend des commandes telles que COMMIT et ROLLBACK.
- ▶ COMMIT est utilisé pour enregistrer les modifications dans la base de données.
- ▶ ROLLBACK est utilisé pour annuler ces modifications.

# SOUS LANGAGE DE SQL

## – LCD (Langage de Contrôle de Données, en anglais DCL)

- ▶ Le quatrième sous-langage, LCD, est utilisé pour contrôler l'accès à la base de données. Il comprend des commandes telles que GRANT et REVOKE.
- ▶ GRANT est utilisé pour donner aux utilisateurs des autorisations d'accès à certains objets de la base de données.
- ▶ REVOKE est utilisé pour supprimer ces autorisations.

## 2. RÈGLES DE PASSAGE DU DIAGRAMME DE CLASSE UML AU MODÈLE RELATIONNEL





# MODELISATION

## – UML

- ▶ Langage de modélisation unifié.
- ▶ Dans le cadre d'un développement d'une application orienté objet et à partir d'un diagramme de classe (niveau conceptuel), on peut modéliser une base de données pour cette application.
- ▶ Il existe des règles permettant de transformer un diagramme de classe en base de données relationnelle.

# RAPPEL DIAGRAMME DE CLASSE UML

## — Principes

- ▶ Schématiser la structure interne d'un système qui sera implémenté dans un langage orienté objet.
  - Classes
  - Attributs
  - Opérations
  - Relations
- ▶ Représenter les données et les traitements du système.
- ▶ Modéliser des bases de données relationnelles ou objets.
- ▶ Le niveau d'abstraction ou de détail dépend de vos objectifs et de la phase à laquelle le projet se trouve.

# LES ATTRIBUTS

- Les classes (modèle conceptuel) se transforment en table (modèle relationnel)
- Attributs des classes deviennent attributs de table (les colonnes de la table)
- Une instance d'une classe est un enregistrement d'une table (appelé tuple ou ligne)
- Les relations entre les classes en UML se transforment en contraintes sur les tables en BDDR.

# LES CONTRAINTES DES DONNEES

Type de contraintes	Correspondance dans le SGBDR
Clé primaire	PRIMARY KEY
Valeur non nulle	NOT NULL
Valeur par défaut	DEFAULT
Incrémentation	AUTO_INCREMENT
Contrainte	CONSTRAINT
Condition	CHECK
Clé étrangère (Référence à une autre table)	FOREIGN KEY fk_name REFERENCES table(field)
Unicité d'une donnée	UNIQUE

# CLE PRIMAIRE

- En UML 2 instances qui ont les mêmes attributs ne sont pas identiques, car ils n'ont pas la même référence en mémoire.
- Dans le modèle relationnel, 2 lignes identiques (ayant les mêmes valeurs pour chaque attribut) provoquent de la redondance (doublons). Pour éviter cette incohérence, dans le modèle relationnel, il y a la notion de clé primaire.

# CLE PRIMAIRE

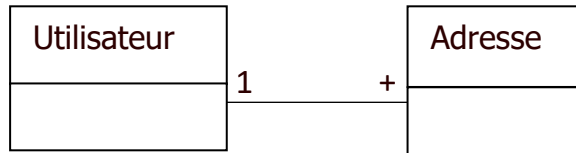
- Attribut qui permet d'assurer la cohérence des données et d'éviter des doublons.
- Contrainte d'unicité d'une donnée donc ne peut pas être nulle.
- À déterminer selon les attributs de la table, en se posant la question, est-ce qu'il y a un attribut qui assure l'unicité d'une ligne?
  - ▶ Oui, alors il est candidat.
  - ▶ Non, alors on crée un nouvel attribut qu'on nomme généralement ID de type nombre.
- La clé primaire peut être un ensemble de plusieurs attributs.
- S'il y a plusieurs candidats, la règle est de choisir la clé primaire minimale (celle qui a le moins de colonne) ou la moins complexe.
- Dans le modèle relationnel, la clé primaire [PK] est isolée des autres attributs.

# CLE ETRANGERE – ASSOCIATION 1 A N

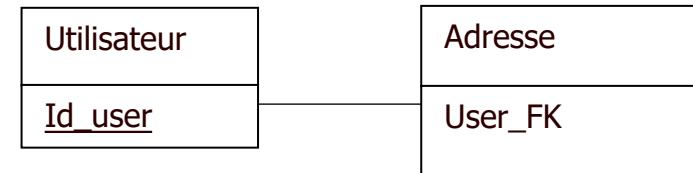
- Les associations dans UML sont des relations dans le modèle relationnel.
- Les associations :
  - ▶ 1 à plusieurs impose la création d'une clé étrangère (colonne supplémentaire) dans le modèle relationnel dans la table qui a la multiplicité plusieurs.
- Une clé étrangère :
  - ▶ Colonne supplémentaire permettant de maintenir la relation entre les tables.
  - ▶ Visible dans le modèle grâce à la notation [FK] après le nom de l'attribut.
  - ▶ La clé étrangère dans une table fait référence à la clé primaire d'une autre table. Donc, elle peut être composé d'une ou de plusieurs colonnes.
  - ▶ Selon le SGBDR on peut rajouter des options à la création de cette colonne.

# CLE ETRANGERE – ASSOCIATION 1 A N

UML



BDD



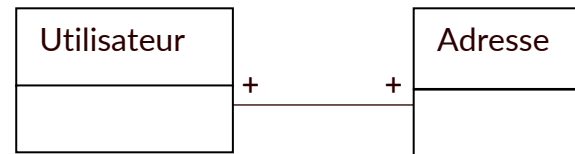


# ASSOCIATIONS N A N

- Association plusieurs à plusieurs dans le diagramme de classe donne lieu dans le modèle relationnel
  - ▶ A la création d'une table supplémentaire composé de deux nouveaux attributs qui sont des clés étrangères faisant respectivement référence à la clé primaire des autres tables.
  - ▶ La clé primaire de cette nouvelle table est composé des 2 clés étrangères qui font chacune référence à une table.

# ASSOCIATIONS N A N

UML



BDD

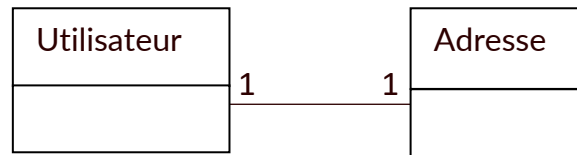


# ASSOCIATIONS 1 A 1

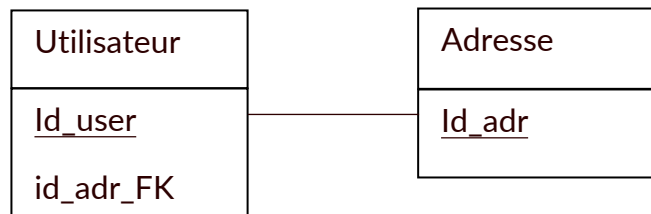
- On a le choix de placer la clé étrangère dans la table que l'on souhaite, cependant, il faut rajouter une contrainte d'unicité pour empêcher la redondance des données au niveau de la clé étrangère.

# ASSOCIATIONS 1 A 1

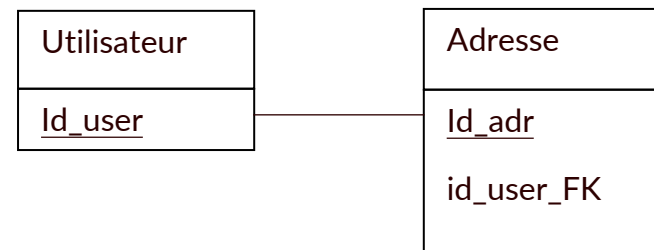
UML



BDD



OU

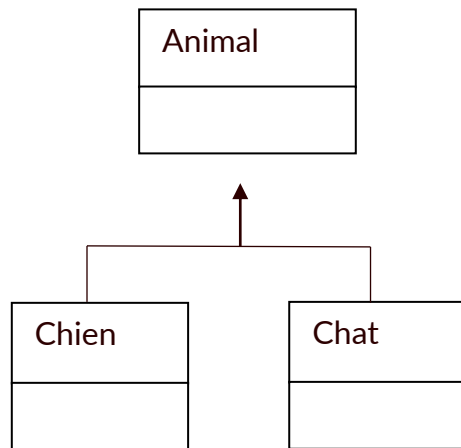


# HERITAGE

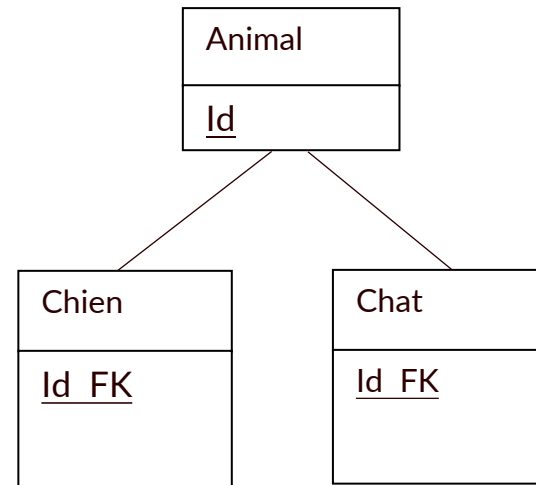
- Il n'y a pas de notion d'héritage dans le modèle relationnel.
- Plusieurs solutions existent pour le mettre en place, la plus simple est l'héritage par référence.
- Toutes les classes impliquées dans une relation d'héritage seront créées.
  - ▶ Dans chaque classe fille, la clé primaire est la même que celle de la classe mère.
  - ▶ La clé primaire de chaque classe fille est également une clé étrangère qui référence la clé primaire de la classe mère.

# HERITAGE

UML



BDD

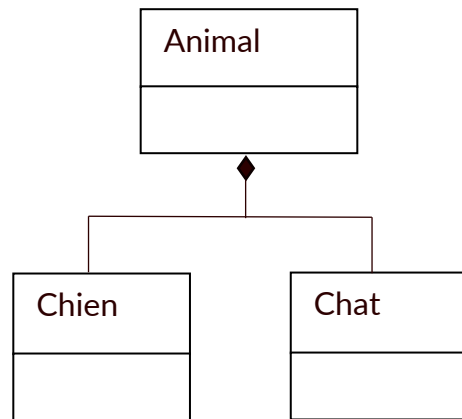


# COMPOSITION

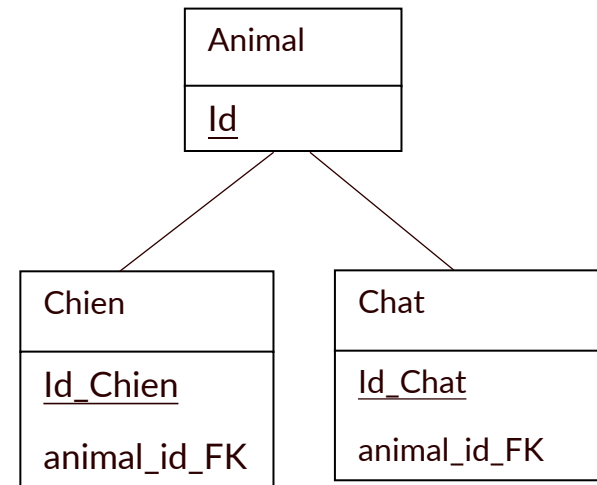
- On traduit une composition par la création d'une nouvelle table comme pour une association plusieurs à plusieurs. De plus, on ajoute la clé primaire de la table composant et la clé étrangère vers la table composite.
- Il est possible aussi de faire un équivalent de composition sans classe intermédiaire.

# COMPOSITION

## UML



## BDD





### 3. CONCEPTION D'UNE BASE DE DONNÉES RELATIONNELLE



# REPRESENTATION D'UNE TABLE

Nom	Format	Longueur maximum	Type de données	Règle de gestion	Exemple
Nom	alphanumérique	50	Chaîne de caractère		Tabuteau
Prénom	alphanumérique	50	Chaîne de caractère		Xavier
Age	nombre	3	Nombre	Entre 16 et 64 ans	50
Email	alphanumérique	80	Chaîne de caractère		<u>mon@email.fr</u>
Ville	alphanumérique	80	Chaîne de caractère		Poitiers

# TYPE DE DONNEES

CATEGORIE	TYPES DANS UNE BASE DE DONNÉES
Alphanumérique	VARCHAR, TEXT, etc.
Numérique	INTEGER, DECIMAL, BOOLEAN
Date et heure	DATE, DATETIME, TIME

Les types de données auront une taille, soit par défaut soit à spécifier. Cette taille correspond à l'espace de stockage qui sera réservé pour stocker un enregistrement (ligne) dans la base de données.

# CREATION

## – Bonne pratique

- ▶ Le nom de la base de données en Anglais, en minuscule, au singulier, sans espace et sans accent ni ponctuation.
- ▶ « \_ » pour séparer les mots qui composent le nom de la base de données.
- ▶ Pour la rédaction des requêtes SQL, écrire les mots clé de SQL en majuscules. Vous pouvez aller à la ligne et indenter pour rendre le code plus lisible.
- ▶ Il est possible de rajouter des options lors de la création, les options dépendent de l'implémentation des différents SGBDR.
- ▶ Le « ; » sert à finir la commande SQL. Cela permet d'écrire un script de plusieurs commandes SQL avant de tout exécuter en une seule fois. Si on écrit une seule commande SQL, le « ; » n'est pas nécessaire.



# CREATION

La création d'une BDDR se base sur le premier sous-langage de SQL.

Cela se fait par la commande CREATE. Elle permet de créer une base de données ainsi que les tables de celle-ci. Elle sert aussi à créer des utilisateurs, des vues, des index ...

# CREATION

Syntaxe pour créer une base de données :

```
CREATE DATABASE NomBase;
```

Syntaxe pour créer une table :

```
CREATE TABLE nom_table (  
    colonne1 type_de_donnee1,  
    colonne2 type_de_donnee2,  
    ...  
);
```

# CREATION D'UNE TABLE AVEC CLE PRIMAIRE

- Exemple de création de table

```
CREATE TABLE books (  
    id INTEGER PRIMARY KEY AUTOINCREMENT,  
    title VARCHAR(150) NOT NULL  
);
```

Dans cette exemple on créer une clé primaire nommée id, qui sera auto incrémenté par le SGBDR à chaque ajout de lignes.

Le NOT NULL signifie qu'on ne peut pas ajouter une ligne sans titre à un livre.

# CREATION D'UNE TABLE AVEC CLE ETRANGERE

- Exemple de création de table

```
CREATE TABLE Orders (  
    OrderID int NOT NULL,  
    OrderNumber int NOT NULL,  
    PersonneID int,  
    PRIMARY KEY (OrderID),  
    CONSTRAINT FK_PersonOrder FOREIGN KEY (PersonID)  
    REFERENCES Persons(PersonID)  
);
```

- **CONSTRAINT** AliasCleEtrangere n'est pas obligatoire. Il permet juste de pouvoir manipuler plus facilement la contrainte (suppression par exemple).



# ENUMERATION

## — Exemple

```
CREATE TABLE shirts (  
    name VARCHAR(40),  
    size ENUM('xs', 's', 'm', 'l', 'xl')  
);
```

Type de données qui restreint les valeurs possibles d'une colonne. Liste fermée. Aucune autre valeur n'est possible hormis celles listées. ENUM n'est pas supportée par tous les SGBDR.

# SUPPRESSION

La suppression d'une BDDR se base sur le premier sous-langage de SQL.

Cela se fait par la commande DROP. Elle permet de supprimer une base de données ainsi que les tables de celle-ci. Elle sert aussi à supprimer des utilisateurs, des vues, des index ...

Pour supprimer une base de données la syntaxe est la suivante :

```
DROP DATABASE NomBase;
```

Pour supprimer une table :

```
DROP TABLE nom_table;
```

# MODIFICATION

La modification d'une BDDR se base sur le premier sous-langage de SQL.

Cela se fait par la commande ALTER. Elle permet d'ajouter, supprimer ou modifier une colonne d'une table de la base de données. Elle sert aussi à renommer une table, ajouter une clé étrangère, ...

# MODIFICATION

Syntaxe générale :

-- ajoute une colonne

```
ALTER TABLE nom_table ADD COLUMN nom_colonne type_colonne;
```

-- supprime une colonne

```
ALTER TABLE nom_table DROP COLUMN nom_colonne;
```

-- modifier le type d'une colonne

```
ALTER TABLE nom_table MODIFY COLUMN nom_colonne  
nouveau_type;
```

-- renommer une table

```
ALTER TABLE nom_table RENAME TO nouveau_nom_table;
```

# CHECK

- Contrainte de vérification des données lors de l'enregistrement des données.
- Empêche l'insertion, lorsque la contrainte n'est pas respectée.
- Les valeurs nulles sont acceptées dans la contrainte.

Une contrainte CHECK peut contenir une vérification sur plusieurs colonnes.

Attention, ne pas être trop strict dans les conditions au risque d'empêcher l'insertion de certaines valeurs tout à fait valides.

# CHECK

Exemple :

```
ALTER TABLE Persons ADD CONSTRAINT Chk_Age CHECK (Age >= 18);
```

```
ALTER TABLE Persons ADD CHECK (Age >= 18);
```

```
CREATE TABLE Persons (  
    id int PRIMARY KEY, Prenom VARCHAR(50),  
    Age int CHECK (Age >= 18 AND Age <= 99) );
```

# UNIQUE

UNIQUE est une contrainte d'unicité. Cela garantit que toutes les valeurs dans la ou les colonnes concernées n'ont pas de doublons.

Exemples :

```
CREATE TABLE utilisateurs (  
    id INT AUTO_INCREMENT PRIMARY KEY,  
    nom VARCHAR(50),  
    email VARCHAR(100) UNIQUE);
```

```
CREATE TABLE exemple (  
    id INT AUTO_INCREMENT PRIMARY KEY,  
    colonne1 INT,  
    colonne2 VARCHAR(50),  
    UNIQUE (colonne1, colonne2) );
```

# NORMALISATION

## — Les formes normales

Une forme normale caractérise le fait que la structure de la base respecte certaines contraintes de modélisation.

Le respect par une base de données d'une forme normale permet de garantir certaines propriétés.

La normalisation consiste à restructurer une base de données pour respecter certaines formes normales, afin d'éviter la redondance des données (des données apparaissent plusieurs fois) et d'assurer l'intégrité des données.

Liste des formes normales : [Formes normales](#)



## 4. MANIPULATION D'UNE BASE DE DONNÉES RELATIONNELLE



# SIGNIFICATION DE CRUD

Cela correspond au deuxième sous-langage de SQL.

CRUD est un acronyme qui représente les opérations de base effectuées sur les données dans une BDD. Chaque lettre de l'acronyme correspond à une opération spécifique :

- C - Create (Créer) : Cette opération implique l'ajout de nouvelles données à la base de données. En SQL, cela est généralement réalisé à l'aide de l'instruction INSERT.
- R - Read (Lire) : Il s'agit de la récupération des données existantes à partir de la base de données. En SQL, cela est généralement réalisé à l'aide de l'instruction SELECT.

# SIGNIFICATION DE CRUD

- U - Update (Mettre à jour) : Cette opération consiste à modifier les données existantes dans la base de données. L'instruction SQL utilisée est UPDATE.
- D - Delete (Supprimer) : Il s'agit de la suppression de données de la base de données. L'instruction SQL utilisée est DELETE ou TRUNCATE.

# INSERT

- Insertion des informations dans les tables de la base de données
  - ▶ Ajout des lignes ou tuples.
  - ▶ Ordre et nombre des colonnes dans la clause INTO doivent correspondre aux valeurs.

```
INSERT INTO client (prenom, nom, ville, age)
VALUES
('Rébecca', 'Armand', 'Saint-Didier-des-Bois', 24),
('Aimée', 'Hebert', 'Marigny-le-Châtel', 36),
('Marielle', 'Ribeiro', 'Maillères', 27),
('Hilaire', 'Savary', 'Conie-Molitard', 58);
```

Un tel exemple sur une table vide va créer le tableau suivant :

id	prenom	nom	ville	age
1	Rébecca	Armand	Saint-Didier-des-Bois	24
2	Aimée	Hebert	Marigny-le-Châtel	36
3	Marielle	Ribeiro	Maillères	27
4	Hilaire	Savary	Conie-Molitard	58

# UPDATE

- Pour modifier les enregistrements dans une table de la base de données.
- Attention ne pas oublier la clause **WHERE** sinon la modification sera affectée à tous les enregistrements de colonne.

Syntaxe :

UPDATE nom\_table

SET nom\_colonne1 = ... ,

nom\_colonne2 = ... ,

...

[WHERE condition];

id	nom	rue	ville	code_postal	pays
1	Chantal	12 Avenue du Petit Trianon	Puteaux	92800	France
2	Pierre	18 Rue de l'Allier	Ponthion	51300	France
3	Romain	3 Chemin du Chiron	Trévérien	35190	France

**Modifier une ligne**

```
UPDATE client
SET rue = '49 Rue Ameline',
    ville = 'Saint-Eustache-la-Forêt',
    code_postal = '76210'
WHERE id = 2
```

**Résultats :**

id	nom	rue	ville	code_postal	pays
1	Chantal	12 Avenue du Petit Trianon	Puteaux	92800	France
2	Pierre	49 Rue Ameline	Saint-Eustache-la-Forêt	76210	France
3	Romain	3 Chemin du Chiron	Trévérien	35190	France

# DELETE

- Supprime les enregistrements dans une table de la base de données. Les incréments automatique des compteurs ne sont pas réinitialisés.
- Attention, comme pour UPDATE, si vous oubliez la clause **WHERE**, toutes les données de la table seront supprimées.

Syntaxe :

DELETE FROM nom\_table [WHERE condition];

Table "utilisateur" :

id	nom	prenom	date_inscription
1	Bazin	Daniel	2012-02-13
2	Favre	Constantin	2012-04-03
3	Clerc	Guillaume	2012-04-12
4	Ricard	Rosemonde	2012-06-24
5	Martin	Natalie	2012-07-02

**Supprimer plusieurs lignes**

Si l'on souhaite supprimer les utilisateurs qui se sont inscrits avant le **10/04/2012**, il va falloir effectuer la requête suivante :

```
DELETE FROM `utilisateur`  
WHERE `date_inscription` < '2012-04-10'
```

id	nom	prenom	date_inscription
3	Clerc	Guillaume	2012-04-12
4	Ricard	Rosemonde	2012-06-24
5	Martin	Natalie	2012-07-02

# TRUNCATE

- Pour supprimer l'intégralité du contenu d'une table. Elle ne prend pas de clause WHERE contrairement à DELETE et elle réinitialise les compteurs d'incrémentation automatique.

Syntaxe :

```
TRUNCATE FROM nom_table;
```

# SELECT ... FROM

- Extraire des informations de la base de données (lecture). On doit indiquer la ou les colonnes de la table à afficher, la table est précisée après le mot clé FROM.
- \* : opérateur universel pour sélectionner toutes les colonnes d'une table.
- **DISTINCT** : évite la duplication des lignes ayant les mêmes résultats pour les champs sélectionnés.

SQL Statement:		
<code>SELECT LastName, FirstName, BirthDate FROM Employees;</code>		
Result:		
Number of Records: 10		
LastName	FirstName	BirthDate
Davolio	Nancy	12/8/1968
Fuller	Andrew	2/19/1952
Leverling	Janet	8/30/1963

SQL Statement:					
<code>SELECT * FROM Employees;</code>					
Result:					
Number of Records: 10					
EmployeeID	LastName	FirstName	BirthDate	Photo	Notes
1	Davolio	Nancy	12/8/1968	EmpID1.pic	Education includes a BA in psychology from Colorado State University. She also completed (The Art of the Cold Call). Nancy is a member of 'Toastmasters International'.
2	Fuller	Andrew	2/19/1952	EmpID2.pic	Andrew received his BTS commercial and a Ph.D. in international marketing from the University of



# WHERE

- La clause WHERE permet de filtrer une mise à jour, une suppression ou l'affichage des données. Une ou plusieurs conditions peuvent être séparées par des opérateurs logiques AND ou OR. D'autres opérateurs permettent d'effectuer les conditions.

```
SELECT * FROM client WHERE ville = 'paris'
```

Cette requête retourne le résultat suivant:

id	nom	nbr_commande	ville
1	Paul	3	paris
4	Gérard	7	paris

Opérateur	Description
=	Égale
<>	Pas égale
!=	Pas égale
>	Supérieur à
<	Inférieur à
>=	Supérieur ou égale à
<=	Inférieur ou égale à
IN	Liste de plusieurs valeurs possibles
BETWEEN	Valeur comprise dans un intervalle donnée (utile pour les nombres ou dates)
LIKE	Recherche en spécifiant le début, milieu ou fin d'un mot.
IS NULL	Valeur est nulle
IS NOT NULL	Valeur n'est pas nulle

# SELECT ... AS ...

- Un alias peut être créé pour chaque colonne affichée par un SELECT. C'est le nom de l'alias qui sera affiché à la place du nom de la colonne lors du résultat.

Exemple :

```
SELECT customer_id AS identifiant, total_amount AS montant_total  
FROM orders;
```

Cela affichera un résultat avec les colonnes « identifiant » et « montant\_total ».

# GROUPEZ LES RESULTATS

## GROUP BY

- Regroupe les informations par groupe à partir d'une ou plusieurs colonnes présentes dans la partie du SELECT.
- Ne fonctionne pas si un champ non agrégé n'est pas dans la partie SELECT.

id	client	tarif	date
1	Pierre	102	2012-10-23
2	Simon	47	2012-10-27
3	Marie	18	2012-11-05
4	Marie	20	2012-11-14
5	Pierre	160	2012-12-03

```
SELECT client, SUM(tarif)
FROM achat
GROUP BY client
```

client	SUM(tarif)
Pierre	262
Simon	47
Marie	38

# FILTRE LES RESULTATS GROUPE

## HAVING

- HAVING s'applique sur la clause GROUP BY qui le précède.

Syntaxe :

```
SELECT ... AS ...  
FROM ...  
GROUP BY ...  
HAVING condition_du_group_by;
```

S'il y a un alias à l'aide de AS dans un SELECT, il peut être utilisé dans la clause HAVING.

# LIMIT, OFFSET ET ORDER BY

- **LIMIT** : Limiter le nombre de résultats.
- **OFFSET** : Commencer à partir du résultat X.

La notation de offset varie selon les SGBDR, se référer à leur documentation.

Exemples :

MySQL : `SELECT * FROM nom_table LIMIT 5, 10; -- 5 = offset et 10 = limit`

PostgreSQL : `SELECT * FROM nom_table LIMIT 10 OFFSET 5`

- **ORDER BY ...** : Organiser les résultats selon une colonne.
- **ORDER BY ... ASC** : Ordre croissant (par défaut).
- **ORDER BY ... DESC** : Ordre décroissant.

Exemple : `SELECT * FROM nom_table ORDER BY nom_colonne DESC;`

# ORDRE DES CLAUSES

L'ordre des clauses dans un SELECT est important.  
Elle se fait de la façon suivante :

```
SELECT ... AS ...  
FROM ...  
WHERE condition  
GROUP BY ...  
HAVING condition_du_group_by  
ORDER BY ...  
LIMIT ...  
OFFSET ...;
```

# INDEX

## — Gains

- ▶ Accélérer la recherche
- ▶ Un index permet de créer et de trier la colonne afin de permettre des temps de réponse beaucoup plus rapide.
- ▶ A créer lorsqu'une colonne est fréquemment demandée dans les requêtes et que les données stockées ne sont pas souvent mises à jour.
- ▶ Lors de la création des clés primaires ou étrangères un index implicite est créé.
- ▶ Par défaut, une fois créé, ils sont gérés par le SGBDR? Ils s'en servent si ça permet de gagner du temps de réponse.



# INDEX

## — Inconvénients

- ▶ Espace de stockage supplémentaire dédié à la sauvegarde de tous les index.
- ▶ Ralentissement des requêtes d'insertion lors de mises à jour importantes parce qu'à chaque insertion, les index doivent être mis à jour.
- ▶ Pour garder leurs bénéfices, les index doivent être tenus à jour.



# INDEX et INDEX UNIQUE

## — Index de recherche

```
CREATE INDEX 'nom_index' ON 'nom_table' ('colonne1');
```

```
CREATE INDEX 'nom_index' ON 'nom_table' ('colonne1', 'colonne2');
```

## — INDEX UNIQUE

C'est un index qui garantit l'unicité d'une valeur dans une colonne ou combinaison des colonnes (appelé index composite).

```
CREATE UNIQUE INDEX 'nom_index' ON 'nom_table' ('colonne1');
```

```
CREATE UNIQUE INDEX 'nom_index' ON 'nom_table' ('colonne1', 'colonne2');
```

# INDEX et INDEX UNIQUE

Exemple à la création d'une table

```
CREATE TABLE clients (  
    id INT PRIMARY KEY,  
    nom VARCHAR(50),  
    ville VARCHAR(50),  
    INDEX idx_nom (nom)  
);
```

-- l'index sera utile seulement s'il y a beaucoup de lignes dans la table.

```
SELECT * FROM clients WHERE nom = ... ;
```

# SUPPRESSION DES INDEX

- La syntaxe varie selon les SGBDR
- Les index sont liés à la base de données
- Les nommer avec un nom unique et le préfixe approprié facilite la gestion
- Les préfixes

PK\_name pour Primary Key

FK\_name pour Foreign Key

UK\_name pour Unique Key

IX\_name pour Index

## MS Access:

```
DROP INDEX index_name ON table_name;
```

## SQL Server:

```
DROP INDEX table_name.index_name;
```

## DB2/Oracle:

```
DROP INDEX index_name;
```

## MySQL:

```
ALTER TABLE table_name  
DROP INDEX index_name;
```

# UNION, INTERSEC, EXCEPT

Ces opérateurs agissent sur le résultat de requêtes multiples. UNION est supporté à partir de MySQL v8.0.19 et les deux autres depuis la v8.0.31.

UNION combine tous les résultats des requêtes en éliminant les doublons. Pour que cela fonctionne, il faut que les requêtes donnent le même nombre de colonne avec le même type.

INTERSECT limite le résultat de requêtes multiples aux lignes communes à ces requêtes en éliminant les doublons.

EXCEPT retourne les résultats qui ne sont pas communs aux requêtes multiples en éliminant les doublons.

# UNION, INTERSEC, EXCEPT

Syntaxe :     SELECT nom, prenom FROM table\_a  
                  UNION -- ou INTERSECT ou EXCEPT  
                  SELECT nom, prenom FROM table\_b;

## 5. OPERATIONS CONDITIONNELLES



# OPERATIONS CONDITIONNELLES

Pour effectuer des opérations conditionnelles il y a les expressions CASE WHEN. Elle peut s'appliquer sur un SELECT, UPDATE, DELETE ou INSERT.

## — Syntaxe générale

SELECT

CASE

WHEN condition1 THEN resultat1

WHEN condition2 THEN resultat2

...

ELSE resultat\_par\_defaut

END

AS un\_alias -- optionnel pour donner un nom à la colonne resultante.

FROM table;

# OPERATIONS CONDITIONNELLES

## — Exemple

```
SELECT order_id, product_name, quantity,  
       CASE  
           WHEN quantity > 10 AND quantity <= 20 THEN 'Quantité moyenne'  
           WHEN quantity > 20 THEN 'Quantité élevée'  
           ELSE 'Quantité faible'  
       END  
AS quantity_category  
FROM orders;
```





**ib  
cegos**

## **6. LES FONCTIONS**



# LES FONCTIONS STRING

- UPPER(X) : X en majuscule.
- LENGTH(X) : nombre de caractère qui constitue X (taille).
- LOWER(X) : X en minuscule.
- TRIM() : Supprime des caractères au début et en fin d'une chaîne de caractères.
- FORMAT() : Formate une donnée selon un certain motif.

# LES FONCTIONS STRING

Attention, chaque SGBDR propose un panel de fonction, se référer à la documentation de chacun.

Exemple :

```
SELECT id, UPPER(prenom) AS prenom_upper FROM Utilisateur;
```

# LES FONCTIONS MATHEMATIQUE ET DATES

## — Dates

- ▶ `DATE()` : Calcul une date à partir des arguments qu'elle reçoit.
- ▶ `TIME()` : Calcul le temps/arguments.
- ▶ `DATETIME()` : Les fonctions liées aux dates dépendent des SGBDR.

# LES FONCTIONS MATHEMATIQUE ET DATES

## – Mathématiques

- ▶ `ABS()` : Valeur absolue.
- ▶ `ROUND()` : Arrondie.
- ▶ `SQRT()` : Racine carrée.
- ▶ `POWER(X, Y)` : Puissance.

Les fonctions mathématiques sont spécifiques à chaque SGBDR, toujours se référer à la documentation du SGBDR.

# FONCTIONS D'AGREGATION DES DONNEES

- AVG(field) : Calcule la moyenne des valeurs de toutes les occurrences.
- COUNT(field) : Compte toutes les occurrences d'un élément. Avec le paramètre universel \*, il compte tout.
- GROUP\_CONCAT(field): Concaténation de toutes les valeurs.

Séparées par un ou des caractères spécifiés via SEPARATOR 'caractères' à la suite de « field ». Il est aussi possible d'utiliser DISTINCT, ORDER BY et LIMIT.

# FONCTIONS D'AGREGATION DES DONNEES

- MAX(field) : Valeur maximale de toutes les occurrences.
- MIN(field) : Valeur minimale de toutes les occurrences.
- SUM(field) : Somme de toutes les occurrences.

# LA CLAUSE OVER

Cela permet de calculer des valeurs agrégées avec des fonctions telle que SUM ou AVG basées sur un sous-ensemble de lignes plutôt que sur l'ensemble total du résultat de la requête. Les sous-ensembles se font à partir d'une colonne sélectionnée et peuvent être ordonnés par un ORDER BY.

Syntaxe :

```
SELECT colonne1, colonne2, ..., colonneN, SUM(colonne3)
```

```
OVER (PARTITION BY uneColonne ORDER BY colonne3 DESC) AS  
aliasColonne3;
```





**ib  
cegos**

## **7. LES JOINTURES**



# QU'EST-CE QU'UNE JOINTURE ?

Cela permet :

- D'associer les tables dans une requête SELECT.
- De combiner les données plusieurs tables dans le résultat de la requête.

# QU'EST-CE QU'UNE JOINTURE ?

Il existe plusieurs type de jointures :

- INNER JOIN : Intersection entre 2 tables.
- LEFT OUTER JOIN : Récupération totale des informations de la table de gauche.
- RIGHT OUTER JOIN : Récupération totale des informations de la table de droite.
- FULL OUTER JOIN : Récupération des informations de chaque table.
- ...

# QU'EST-CE QU'UNE JOINTURE ?

## — Exemple :

```
SELECT Orders.OrderID, Customers.CustomerName  
FROM Orders  
INNER JOIN Customers ON Orders.CustomerID = Customers.CustomerID;
```

# QU'EST-CE QU'UNE JOINTURE ?

## — Remarques :

- ▶ En MySQL les mots clés INNER et OUTER sont optionnels. On peut donc écrire JOIN au lieu de INNER JOIN. RIGHT JOIN au lieu de RIGHT OUTER JOIN ...
- ▶ Les OUTER retournent des valeurs NULL s'il n'y a pas de correspondance entre les tables.
- ▶ Le JOIN (INNER) est la jointure la plus utilisée.

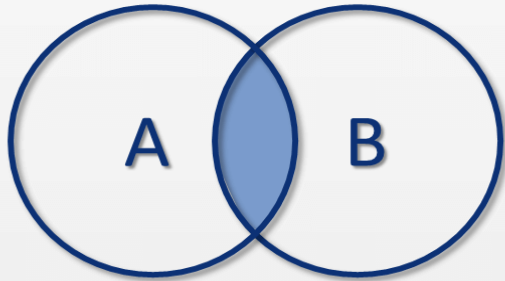
# QU'EST-CE QU'UNE JOINTURE ?

- En plus des 3 OUTER et du INNER, Il existe deux autres jointures moins souvent utilisées :

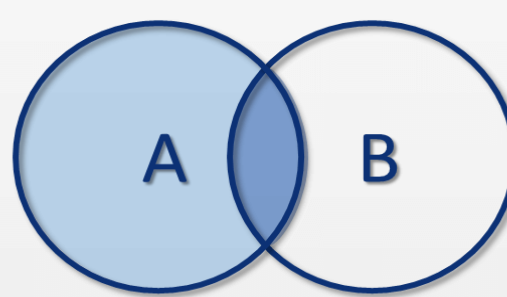
- ▶ CROSS JOIN : C'est la jointure croisée. Retourne le produit cartésien des deux tables. C'est-à-dire toutes les combinaisons possible des lignes.
- ▶ SELF JOIN : C'est l'auto-jointure. Elle permet la jointure d'une table à elle-même. Utile lorsque vous devez comparer des lignes au sein de la même table.

# JOINTURES

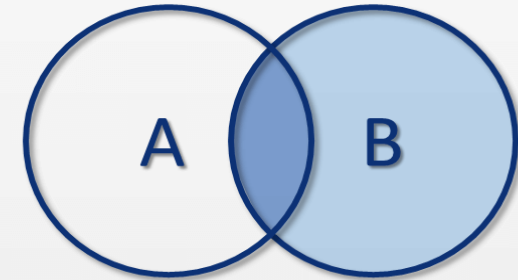
## INNER JOIN



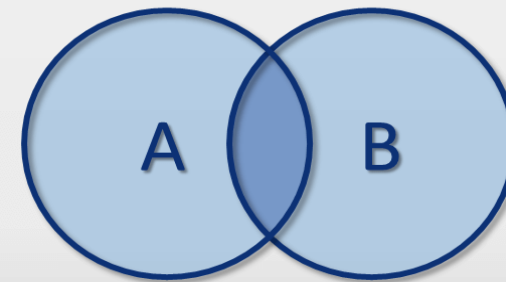
## OUTER JOIN



LEFT



RIGHT



FULL

# JOINTURES





ib  
cegos

## 8. TRANSACTION



# TRANSACTION

Les transactions d'une BDDR se base sur le troisième sous-langage de SQL.

- Cela garantie la cohérence d'une base de données lors de l'exécution de plusieurs requêtes.
- Tout ou rien. Les requêtes entre le début et la fin d'une transaction doivent toutes s'exécuter avec succès sinon tout est annulé (la base de données va défaire ce qui s'est effectué avec succès).

# TRANSACTION

## — Exemple :

START TRANSACTION;

SELECT \* FROM nom\_table WHERE (condition);

UPDATE nom\_table2 SET nom\_colonne = valeur WHERE (condition);

COMMIT; -- valide la transaction

ROLLBACK; -- annule la dernière transaction.

ib  
cegos

## 9. TRIGGER



# TRIGGER

- Un Trigger (déclencheur) renferme une requête SQL qui se déclenche lors d'un évènement survenue sur une table.
- 3 types d'évènements trigger :
  - ▶ Insertion
  - ▶ Modification
  - ▶ Suppression
- Il faut spécifier dans le Trigger quand est-ce que la requête SQL présente doit s'exécuter. Elle s'exécute soit avant soit après l'évènement.

# TRIGGER

## — Règle :

- ▶ Un seul trigger par table, par évènement et par moment.
- ▶ Attention, les triggers ont un impact significatif au niveau des performances de la base de données, à éviter au maximum.

# TRIGGER

## — Syntaxe

```
DELIMITER //  
CREATE TRIGGER nom_declencheur  
{BEFORE | AFTER} {INSERT | UPDATE | DELETE}  
ON nom_table  
FOR EACH ROW  
BEGIN  
    -- Instructions SQL à exécuter ;  
END;  
//  
DELIMITER ;
```

**DELIMITER** permet de changer le caractère qui indique la fin d'une requête SQL.

# TRIGGER

**BEFORE** : s'exécute avant INSERT, UPDATE ou DELETE.

**AFTER** : s'exécute après INSERT, UPDATE ou DELETE.

**NEW** fait référence aux nouvelles valeurs de la ligne (avant l'insertion ou la mise à jour).

**OLD** fait référence aux anciennes valeurs de la ligne (avant la mise à jour ou la suppression).

Pour le INSERT, seul NEW existe.

Pour le UPDATE, NEW et OLD existent.

Pour DELETE, seul OLD existe.



## 10. TERMINAL MySQL ET GESTION DES PRIVILÈGES



# INSTALLATION MYSQL

## [Installation MySQL Community Downloads](#)

Installez **uniquement** MySQL Server 8.0.3x et MySQL Shell 8.0.3x.  
Télécharger la version la plus légère.


### MySQL Installer 8.0.32

Select Operating System:

Microsoft Windows

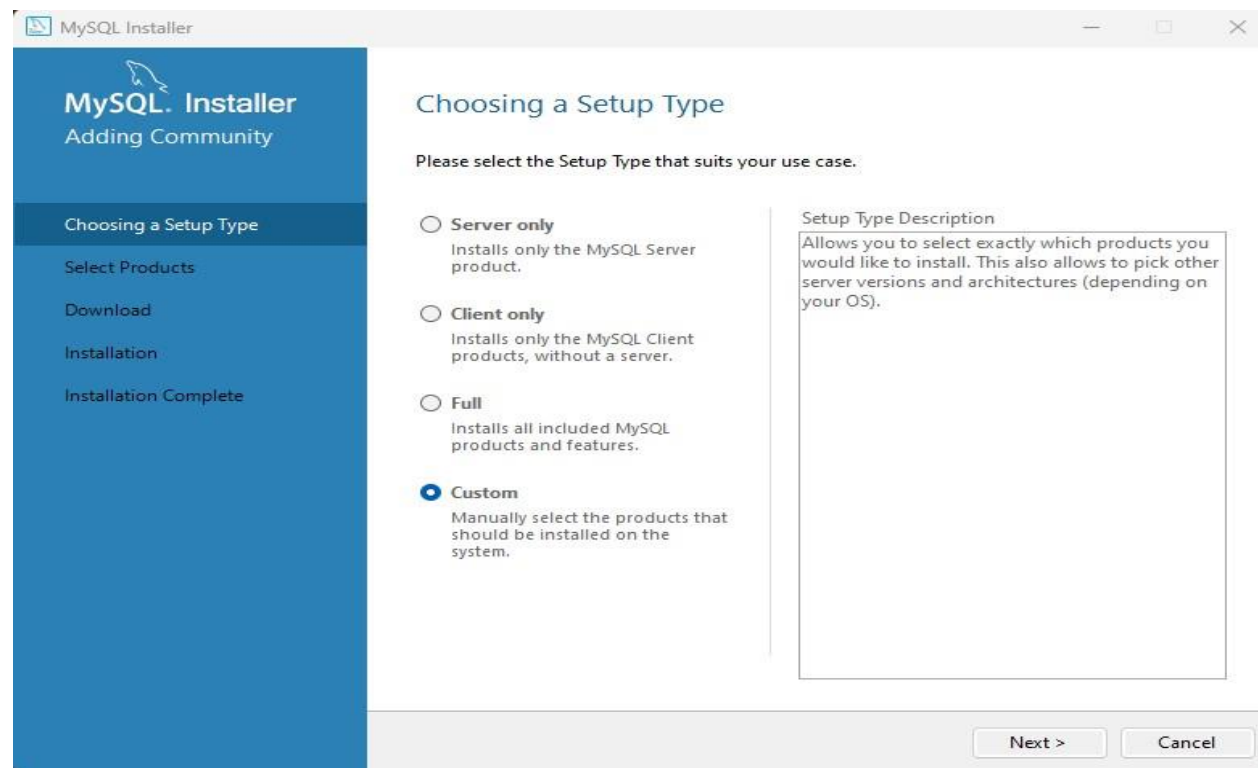
Looking for previous GA versions?

<b>Windows (x86, 32-bit), MSI Installer</b> (mysql-installer-web-community-8.0.32.0.msi)	8.0.32	2.4M	<a href="#">Download</a>
		MD5: 0f882590f8338adc614e9dc5cb00ca0b   <a href="#">Signature</a>	
<b>Windows (x86, 32-bit), MSI Installer</b> (mysql-installer-community-8.0.32.0.msi)	8.0.32	437.3M	<a href="#">Download</a>
		MD5: a29b5817cba2c7bc0e0b97e897c2591f   <a href="#">Signature</a>	

 We suggest that you use the [MD5 checksums](#) and [GnuPG signatures](#) to verify the integrity of the packages you download.

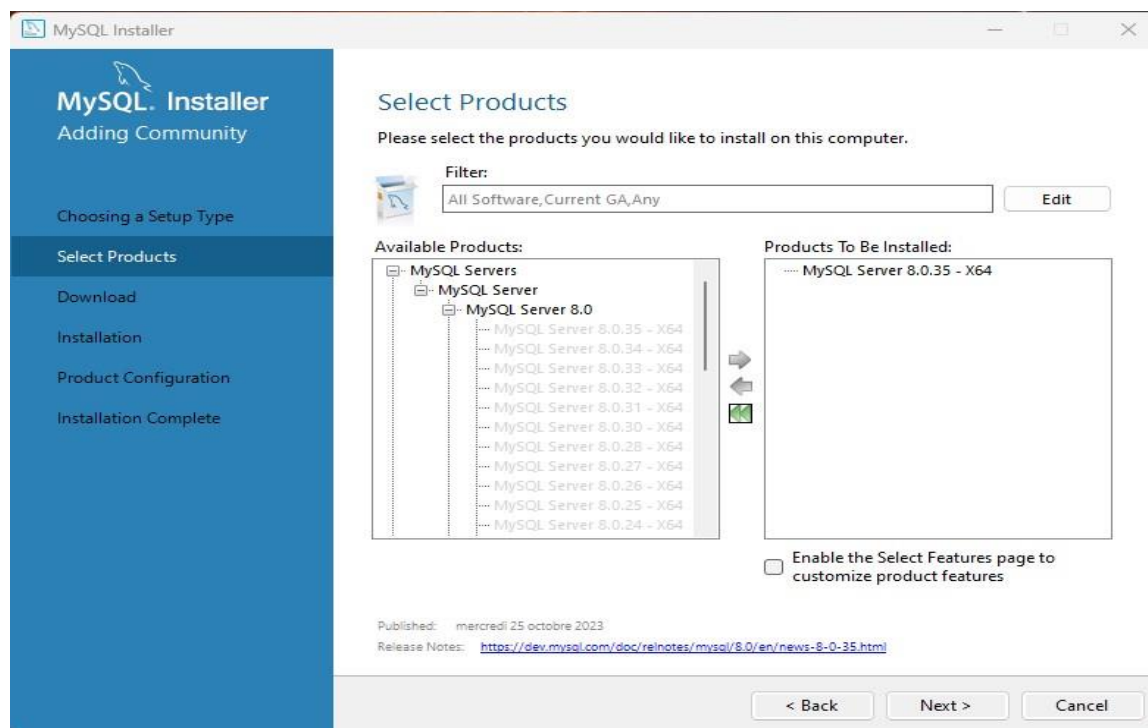
# INSTALLATION MYSQL

Exécuter le programme d'installation et choisir « Custom » puis cliquer sur « next ».



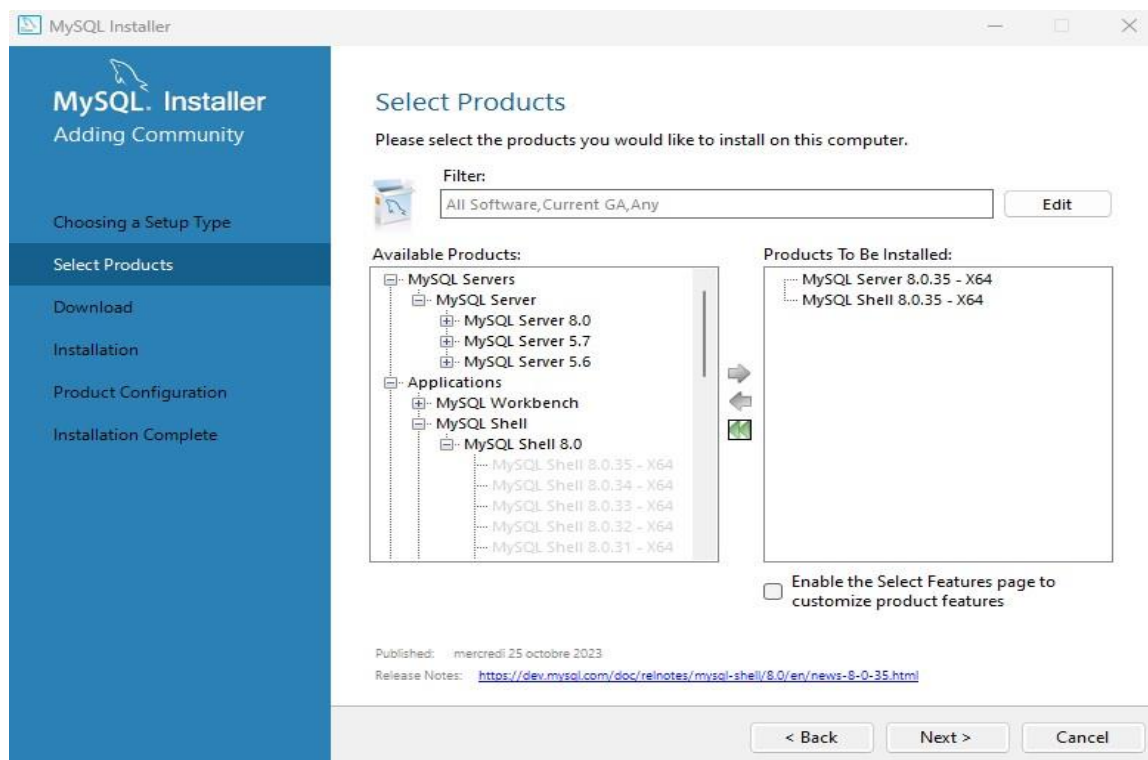
# INSTALLATION MYSQL

Choisir la dernière version MySQL Server 8.0.x et cliquer sur la flèche verte vers la droite pour valider la sélection.



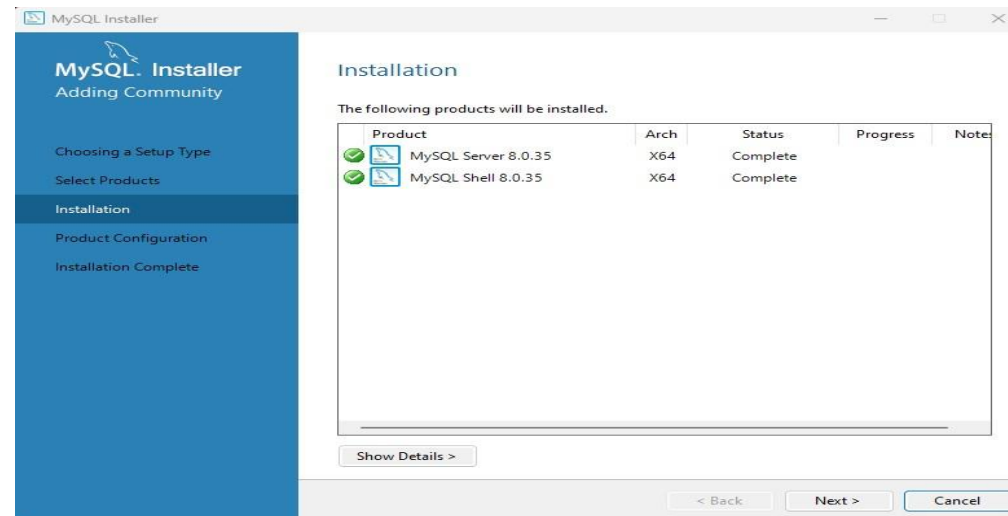
# INSTALLATION MYSQL

Choisir la dernière version MySQL Shell 8.0.x et cliquer sur la flèche verte vers la droite pour valider la sélection. Puis cliquer sur le bouton « next ».



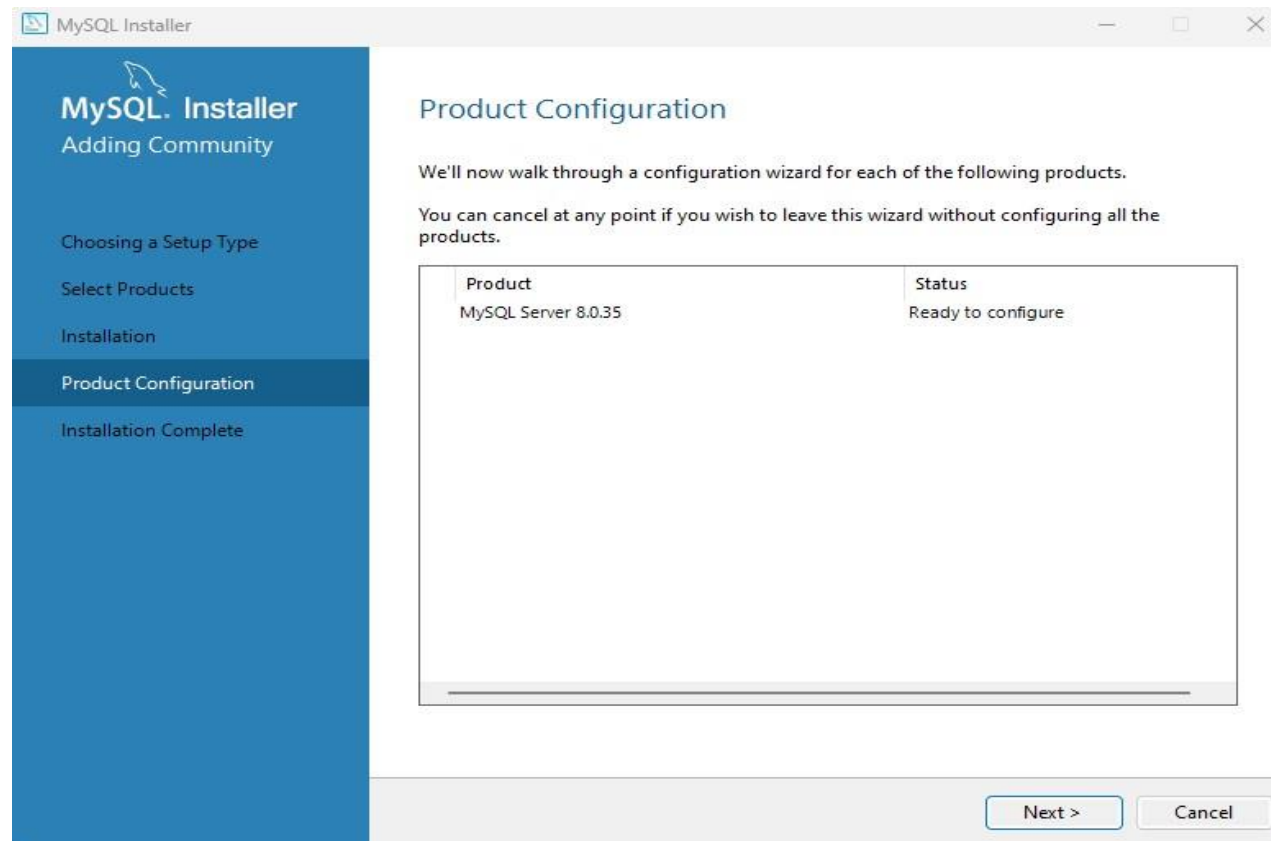
# INSTALLATION MYSQL

Si une ou toutes les parties sélectionnées apparaissent c'est qu'elle sont pas encore installée sur le PC. Dans ce cas, cliquer sur le bouton « Execute » pour lancer le téléchargement puis, l'installation. Si rien n'apparaît c'est sûrement que les parties sélectionnées sont déjà installées sur le PC. Une fois fini, cliquer sur « next ».



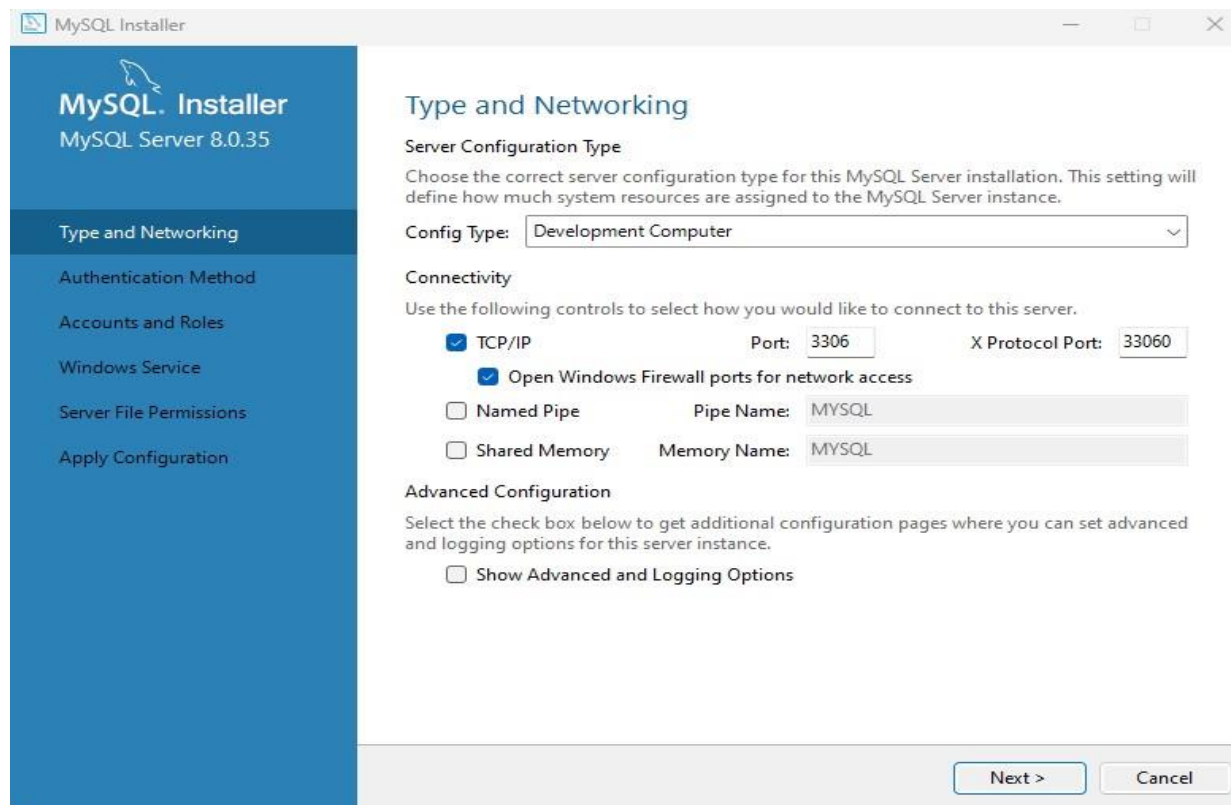
# INSTALLATION MYSQL

Cliquer sur « next ».



# INSTALLATION MYSQL

Laisser la configuration par défaut et finir l'installation et cliquer sur « next ».



The screenshot shows the MySQL Installer window for MySQL Server 8.0.35. The left sidebar contains the following navigation items: MySQL. Installer, MySQL Server 8.0.35, Type and Networking (selected), Authentication Method, Accounts and Roles, Windows Service, Server File Permissions, and Apply Configuration. The main content area is titled 'Type and Networking' and includes the following sections:

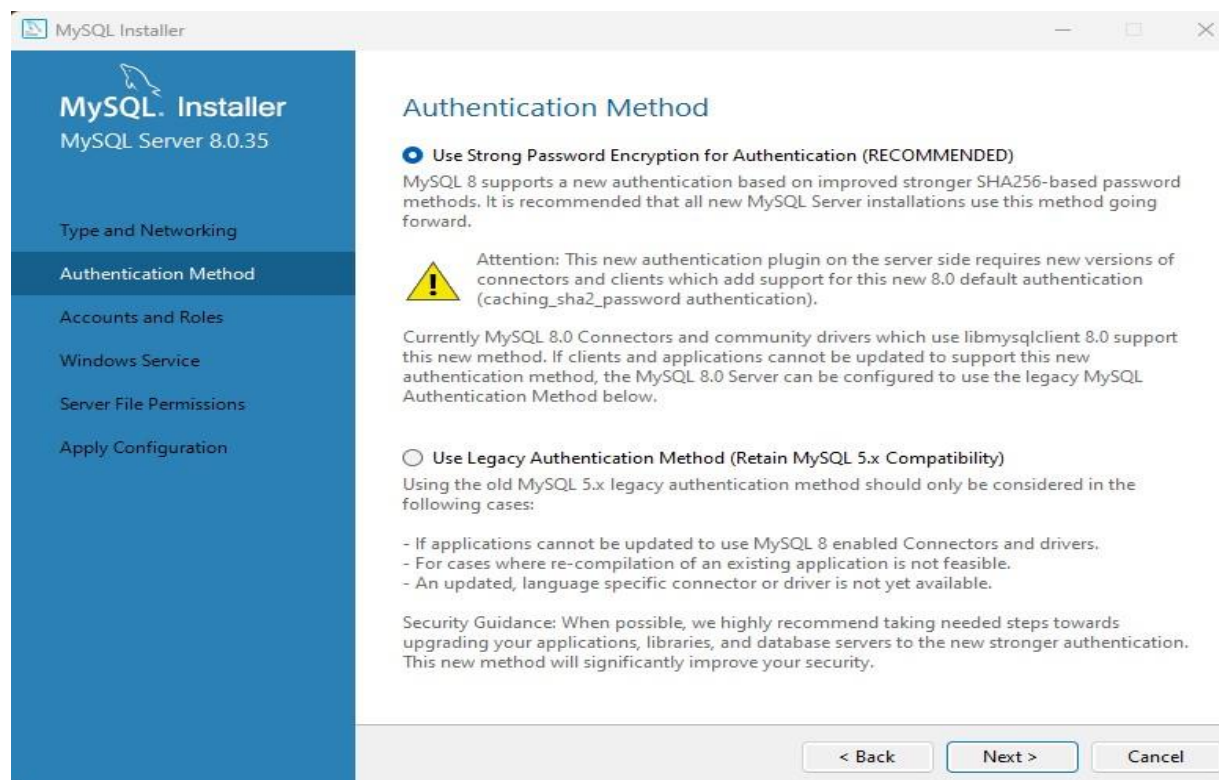
- Server Configuration Type**: Choose the correct server configuration type for this MySQL Server installation. This setting will define how much system resources are assigned to the MySQL Server instance. Config Type: Development Computer (dropdown menu).
- Connectivity**: Use the following controls to select how you would like to connect to this server.
  - ☒ TCP/IP: Port: 3306, X Protocol Port: 33060.
  - ☒ Open Windows Firewall ports for network access.
  - ☐ Named Pipe: Pipe Name: MYSQL.
  - ☐ Shared Memory: Memory Name: MYSQL.
- Advanced Configuration**: Select the check box below to get additional configuration pages where you can set advanced and logging options for this server instance.
  - ☐ Show Advanced and Logging Options.

At the bottom right, there are 'Next >' and 'Cancel' buttons.



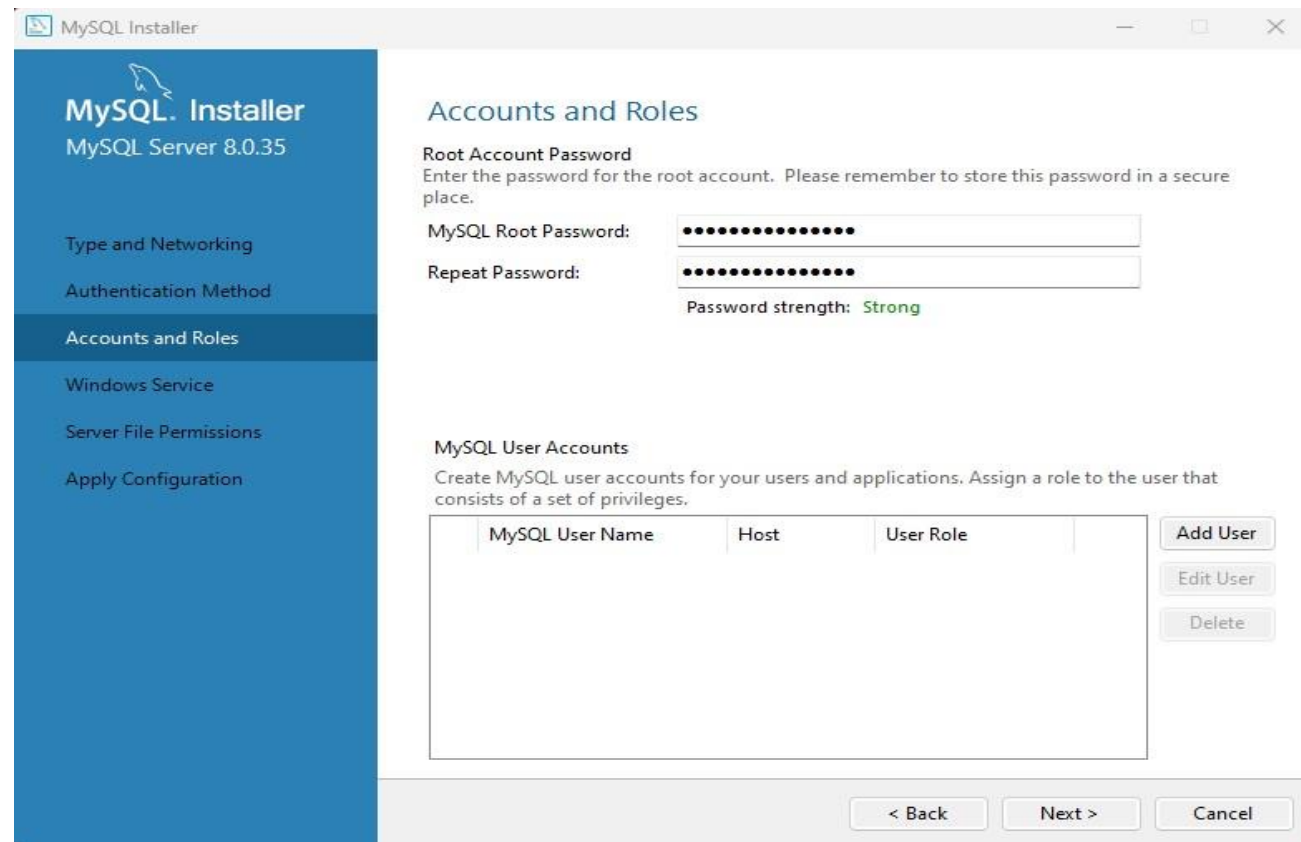
# INSTALLATION MYSQL

Laisser le choix d'un mot de passe fort par défaut et cliquer sur « next ».



# INSTALLATION MYSQL

Choisir un mot de passe fort, confirmer et cliquer sur « next ».



The screenshot shows the 'MySQL Installer' window for 'MySQL Server 8.0.35'. The left sidebar lists the installation steps: 'Type and Networking', 'Authentication Method', 'Accounts and Roles' (which is the current step), 'Windows Service', 'Server File Permissions', and 'Apply Configuration'. The main area is titled 'Accounts and Roles' and contains two sections. The first section, 'Root Account Password', prompts the user to enter a password for the root account, with a note to store it securely. It includes two password input fields (one for the password and one for repeating it) and a 'Password strength' indicator showing 'Strong' in green. The second section, 'MySQL User Accounts', prompts the user to create MySQL user accounts and assign roles. It features a table with columns for 'MySQL User Name', 'Host', and 'User Role'. To the right of the table are three buttons: 'Add User', 'Edit User', and 'Delete'. At the bottom of the window are three buttons: '< Back', 'Next >', and 'Cancel'.

MySQL Installer  
MySQL Server 8.0.35

Type and Networking  
Authentication Method  
**Accounts and Roles**  
Windows Service  
Server File Permissions  
Apply Configuration

### Accounts and Roles

**Root Account Password**  
Enter the password for the root account. Please remember to store this password in a secure place.

MySQL Root Password:

Repeat Password:

Password strength: **Strong**

**MySQL User Accounts**  
Create MySQL user accounts for your users and applications. Assign a role to the user that consists of a set of privileges.

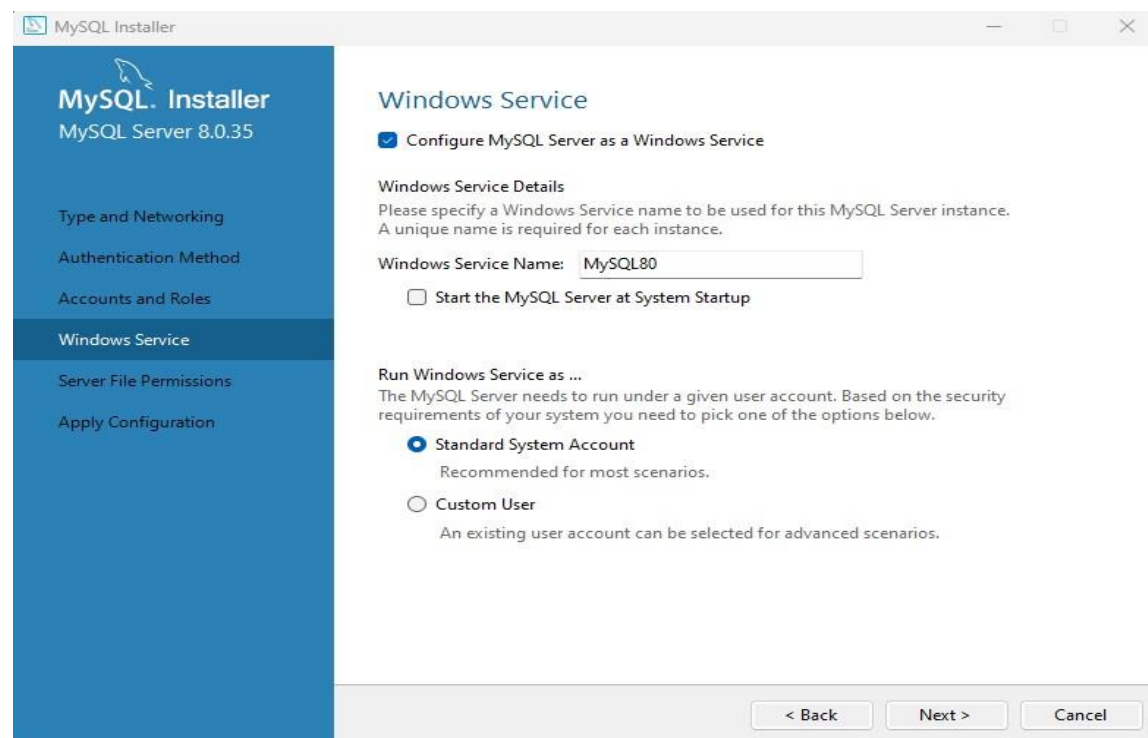
MySQL User Name	Host	User Role
-----------------	------	-----------

Add User  
Edit User  
Delete

< Back   Next >   Cancel

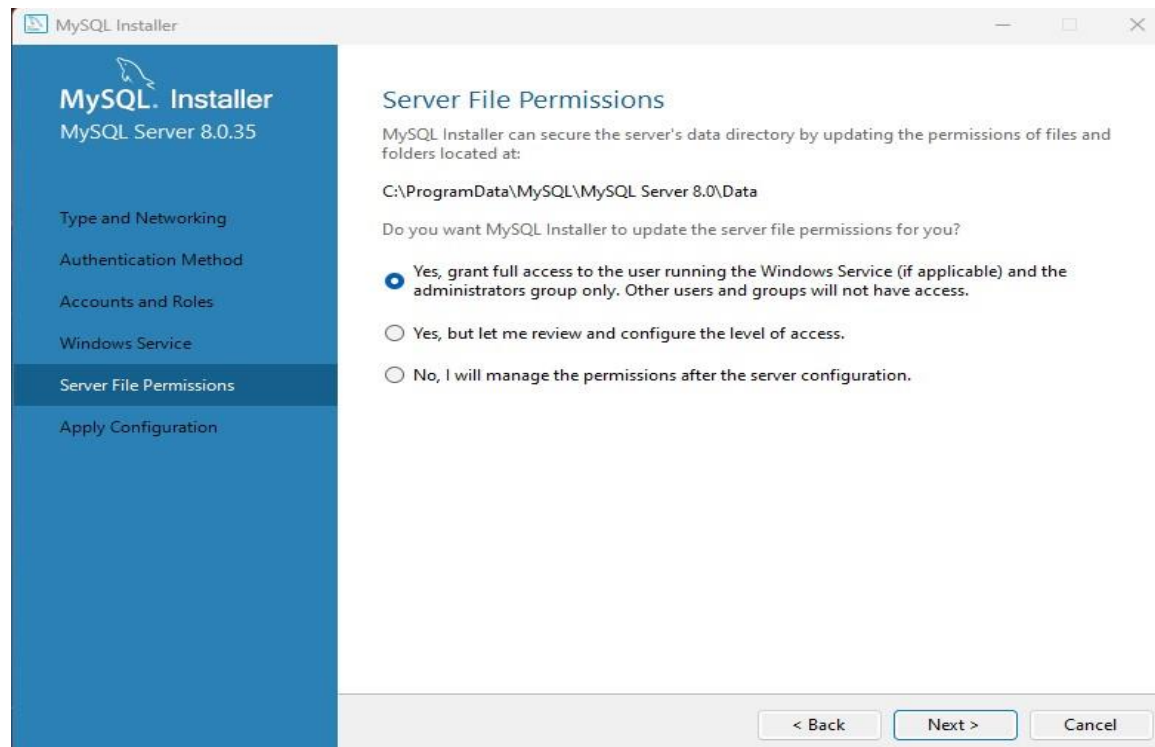
# INSTALLATION MYSQL

Laisser par défaut ou décocher la case de démarrage automatique de MySQL Server lors du démarrage de Windows et cliquer sur « next ».



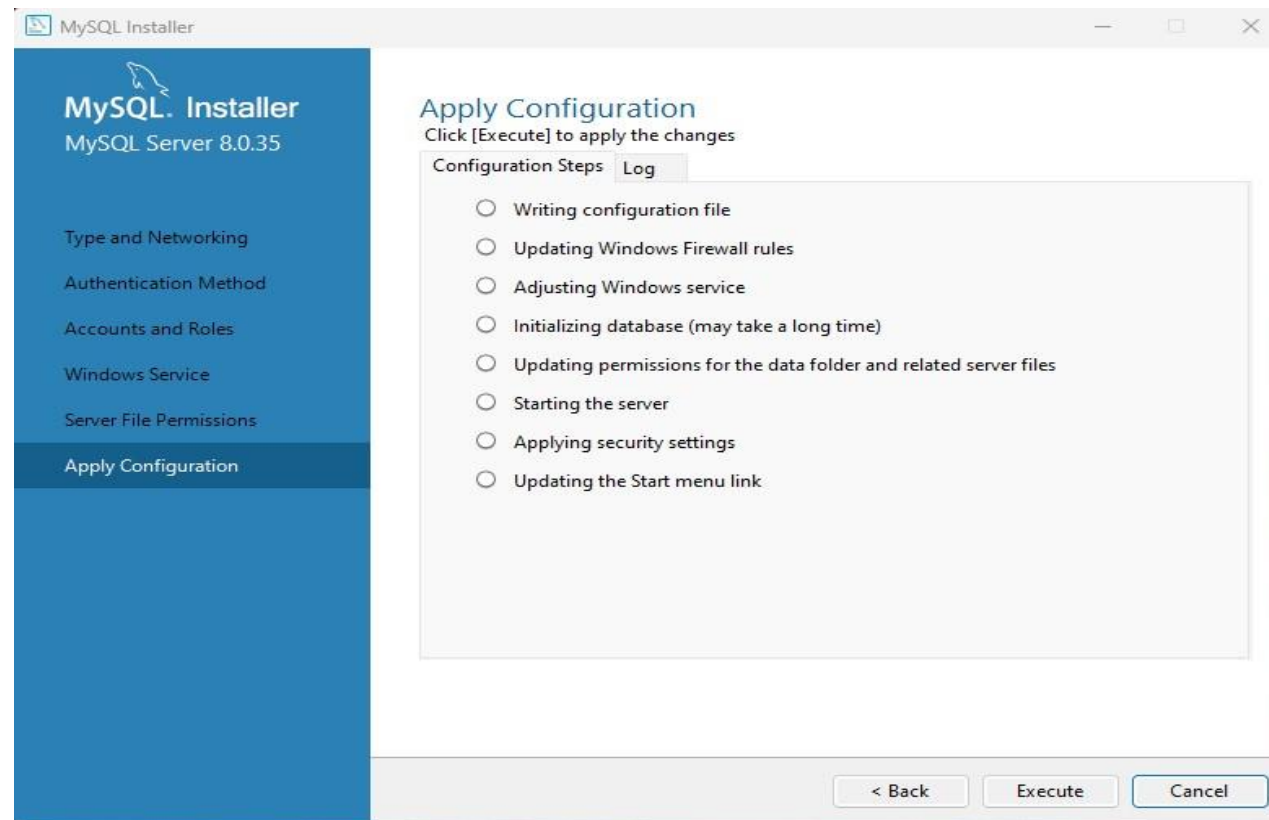
# INSTALLATION MYSQL

Laisser par défaut pour que l'utilisateur root puisse avoir tout les droits et cliquer sur « next ».



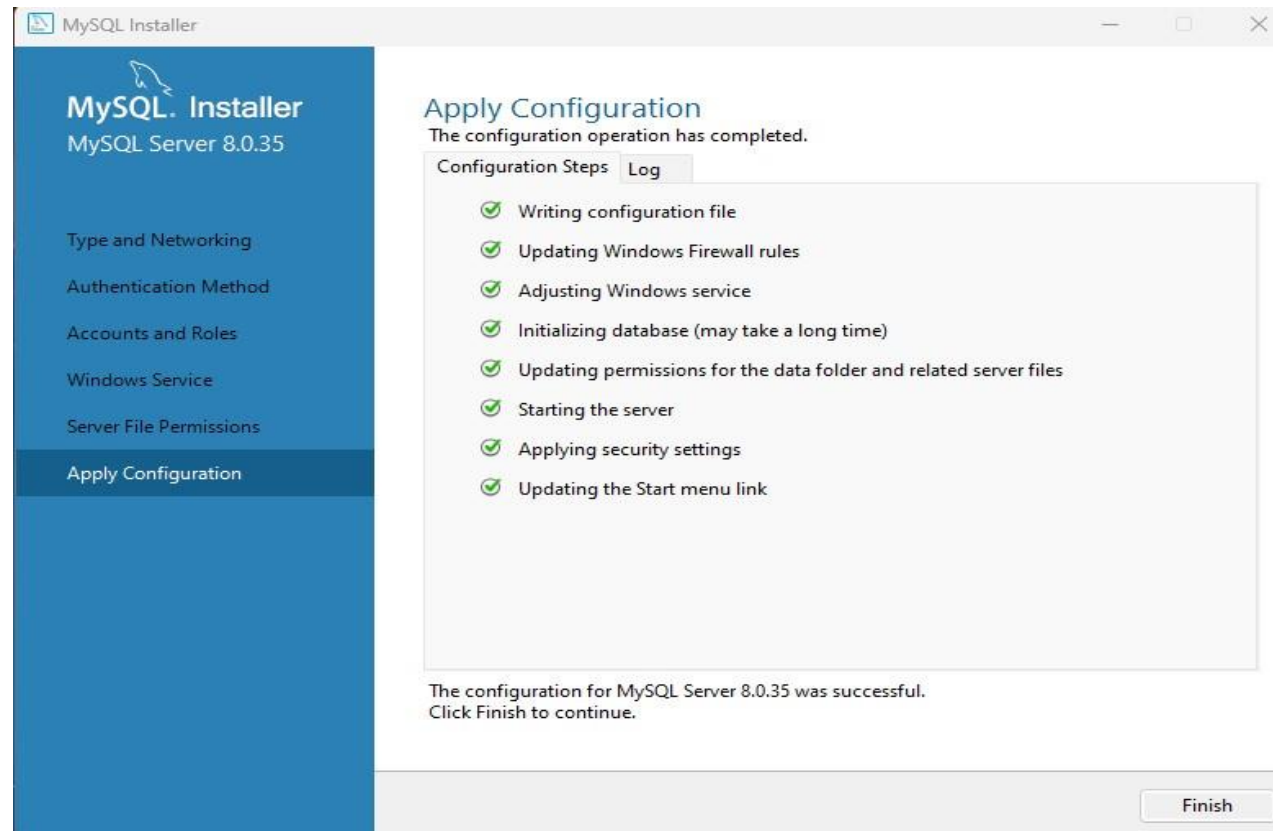
# INSTALLATION MYSQL

Cliquer sur « Execute » pour appliquer la configuration.



# INSTALLATION MYSQL

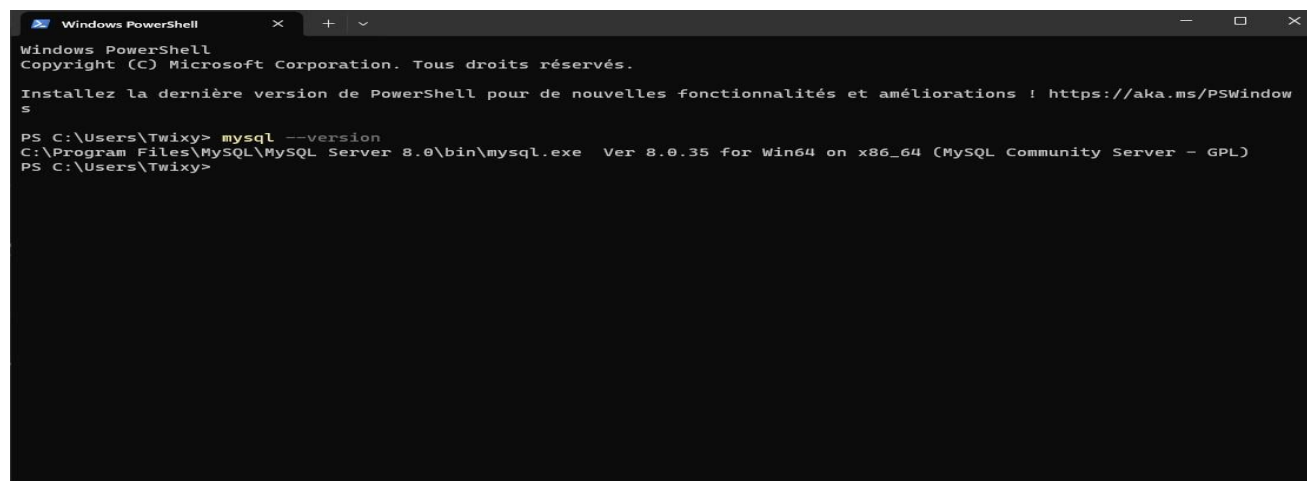
Cliquer sur « Finish » pour finir la configuration.



# INSTALLATION MYSQL

Modifier la variable d'environnement « C:\Program Files\MySQL\MySQL Shell 8.0\bin » en « C:\Program Files\MySQL\MySQL Server 8.0\bin » puis valider.

Lancer powershell et taper « `mysql --version` » cela devrait afficher la version de mysql.



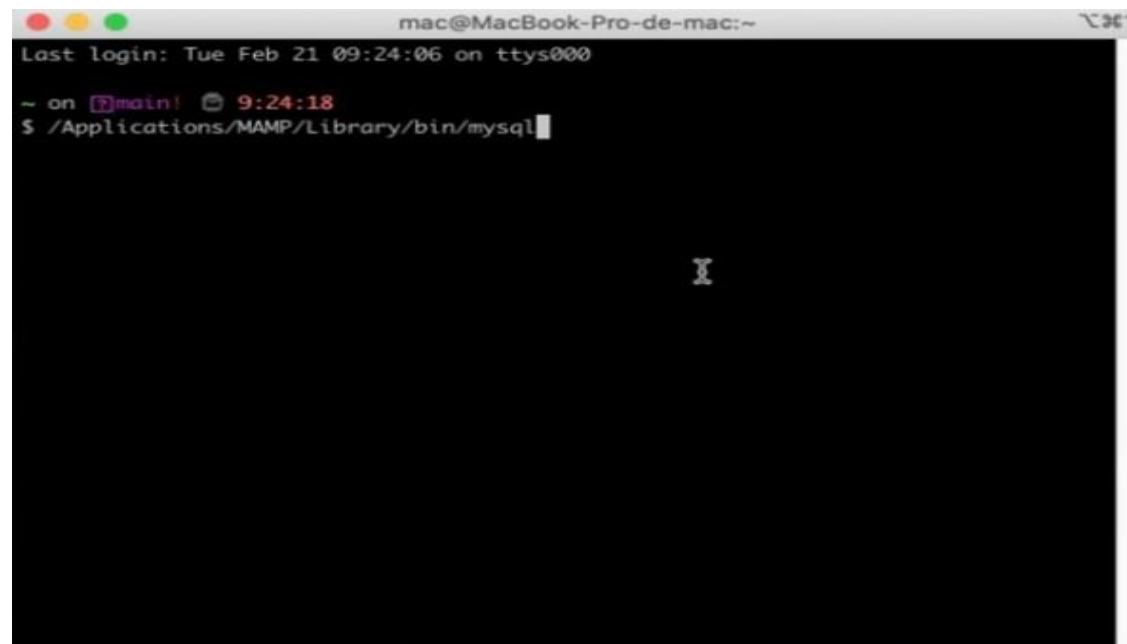
```
Windows PowerShell
Copyright (C) Microsoft Corporation. Tous droits réservés.

Installez la dernière version de PowerShell pour de nouvelles fonctionnalités et améliorations ! https://aka.ms/PSWindows

PS C:\Users\Twixy> mysql --version
C:\Program Files\MySQL\MySQL Server 8.0\bin\mysql.exe Ver 8.0.35 for Win64 on x86_64 (MySQL Community Server - GPL)
PS C:\Users\Twixy>
```

# INSTALLATION MYSQL

Pour mac, à partir du terminal, lancer les commandes MySQL en utilisant le chemin complet « Applications/MAMP/Library/bin/mysql » ou « Applications/XAMPP/xamppfiles/bin/mysql ».

A screenshot of a macOS terminal window. The title bar at the top reads "mac@MacBook-Pro-de-mac:~". The terminal shows the following text: "Last login: Tue Feb 21 09:24:06 on ttys000", followed by a prompt "~ on [?]main: 9:24:18", and then the command "\$ /Applications/MAMP/Library/bin/mysql" being entered. The cursor is at the end of the command line.

```
mac@MacBook-Pro-de-mac:~  
Last login: Tue Feb 21 09:24:06 on ttys000  
~ on [?]main: 9:24:18  
$ /Applications/MAMP/Library/bin/mysql
```



# DEMARRER MYSQL

Si l'option de démarrage automatique avec Windows n'avais pas été coché, il faut démarrer MySQL quand on en a besoin.

Pour ce faire, aller dans les services Windows et démarrer le service MySQL80.

Pour l'éteindre faire la même opération en choisissant d'arrêter le service.

# UTILISATION MYSQL DANS UN TERMINAL

- Connexion à MySQL depuis un terminal

Exécuter la commande suivante pour se connecter avec l'utilisateur root :

```
mysql -u root -p
```

Entrer le mot de passe + appuyer sur le bouton Entrer

Le prompt `mysql>` apparaît. Vous pouvez exécuter toutes les requêtes SQL depuis ce terminal. Ne pas oublier de mettre un « ; » à chaque fin de requête.

# UTILISATION MYSQL DANS UN TERMINAL

- Exemple :

`show databases;` -- affiche toutes les bases dans MySQL

`show tables;` -- affiche les colonnes des tables de la bdd sélectionnée via `use`.

- Pour quitter l'interface de MySQL

`quit`

# UTILISATION MYSQL DANS UN TERMINAL

- Créer un nouvel utilisateur 'toto' avec mot de passe '123456'.

```
CREATE USER 'toto'@'localhost' IDENTIFIED BY '123456';
```

```
CREATE USER 'toto'@'localhost' IDENTIFIED BY SELECT MD5('123456');
```

Pour le cryptage MD5 il faut avoir installer MySQL avec le module correspondant au MD5. Il existe d'autre cryptage installable tel que SHA2.

# UTILISATION MYSQL DANS UN TERMINAL

- Voir les utilisateurs existant

`SELECT user, host FROM mysql.user; -- permet de voir les utilisateurs existant.`

Parfois on peut avoir besoin d'entourer le nom de la table de backtick.  
Lorsque le nom de la table est un nombre par exemple.

Code du backtick : `AltGr + 7 + espace`

# UTILISATION MYSQL DANS UN TERMINAL

- Modifier le mot de passe d'un utilisateur

```
SET PASSWORD FOR 'user'@'localhost' = 'new password';
```

- Modifier le nom d'utilisateur

```
RENAME USER 'user'@'localhost' TO 'new_user'@'localhost';
```

- Supprimer un utilisateur

```
DROP USER 'user'@'localhost';
```

Si on quitte et qu'on se reconnecte avec le nouvel utilisateur, on s'aperçoit que lorsqu'on essaye d'exécuter un simple `SELECT` cela n'est pas autorisé. C'est normal car l'utilisateur nouvellement créé a aucun droit.

# PRIVILEGES UTILISATEUR

Privilèges global ou sur une base de données	Signification
ALL PRIVILEGES	Total accès
CREATE	Droits de créer des tables et des bases de données
DROP	Droits de supprimer des tables et des bases de données
DELETE	Droits de supprimer des tuples (enregistrements) des tables
INSERT	Droits d'insérer des tuples dans les tables
SELECT	Droits d'extraire les données des tables
UPDATE	Droits de mettre à jour des tuples des tables
GRANT OPTION	Droits d'accorder ou révoquer des droits à d'autres utilisateurs

# ATTRIBUTION DES PRIVILEGES

L'attribution des privilèges constitue le quatrième sous-langage de SQL.

- Attribuer des droits spécifiques à un utilisateur sur une base de données.

```
GRANT INSERT, SELECT, UPDATE, DELETE ON database_name.* TO 'user'@'localhost';
```

Le « . \* » est pour donner les droits sur tout (les tables, les vues, les triggers, ...).



# ATTRIBUTION DES PRIVILEGES

- Attribuer tous les droits à un utilisateur sur toutes les BDD.

```
GRANT ALL PRIVILEGES ON *.* TO 'user'@'localhost' WITH GRANT OPTION;
```

- Pour confirmer la prise en compte des nouveaux droits.

```
FLUSH PRIVILEGES;
```

# VOIR, REVOQUER DES DROITS UTILISATEUR

- Voir les droits d'un utilisateur.

```
SHOW GRANTS FOR 'user'@'localhost';
```

- Révoquer les droits d'un utilisateur.

```
REVOKE ALL PRIVILEGES, GRANT OPTION FROM 'user'@'localhost';
```

```
REVOKE INSERT ON database.* FROM 'user'@'localhost';
```

# IMPORTER UNE BDD SQL

Se connecter au terminal MySQL via un utilisateur ayant les droits.  
Créer la BDD vide et utiliser la BDD.

```
source chemin_complet/nom_fichier.sql;
```

Attention, sous Windows il faut :

Remplacer les « \ » par des « / ».

Ne pas avoir d'espace dans les noms de dossier et fichier.



ib  
cegos

**FIN**

