

Secteur : **Digital & IA**

Manuel de cours

M106 : Manipuler des bases de données

1^{ère} Année

Filière :

Développement
Digital
(Tronc commun)



Remerciements

La DRIF remercie les personnes qui ont contribué à l'élaboration du présent document :

Équipe de conception :

BOUDIAF Saida *Digital learning manager/
Project manager*

MIHOUBI Fattoum, *Cheffe de projet pédagogique/
Ingénierie pédagogique*

CROUZOULON Jonathan, *Directeur pédagogique/
Chef de projet pédagogique*

Équipe de rédaction :

BOUKHAYMA Khaoula, *Data engineer (Consultante Data)*

Équipe de lecture :

LAOUIJA Soukaina, *Formatrice Animatrice au CDC Digital & IA*

EL KHATABI GHIZLANE, *Formatrice Animatrice au CDC Digital & IA*

Équipe de validation :

LAOUIJA Soukaina, *Formatrice Animatrice au CDC Digital & IA*

EL KHATABI GHIZLANE, *Formatrice Animatrice au CDC Digital & IA*

Les utilisateurs de ce document sont invités à communiquer à la DRIF et au CDC Digital & IA toutes les remarques et suggestions afin de les prendre en considération pour l'enrichissement et l'amélioration de ce module.

SOMMAIRE



1. Concevoir une base de données

- Analyser le cahier de charges
- Modéliser les données
- Normaliser les données

2. Préparer l'environnement

- Exploiter un outil de modélisation
- Préparer le serveur MySQL

3. Manipuler les données

- Créer une base de données
- Réaliser des requêtes SQL
- Administrer une base de données

MODALITÉS PÉDAGOGIQUES



1

LE GUIDE DE SOUTIEN

Il contient le résumé théorique et le manuel des travaux pratiques



2

LA VERSION PDF

Une version PDF est mise en ligne sur l'espace apprenant et formateur de la plateforme WebForce Life



3

DES CONTENUS TÉLÉCHARGEABLES

Les fiches de résumés ou des exercices sont téléchargeables sur WebForce Life



4

DU CONTENU INTERACTIF

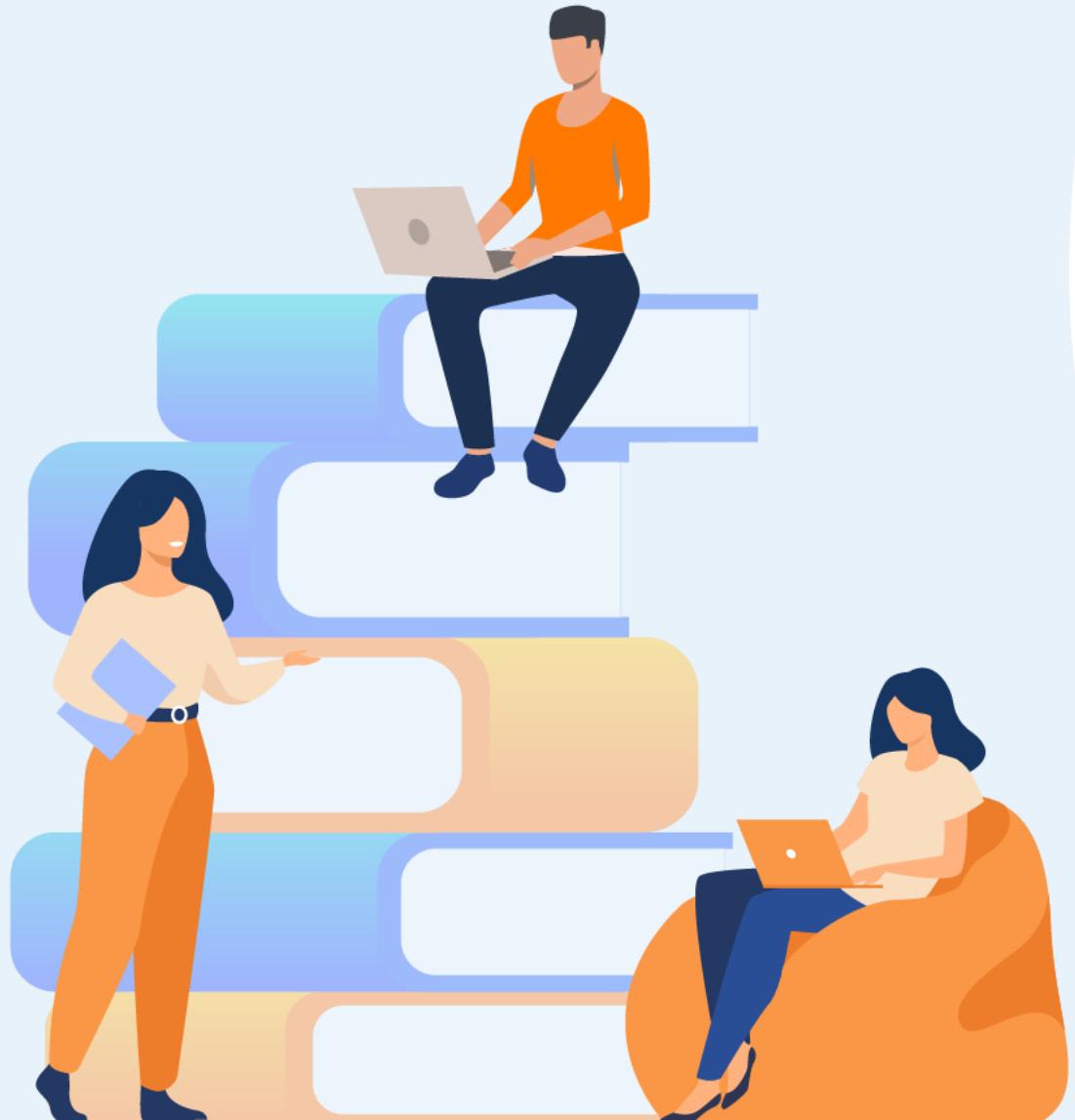
Vous disposez de contenus interactifs sous forme d'exercices et de cours à utiliser sur WebForce Life



5

DES RESSOURCES EN LIGNES

Les ressources sont consultables en synchrone et en asynchrone pour s'adapter au rythme de l'apprentissage



PARTIE 1

Conception d'une base de données

Dans ce module, vous allez :

- Analyser les données d'un cahier de charges
- Construire des modèles conceptuels (MCD) et logiques (MLD) de données
- Maîtriser la normalisation
- Connaitre les règles de passage du MCD au MLD normalisé



 17 Heures



CHAPITRE 1

ANALYSE DU CAHIER DES CHARGES



Ce que vous allez apprendre dans ce chapitre :

- La lecture et l'analyse d'un cahier des charges
- L'identification des limites du projet
- L'analyse des données et des traitements de la situation présentée

04 Heures



CHAPITRE 1

ANALYSE DU CAHIER DES CHARGES

1. **Lecture d'un cahier des charges**
2. Description des limites du projet
3. Analyse des données et des traitements de la situation présentée

01 - ANALYSE DU CAHIER DES CHARGES

Lecture d'un cahier des charges



Introduction

- Dans chaque organisation, il y'a une quantité importante d'informations qui sont échangées afin d'assurer le bon fonctionnement de cette organisation ainsi que la communication avec son environnement. Dans le but d'être utilisables dans les activités opérationnelles quotidiennes ou encore dans la prise de décision. Ces informations doivent être bien organisées et stockées. Il est donc nécessaire pour chaque organisme d'avoir une structure fonctionnelle et technique de gestion de l'information.

Le système d'information

- Le système d'information représente l'ensemble des éléments participants aux activités d'acquérir, de stocker, de traiter et de communiquer les informations au sein d'une organisation. Il se compose des acteurs suivants :
 - ✓ **Les individus** : En plus des spécialistes des Systèmes d'Information chargés de la conception, la mise en œuvre et la gestion du système d'information, cette catégorie comprend aussi les personnes qui utilisent ce dernier pour acquérir, communiquer, stocker ou traiter des informations.
 - ✓ **Le matériel** : Il s'agit de tout dispositif physique permettant d'émettre, manipuler ou stocker l'information.
 - ✓ **Les logiciels et les procédures** : Ce sont les programmes qui sont nécessaires au fonctionnement du Système d'Information ainsi que les procédures qui gèrent les traitements manuels et automatisés.
 - ✓ **Les données** : Elles constituent la matière première des traitements : saisies, déduites ou calculées.
- Dans le cadre d'un système d'information, un projet informatique a pour objectif de construire une application informatique (logiciel et base de données) qui va servir comme un support informatisé, inclus dans un système d'information organisationnel.

01 - ANALYSE DU CAHIER DES CHARGES

Lecture d'un cahier des charges



Définitions :

Un projet informatique

- Un projet informatique est un projet dont les livrables sont des outils ou services informatiques (logiciels, systèmes d'information, sites web...). Il s'agit de projets généralement complexes. Ceci est principalement dû à la grande diversité des intervenants (techniciens, responsables métier, marketeurs, gestionnaires....) ainsi qu'à la difficulté de définir toutes les exigences.
- Le processus de développement d'un projet informatique passe par 5 phases :
 - 1. Elaboration du schéma directeur** : Il s'agit d'une étude globale du système d'information à construire. Le but de cette étape est de réaliser le schéma directeur ainsi que Le plan de développement informatique
 - 2. Etude préalable** : Il s'agit d'une étude critique de l'existant et de la définition des objectifs du nouveau système. Le but de cette étape est de produire un dossier d'étude et la prise de décision du choix de la solution.
 - 3. Etude détaillée** : Il s'agit de fournir avec précision la description de la solution souhaitée : Définir logiquement les données et les traitements informatiques, les interfaces, le matériel... et construire le planning de réalisation. Le but de cette étape est de produire un cahier des charges fonctionnel et technique.
 - 4. Réalisation** : Elle consiste à la production du logiciel, l'implantation des bases de données et la mise en place de la solution.
 - 5. Mise en œuvre de la solution et assurer la maintenance** : Adapter la solution aux évolutions de l'environnement.

01 - ANALYSE DU CAHIER DES CHARGES

Lecture d'un cahier des charges



Définitions :

Le cahier des charges

- Le cahier des charges est un document essentiel à l'élaboration et la réalisation d'un projet. Il s'agit du document sur lequel les développeurs se basent pour concevoir et implémenter une base de données.
- Il présente une description détaillée du besoin des utilisateurs à savoir :
 - ✓ Le contexte général.
 - ✓ L'objectif du projet.
 - ✓ Les fonctionnalités attendues.
 - ✓ Les flux d'information et les processus métier.
 - ✓ Les règles de gestions des données.
- Il existe deux types de cahier des charges:
 - ✓ Le cahier des charges technique (CDCT): Il contient les exigences et contraintes techniques, économiques, industrielles, environnementales et matérielles d'un projet. Il sert à définir l'environnement technique: Architecture technique, les outils à utiliser, les technologies..
 - ✓ Le cahier des charges fonctionnel (CDCF) décrit la structure, les besoins et fonctionnalités attendues du maître d'ouvrage. Il contient les informations qui permettent d'adresser les exigences liées au projet en précisant les conditions de réalisation. Le CDCF doit comporter assez de détails pour être compréhensible par tous les acteurs du projet.

01 - ANALYSE DU CAHIER DES CHARGES

Lecture d'un cahier des charges



La structure d'un cahier des charges

Un cahier de charge se compose de cinq éléments essentiels :

1- Contexte et présentation du projet : On commence par présenter l'entreprise et l'importance du projet dans son plan stratégique. On définit aussi les acteurs cibles et les objectifs, et le périmètre du projet. Cette partie contient aussi la description de l'existant (si d'autres implantations existent déjà)

Exemples :

Présenter l'entreprise:

- Le groupe Hospitalier SantéPro se compose de 4 hôpitaux. Sa mission est de fournir des services de santé pour les habitants de la région.

Présenter le projet:

- Refonte d'un système d'information hospitalier dans le but de :
- Augmenter la productivité du personnel
- Collecter plus d'information depuis les différents processus
- Minimiser le délai d'attente des patients

Définir le périmètre:

- La plateforme est utilisée par les différents hôpitaux du groupe, répartis sur la région. Il s'agit de plus de 2000 utilisateurs qui accèdent de manière journalière.

01 - ANALYSE DU CAHIER DES CHARGES

Lecture d'un cahier des charges



La structure d'un cahier de charge : (suite)

2- Description graphique et ergonomique : On y décrit la charte graphique ainsi que tous les éléments graphiques et ergonomiques exigés relatifs au nouveau projet.

Exemples :

- ✓ le logo
- ✓ la typographie
- ✓ les couleurs
- ✓ les illustrations

3- Description fonctionnelle et technique : Cette étape décrit qu'il faut définir toutes les spécifications techniques et fonctionnelles des livrables.

Exemples:

- ✓ Plateforme technique
- ✓ Technologies de développement
- ✓ Sécurité
- ✓ Données à collecter
- ✓ Règles de gestion

01 - ANALYSE DU CAHIER DES CHARGES

Lecture d'un cahier des charges



La structure d'un cahier de charge : (suite)

4- Définition des résultats attendus : On présente dans ce stade toutes les prestations attendues à la fin du projet ainsi que les délais de livraison:

Exemples:

- ✓ Livrer des exécutables /packages.
- ✓ Serveur web configuré et installé sur les lieux

5- Budgétisation et fixation des délais : Cette phase concerne l'estimation du budget global permettant d'aiguiller les potentiels prestataires pour la réalisation de leurs devis :

- ✓ Un délai de réalisation de 200 jrs ouvrables
- ✓ Budget global de 1M de Dirhams.

01 - ANALYSE DU CAHIER DES CHARGES

Lecture d'un cahier des charges



Exemple d'un cahier des charges

The screenshot shows a template for a software development requirements document. At the top left, it says "CAHIER DES CHARGES" and "DÉVELOPPEMENT D'UN LOGICIEL". On the right side, there is a "SOMMAIRE" section with the following table of contents:

A. Présentation de l'entreprise
A.1. Les objectifs
A.2. Les cibles
B. Les concurrents
C. Développement du logiciel
C.1. Caractéristiques et fonctionnalités
C.2. Structure du logiciel
D. Charte graphique du logiciel
E. Les spécificités et les livrables
E.1. Les contraintes techniques
E.2. Les livrables
E.3. Le planning

At the bottom left, there is a form with fields for company information:

Nom de l'entreprise :
Nom du projet :
Personne à contacter dans l'entreprise :
Adresse :
Tel :
Email :

At the bottom center, it says "Exemple de cahier des charges pour développement logiciel".

L'exemple ci-contre illustre les informations que l'on peut indiquer sur la page de couverture ainsi que les rubriques qui y figurent sur le sommaire.





CHAPITRE 1

ANALYSE DU CAHIER DES CHARGES

1. Lecture d'un cahier des charges
2. **Description des limites du projet**
3. Analyse des données et des traitements de la situation présentée

01 - ANALYSE DU CAHIER DES CHARGES

Description des limites du projet



Introduction :

- Le cahier des charges représente les attentes et les besoins du client ainsi que les contraintes du client.
- En procédant à la lecture du cahier des charges, il faut définir le périmètre du projet :
 - Le contexte du projet.
 - L'ensemble des données que le système est supposé gérer et stocker.
 - Les conditions et règles de gestion exprimées par le client.



Il faut aussi définir les limites du projet et répondre aux questions suivantes :

Quels sont les éléments de données cités par le document ?

Qui fait quoi ?

Quelles sont les données qui peuvent/doivent être précisées ?

Quand s'arrêter ? A-t-on tout pris en compte ?

Quelles limitations présente la situation actuelle ?

01 - ANALYSE DU CAHIER DES CHARGES

Description des limites du projet



Périmètre d'un projet :

Le périmètre du projet correspond à la délimitation précise du projet. Il s'agit de la liste des objectifs, des produits livrables, des affectations, des dépenses et des délais qui doivent être respectés. Ces termes de références définissent aussi les limites du projet. Si le périmètre d'un projet est efficacement tracé, la gestion des améliorations qui surviennent lors de la mise en œuvre et de la maintenance devient plus simple.

Concernant un projet lié aux Systèmes d'Information (mise en place d'un nouvel ERP, évolution d'un SI en fonction d'une nouvelle organisation, développement d'une plateforme web ...), le périmètre total est l'identification et le recensement des applications/modules impactés par le projet.

A partir du cahier des charges ainsi que des échanges avec les porteurs du projet, on peut définir le périmètre et dresser les limites du projet en suivant les étapes suivantes :

1. Définir les buts : Il s'agit des objectifs à réaliser par le biais du projet.

Exemple : Dans le cas d'un projet d'informatisation des activités d'un centre de formation, le but serait de : *créer des formulaires d'inscriptions pour les étudiants qui vont simplifier les processus d'inscription et leur prise en charge et de suite assurer l'accès aux informations nécessaires à la bonne gestion du centre.*

2. Définir les livrables : Il faut identifier les résultats attendus du projet : c'est-à-dire l'ensemble des livrables. L'identification des livrables permet de détecter les dérives des objectifs si celles-ci surviennent.

Exemple : Quelles choses tangibles nous devons créer pour le compte du client (Centre de formation) ? Dans ce cas, il s'agit du formulaire informatisé ainsi que la base de données des inscriptions.

01 - ANALYSE DU CAHIER DES CHARGES

Description des limites du projet



3. Définir les tâches et les activités du projet

Il s'agit des moyens qui vont permettre la création des livrables et la réalisation des but du projet. Les livrables sont ainsi découpés en tâches et activités distinctes. Ceci permet de faciliter la gestion des projets surtout quand la complexité est importante.

Exemple : Pour créer le formulaire d'inscription on doit :

- Choisir et rédiger un exemplaire du formulaire
- Développer une interface de saisie du formulaire.
- Créer une base de données pour stocker les informations du formulaire...

4. Définir les contraintes du projet

Les trois principales contraintes d'un projet sont le budget, le temps et la portée:

Contrainte de budget ou cout : L'ensemble des ressources financières (frais du matériels, services et ressources humaines) nécessaires pour la réalisation du projet dans le respect des limites et délais prédéfinis.

Contrainte de temps : Le calendrier de livraison du projet en totalité ainsi que des différentes phases du projet.

Contrainte de portée : La définition des objectifs, des livrables, des fonctionnalités et des tâches à accomplir pour la finalisation du projet.

Exemple : Voici quelques contraintes applicables à votre questionnaire client :

- Le projet doit être bouclé en 6 mois
- Le budget total pour le projet ne doit pas dépasser 50 000 Dirhams
- L'équipe de développement ne pourra pas finaliser la conception dans 3 mois.



CHAPITRE 1

ANALYSE DU CAHIER DES CHARGES

1. Lecture d'un cahier des charges
2. Description des limites du projet
3. **Analyse des données et des traitements de la situation présentée**

01 - ANALYSE DU CAHIER DES CHARGES

Analyse des données et des traitements de la situation présentée



Introduction :

- On inclut souvent dans un projet informatique : les bases de données et le système informatique comprenant les ressources et infrastructures réseau, les applications et aussi les règles et dispositifs de sécurité.
- L'élaboration des bases de données afin de gérer l'accès aux données, le stockage et le traitement représentent un pilier du livrable d'un projet.
- Le cahier des charges relatif à un projet informatique indique les différents volets qui concernent ce projet, notamment la gestion des données.

Comment, à partir de ce cahier de charges, élaborer la solution souhaitée ?

- Il faut utiliser des méthodes de modélisation et de conception du système et de la base de données.
- Dans ce qui suit, on s'intéresse au volet du cahier des charges qui concerne la base de données.

01 - ANALYSE DU CAHIER DES CHARGES

Analyse des données et des traitements de la situation présentée



Définition :

- Une des fonctions d'un système informatique dans une organisation est de stocker et gérer les données nécessaires à son bon fonctionnement, d'où la nécessité du concept des bases de données.

Une base de données

- **Une base de données** est une structure permettant de stocker un grand nombre d'informations afin d'en faciliter l'utilisation.

Objectifs :

- Centraliser le stockage des informations
- Faciliter l'accès à l'information
- Assurer la justesse et la cohérence des informations stockées surtout lors des modifications
- Garantir l'intégrité et la confidentialité des données

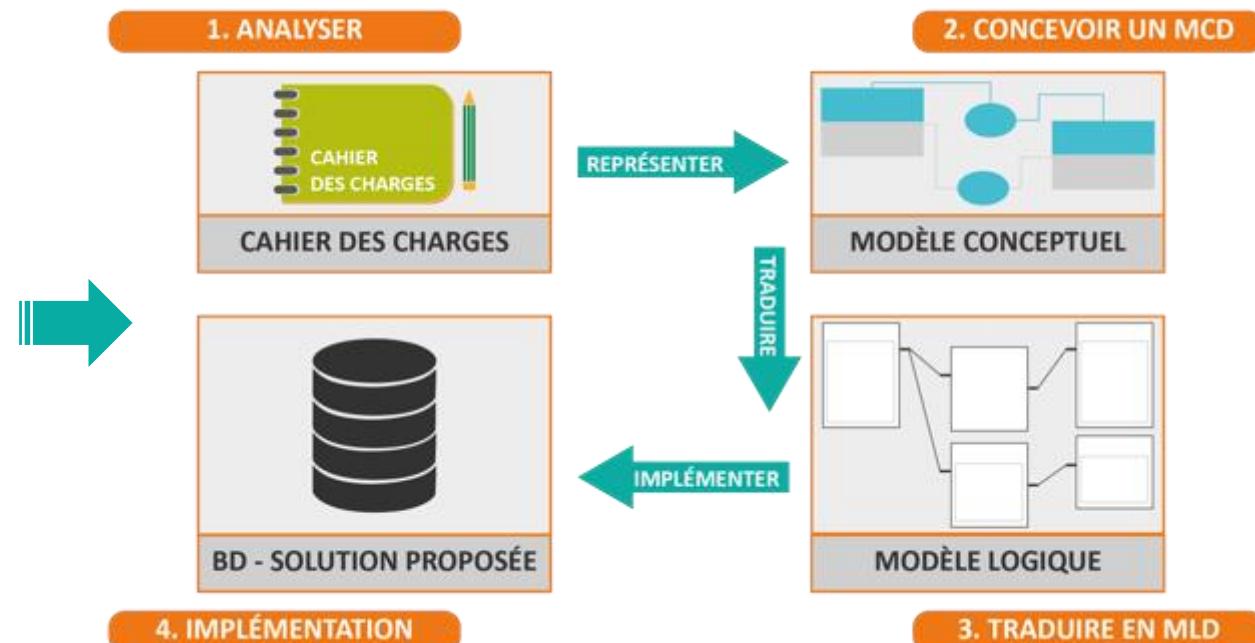
01 - ANALYSE DU CAHIER DES CHARGES

Analyse des données et des traitements de la situation présentée

- La conception d'une base de données passe par quatre phases comme illustré par le schéma :

1. Analyse du cahier des charges et clarification du besoin du client.
2. Conception d'un modèle conceptuel qui représente tous les éléments nécessaires du projet.
3. Traduction du modèle conceptuel en modèle logique.
4. Implémentation de la base de données proposée.

Dans le schéma ci-contre, la première étape de la conception d'une base de données se base sur l'analyse pertinente du cahier des charges et la bonne compréhension des besoins exprimés par les utilisateurs. Elle est par la suite essentielle et délicate en même temps.



Les phases de conception d'une base de données

01 - ANALYSE DU CAHIER DES CHARGES

Analyse des données et des traitements de la situation présentée



Exemple : Cahier des charges du projet: Gestion d'un centre de formation

- Un centre de formation désire stocker et gérer des données concernant les étudiants et les formations dans lesquelles ils sont inscrits. Le travail demandé est la modélisation des données persistantes et la représentation sous forme tabulaire de ces données telles qu'elles seront stockées dans la base de données.
- Les étudiants choisissent la formation et la session de cette formation dans laquelle ils veulent s'inscrire et payent le prix de la formation.
- Un étudiant est défini par **son numéro de CIN**. Il est, lors de son inscription, amené à remplir une fiche contenant **son nom et prénom, sa date de naissance, son adresse, sa ville et son niveau scolaire**.
- Puis, depuis le catalogue des **formations**, il doit choisir la formation souhaitée, et la **session** relative à cette formation. Il indique aussi le **type de cours** qu'il veut suivre (présentiel ou à distance). Une fiche d'inscription est conservée par l'administration (voir annexe).

Définition de l'**objectif** de la base de données :

- Gérer les données des étudiants, formations et inscriptions.

Définition des **processus** métier :

- Gérer les données des étudiants, formations et inscriptions.

Définition des **données** de la base de données :

- Étudiants (CIN, nom, prénom...).
- Formations, sessions, type des cours...
Il faut consulter les documents en annexe pour voir les fiches actuelles contenant la liste des données collectées.

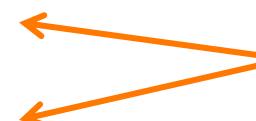
01 - ANALYSE DU CAHIER DES CHARGES

Analyse des données et des traitements de la situation présentée



Exemple : (suite)

- Pour chaque formation, le catalogue précise **le code, le titre, la durée, le prix et les spécialités (code et nom)** qui concernent cette formation ainsi que **les sessions ouvertes avec leurs date début et date fin**.



Définition des **données** de la base de données :

- Formation (code, titre, durée, prix...)
- Spécialité...
- Session (date début, date fin...)

- Voici quelques règles de gestion mises en œuvre par la direction du centre :

- Un étudiant peut être inscrit dans plusieurs sessions de formations.
- La formation peut se tenir en plusieurs sessions.
- Un étudiant ne peut pas être inscrit à plusieurs sessions de la même formation.
- Une formation n'est ouverte que s'il y a plus de 10 étudiants inscrits.
- Une formation peut faire partie de plusieurs spécialités.



Définition des **règles de gestion** :

- Conditions et contraintes à respecter lors de la modélisation de la base de données.

01 - ANALYSE DU CAHIER DES CHARGES

Analyse des données et des traitements de la situation présentée



- Les règles de gestion ainsi que les informations collectées permettent de définir les éléments de la base de données que nous allons construire, les relations entre ces éléments et aussi d'assurer **l'intégrité** des données :
 - Exhaustivité
 - Exactitude
 - Cohérence des données

Exemple :

- Si l'on veut modéliser les données de ce centre, nous allons créer un ensemble de tables liées entre elles par des relations :
 - Une table (ou entité) **ÉTUDIANT** qui contiendra des attributs :
 - Nom
 - Prénom
 - Adresse...
 - Une table (ou entité) **FORMATION** qui contiendra des attributs :
 - Titre
 - Durée
 - Prix...
 - Une table (ou entité) **SESSION** qui contiendra des attributs :
 - Code Formation
 - Date
 - Lieu...



CHAPITRE 2

MODÉLISATION DES DONNÉES

Ce que vous allez apprendre dans ce chapitre :

- L'élaboration des dictionnaires de données
- L'identification des dépendances fonctionnelles
- La construction du Modèle Conceptuel de Données (MCD)



08 heures



CHAPITRE 2

MODÉLISATION DES DONNÉES

1. **Contraintes déduites des règles de gestion**
2. Dictionnaire de données
3. Construction du graphe de dépendances fonctionnelles
4. Règles de passage du graphe au modèle conceptuel de données
5. Construction du modèle conceptuel de données

02 - Modélisation des données

Contraintes déduites des règles de gestion

- Les règles de gestion fournies par le cahier des charges permettent d'identifier les éléments de données de la base à concevoir. Ces règles doivent être traduites en contraintes afin d'assurer l'intégrité des données et la validation des modèles à construire.
- Identification des éléments de données :** Exemple du cahier des charge du centre de formation :



Les attributs des entités «**ÉTUDIANT**», «**FORMATION**», «**SESSION**» et «**SPÉCIALITÉ**» peuvent être déduits du texte ainsi que des fiches de renseignement données en annexe du cahier des charges.

Fiche d'évaluation					
Stage :					
Dates session:	Du _____ au _____				
Nom stagiaire et affectation					
Mettre une croix dans la case de la grille qui vous paraît le mieux correspondre à votre appréciation.					
Rubriques	4	3	2	1	0
Objectifs annoncés / objectifs atteints Correspondance entre le contenu de la formation et les objectifs décrits dans la fiche de stage.	Très bien	Bien	Moyenne	Faible	Insuffisante
Apports Apports sur le plan professionnel et/ou personnel de la participation à ce stage.	Très bien	Bien	Moyen	Faible	Insuffisant
Clarté du message Qualité pédagogique du message transmis par les intervenants.	Très bien	Bien	Moyen	Faible	Insuffisant
Équilibre théorie / illustrations Les illustrations ou applications pratiques sont pertinentes et viennent soutenir efficacement le discours théorique.	Très bien	Bien	Moyen	Faible	Insuffisant
Durée, rythme Adéquation entre la durée de la formation et le contenu.	Très bien	Bien	Moyens	Peu adaptées	Inadaptées
Animation, participation du groupe Prise en compte des réactions du groupe, participation du groupe.	Très bien	Bien	Moyenne	Faible	Insuffisante
Qualité de la documentation pédagogique Supports de cours, mise à disposition de documents en ligne...	Très bien	Bien	Moyenne	Faible	Insuffisante
Conditions matérielles Accueil, qualité du matériel mis à disposition...	Très bien	Bien	Moyennes	Faibles	Insuffisantes



02 - Modélisation des données

Contraintes déduites des règles de gestion



- Toujours dans le même sillage de l'exemple du cahier des charges relatif au centre de formation, les contraintes ci-après ont été identifiées :
 - Un étudiant peut être inscrit dans plusieurs sessions de formations.
 - La formation peut se tenir en plusieurs sessions.
 - Un étudiant ne peut pas être inscrit à plusieurs sessions de la même formation.
 - Une formation n'est ouverte que si il y a plus de 10 étudiants inscrits.
 - Une formation peut faire partie de plusieurs spécialités.
- Le tableau ci-après récapitule les règles de gestion relatives toujours au même exemple :

Règle N°	Énoncé de la règle
1	Un élément de l'entité ÉTUDIANT peut être associé à plusieurs éléments de l'entité SESSION.
2	Un élément de l'entité SESSION concerne un élément unique de l'entité FORMATION.
3	Un élément de l'entité FORMATION peut être associé à plusieurs éléments de l'entité SESSION.
4	Un élément de l'entité FORMATION peut être associé à un ou plusieurs éléments de l'entité SPÉCIALITÉ.



CHAPITRE 2

MODÉLISATION DES DONNÉES

1. Contraintes déduites des règles de gestion
2. **Dictionnaire de données**
3. Construction du graphe de dépendances fonctionnelles
4. Règles de passage du graphe au modèle conceptuel de données
5. Construction du modèle conceptuel de données

02 - MODÉLISATION DES DONNÉES

Dictionnaire des données



Avant la phase de conception du modèle conceptuel des données, il faut relever d'abord trois types d'informations à partir du cahier des charges.

Concept :

Il s'agit d' « objets » ou choses qui vont par la suite être des entités du schéma entité-association. Ce sont des éléments complexes qui peuvent être décomposés en plusieurs informations sous formes de « données ».

Exemples : étudiant, formation, session...

Contre-exemple : CIN, durée de la formation, nom de la session...

Donnée :

C'est une information élémentaire, qui ne peut pas être décomposée. Elle se trouve souvent liée à un concept.

Exemples : CIN, durée de la formation, nom de la session, salle, adresse étudiant...

Contre-exemple : numéro (sans préciser relatif à qui ou à quoi)..

Valeur :

Il s'agit d'occurrences ou exemples des données d'un concept.

Exemples : «G434568» valeur de la donnée : numéro de la CIN, relative au concept : étudiant.

Contre-exemples : nom de la formation, numéro de la CIN, étudiant...

02 - MODÉLISATION DES DONNÉES

Dictionnaire des données



Afin de réaliser un bon relevé d'informations, il faut :

- Bien identifier un **concept** : identifier les noms des objets, choses, personnes et types qui ont des données y afférentes.
- Relever uniquement les concepts et données qui concernent le système à concevoir. (faire attention aux détails inutiles).

Exemple :

- Le centre de formation « CF excellence » offre des formations aux étudiants des villes suivantes : Tanger, Rabat, Casablanca.
- **On constate que :**
 - « CF excellence » n'est pas une donnée, c'est le nom de l'organisation.
 - Formation, étudiant, session, spécialité sont des concepts.
 - Il y a une donnée rattachée au concept Étudiant qui est « Nom Ville ».
 - Tanger, Rabat, Casablanca sont des valeurs de la donnée « Nom Ville ».
 - Il y a un lien entre les concepts Formation et Étudiant.
 - ...

02 - MODÉLISATION DES DONNÉES

Dictionnaire des données



Dictionnaire des données : définitions

- Le dictionnaire des données contient toutes les données nécessaires qui vont être conservées dans la base de données. Il est souvent présenté sous forme d'un tableau qui indique pour chaque donnée les informations suivantes :
 - **Le code** : il s'agit d'un libellé désignant une donnée.
 - **La désignation** : description de la donnée.
 - **Le type de données** :
 - **Alphabétique** : lorsque les valeurs de la donnée sont composées de caractères alphabétiques.
 - **Numérique** : lorsque les valeurs de la donnée sont composées de nombres.
 - **Alphanumérique** : lorsque les valeurs de la donnée sont composées de caractères alphabétiques et numériques.
 - **Date** : quand il s'agit d'une date.
 - **Booléen** : vrai ou faux.
 - **La taille** : elle exprime la longueur des valeurs.
 - **Observations** : qui peut contenir des informations complémentaires.

02 - MODÉLISATION DES DONNÉES

Dictionnaire des données



Dictionnaire des données : définitions

- Après le relevé des données, il faut préciser :

1 - LA NATURE DE CHAQUE DONNÉE

Est-ce qu'il s'agit d'un concept, d'une donnée ou d'une valeur ?



2 - SON RÔLE

Est-ce qu'elle identifie un concept, est-ce qu'elle est calculée ou élémentaire ?



3 - LE CONCEPT AUQUEL ELLE APPARTIENT

Est-ce qu'elle appartient à un ou plusieurs concepts ?

02 - MODÉLISATION DES DONNÉES

Dictionnaire des données



- Ensuite, procéder aux contrôles suivants :
 - **Imprécisions** : s'assurer que les champs sont bien nommés et décrits

Exemple : « Ville » tout court n'est pas précis.

➤ On optera plutôt pour « Ville étudiant ».
 - **Polysémies** : il s'agit de deux données portant le même nom mais qui désignent des choses différentes.

Exemple : « Nom » relatif à l'étudiant et « Nom » relatif à la session.

➤ Il faut renommer les deux données en : Nom étudiant et Nom Session.
 - **Synonymes** : il s'agit de deux descriptions différentes qui désignent la même donnée.

Exemple : code CIN et numéro CIN.

➤ Il faut garder une seule description et supprimer les autres synonymes.
 - **Les données calculées** ne doivent pas figurer dans le dictionnaire des données, mais plutôt, il faut préciser les éléments qui ont permis ce calcul.

Par exemple : $\text{PRIXTTC} = \text{PRIXHT} * \text{TVA}$

➤ Les données à retenir seront : PRIX HT et TVA.

02 - MODÉLISATION DES DONNÉES

Dictionnaire des données



Bonne pratique :

- Afin d'uniformiser la nomenclature des données, on adopte une formule se composant de : objet (en minuscules) + raccourcis de nom du concept qu'elle représente (première lettre en majuscule).

Exemple :

- Numéro CIN de l'étudiant → numCINEtu
- Titre de la formation → titreForm

Le dictionnaire de données relatif à notre exemple de la gestion du centre de formation est comme suit : (1/2)

Code donnée	Désignation	Type	Taille	Observation
numCINEtu	Numéro CIN	Alphanumérique	9	Identifiant de l'étudiant
nomEtu	Nom de l'étudiant	Alphabétique	30	
prenomEtu	Prénom de l'étudiant	Alphabétique	30	
dateNaissEtu	Date de naissance	Date		
niveauEtu	Niveau scolaire	Alphanumérique	15	
nomvilleEtu	Nom de la ville	Alphabétique	15	
AdresseEtu	Adresse de l'étudiant	Alphanumérique	90	

02 - MODÉLISATION DES DONNÉES

Dictionnaire des données



- Le dictionnaire de données relatif à notre exemple de la gestion du centre de formation : (2/2)

Code donnée	Désignation	Type	Taille	Observation
codeForm	Code de la formation	Alphanumérique	9	Identifiant de la formation
titreForm	Titre de la formation	Alphanumérique	30	
dureeForm	Durée de la formation	Numérique	3	
prixForm	Prix de la formation	Numérique	5	
codeSess	Code de la session	Alphanumérique	9	Identifiant de la session
nomSess	Nom de la session	Alphanumérique	30	
dateDebutSess	Date du début de la session	Date		
dateFinSess	Date de la fin de la session	Date		
codeSpec	Code de la spécialité	Alphanumérique	10	
nomSpec	Nom de la spécialité	Alphanumérique	30	
descSpec	Description de la spécialité	Alphanumérique	90	
typeCours	Type de cours	Alphabetique		Distanciel ou présentiel



CHAPITRE 2

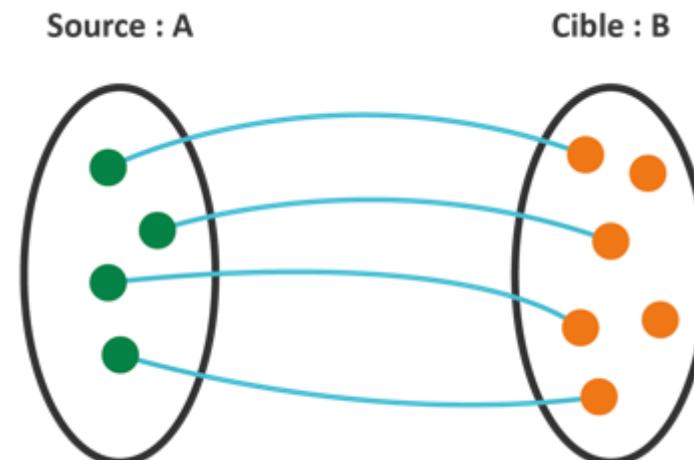
MODÉLISATION DES DONNÉES

1. Contraintes déduites des règles de gestion
2. Dictionnaire de données
3. **Construction du graphe de dépendances fonctionnelles**
4. Règles de passage du graphe au modèle conceptuel de données
5. Construction du modèle conceptuel de données

- Après le recensement des données dans un dictionnaire de données, l'étape suivante est celle de découvrir les relations entre ces données.

Dépendances fonctionnelles : Définition

- Soit deux groupes de données : A (source) et B (cible).
 - On entend par dépendance fonctionnelle une relation entre les deux groupes de données A et B de telle façon que :
 - Un élément du groupe A (source) permet de déterminer un et un seul élément du groupe B (cible).



Dépendances entre les éléments des deux ensembles A et B

02 - Modélisation des données

Construction du graphe de dépendances fonctionnelles



Exemple :

Code de Formation	Titre de Formation	Durée	Prix
ID01	Introduction au développement	3 mois	2500
CCP01	C/C++	30 jours	3000
ID02	Introduction au développement	3 mois	2700
BD001	Base de données	30 jours	2500

- Pour le concept Formation, la valeur du code de formation ID01 détermine que le titre de la formation est « Introduction au développement ».
- On peut déduire que :
 - Le code de formation **détermine une seule occurrence** du titre de formation.
 - Le titre de formation **dépend** du code de formation.
 - Cette relation est symbolisée sous cette forme : codeForm -> titreForm.
 - Cette relation n'est pas réversible : plusieurs formations peuvent avoir le même titre.

02 - Modélisation des données

Construction du graphe de dépendances fonctionnelles



La liste des dépendances fonctionnelles :

- La liste des dépendances fonctionnelles est élaborée à partir du dictionnaire des données.
- Il ne faut retenir que les dépendances directes et donc éliminer les transitivités :
- C'est-à-dire que si $D1 \rightarrow D2$ et $D2 \rightarrow D3$, alors $D1 \rightarrow D3$ est obtenue par transitivité et n'est pas, par la suite, directe.

Exemple :

- Voici la liste des dépendances fonctionnelles construite à partir du dictionnaire de données de notre exemple du centre de formation :

SOURCE	CIBLE
numCINETu →	nomEtu
numCINETu →	prenomEtu
numCINETu →	dateNaissEtu
numCINETu →	niveauEtu
numCINETu →	nomvilleEtu
numCINETu →	AdresseEtu

SOURCE	CIBLE
codeSpec →	nomSpec
codeSpec →	descSpec
codeSpec →	codeForm
codeSess →	nomSess
codeSess →	codeForm
codeSess →	dateDebutSess
codeSess →	dateFinSess

SOURCE	CIBLE
codeForm →	titreForm
codeForm →	dureeForm
codeForm →	prixForm

SOURCE	CIBLE
numCINETu + codeSess →	typeCours

02 - Modélisation des données

Construction du graphe de dépendances fonctionnelles



Contre-exemples :

SOURCE	CIBLE	REMARQUE
numCINETu →	nomForm	1) Un étudiant peut être inscrit dans plusieurs formations à la fois.
codeform →	nomSpec	2) Une formation peut concerner plusieurs spécialités.
NomEtu →	NomForm	3) NomEtu n'induit pas NomForm, aucune relation directe n'existe entre les deux.

1) Cette dépendance suggère que l'on peut déduire le nom de la formation à partir du code de CIN de l'étudiant. Ceci n'est pas correct du à la règle de gestion qui indique qu'un élève peut être inscrit dans plusieurs formations.

2) Cette dépendance suggère que l'on peut déduire le nom de la spécialité à partir du code de la formation. Ceci n'est pas correct du à la règle de gestion qui indique qu'une formation peut apparaître dans le curriculum de plusieurs spécialités.

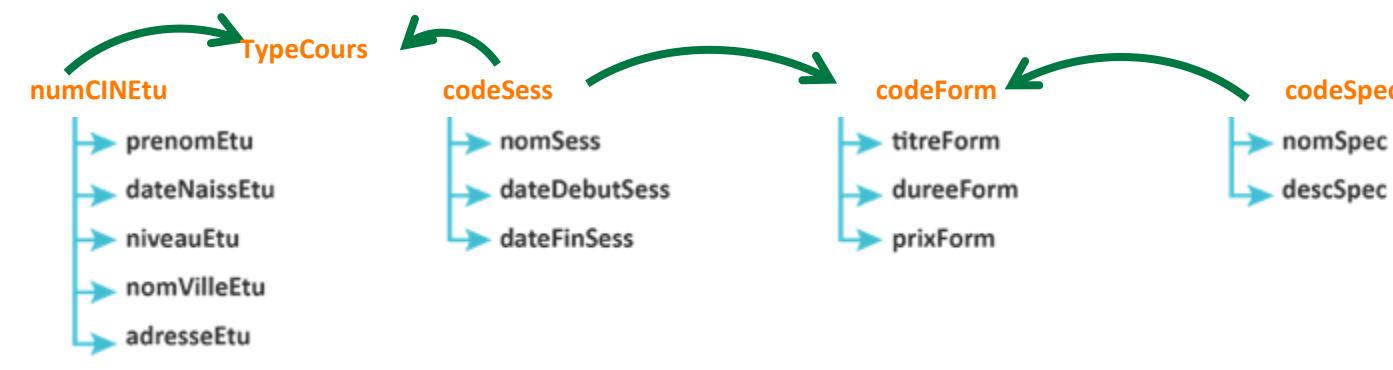
3) Cette dépendance suggère que l'on peut déduire le nom de la formation à partir du nom de l'étudiant. Ceci n'est pas correct car d'une part, le nom de l'étudiant n'est pas un identifiant, et qu'aucune relation directe n'existe entre ces deux attributs.

02 - Modélisation des données

Construction du graphe de dépendances fonctionnelles

Le graphe des dépendances fonctionnelles :

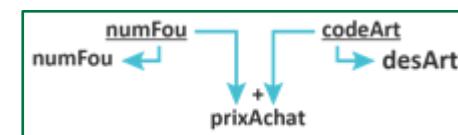
- Le graphe des dépendances fonctionnelles est la représentation graphique des dépendances fonctionnelles entre les données. Dans le cas du même exemple, le graphe des dépendances fonctionnelles est le suivant :



- On remarque que la donnée **typeCours** dépend de la combinaison de **numCINEtu** et **codeSess**.

Par exemple :

- Le prix d'un article varie d'un fournisseur à l'autre.
- La DF sera présentée ainsi : numFournisseur + CodeArticle → PrixArticle
- Le signe + indique que les données numFournisseur et CodeArticle doivent être groupées pour obtenir le prix de l'article.





CHAPITRE 2

MODÉLISATION DES DONNÉES

1. Contraintes déduites des règles de gestion
2. Dictionnaire de données
3. Construction du graphe de dépendances fonctionnelles
4. **Règles de passage du graphe au modèle conceptuel de données**
5. Construction du modèle conceptuel de données

02 - Modélisation des données

Règles de passage du graphe DF au MCD

- La phase de conception des systèmes d'information, et bien évidemment les bases de données, nécessite le recours à des méthodes de modélisation. C'est-à-dire la représentation virtuelle des processus et données de telle façon à bien comprendre l'existant et bien définir les futures livrables.
Il existe plusieurs méthodes d'analyse et de conception, une des méthodes les plus utilisées étant la méthode **MERISE**.

Qu'est-ce que la méthode Merise ?

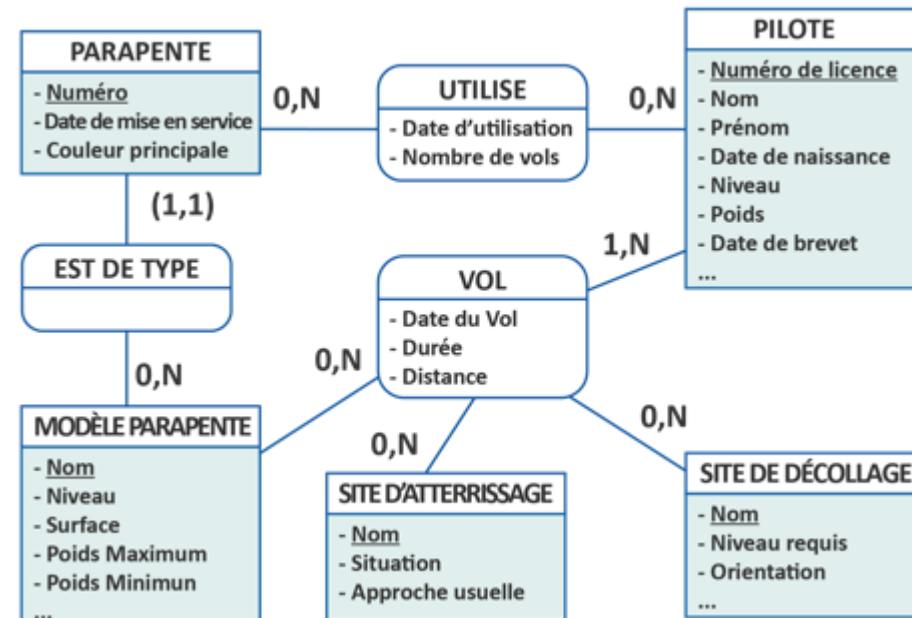
- La méthode Merise (**Méthode d'étude et de réalisation informatique pour les systèmes d'entreprise**) date de la fin des années 1970 en France. Il s'agit d'une méthode d'analyse et de conception de systèmes d'information qui se base sur le principe de la séparation des données et des traitements.
- La méthode Merise propose une démarche basée sur trois niveaux (ou cycles) : la conception, l'organisation et la technique. En effet, modéliser un système revient à produire une analyse globale de sa fonction : Décrire ce qu'il fait avant de se focaliser sur comment il le fait. Les données étant séparées des traitements, il faut vérifier la concordance entre données et traitements afin de vérifier que toutes les données nécessaires aux traitements sont présentes et qu'il n'y a pas de données superflues. Les trois niveaux de représentation des données, sont détaillés ci-dessous.
- **Niveau conceptuel** : le *modèle conceptuel des données (MCD)* décrit les entités du monde réel, en terme d'objets, de propriétés et de relations, indépendamment de toute technique d'organisation et d'implantation des données.
- **Niveau logique** : le *modèle logique des données (MLD)* adapte le modèle conceptuel au contexte organisationnel. Il s'agit d'une transcription du MCD dans un formalisme adapté à une implémentation ultérieure sous forme de base de données.
- **Niveau physique** : le *modèle physique des données (MPD)* permet d'établir la manière concrète dont la base de données sera construite.

02 - Modélisation des données

Règles de passage du graphe DF au MCD

Le modèle conceptuel de données :

- Le modèle conceptuel des données (MCD) formalise les données qui vont être stockées dans la base de données.
- Il s'agit donc d'une représentation des données, facile à comprendre, et qui permet de décrire la base de données à l'aide d'entités. La description par la méthode des entités association (MERISE) utilise les concepts suivants :
 - Entité
 - Association
 - Identifiants
 - Attributs
 - Cardinalité



Exemple d'un MCD

Entité et attributs :

- À partir du dictionnaire de données, on regroupe les données élémentaires par concept appelé **entité**. Une **entité** est un élément unique décrit par un ensemble de **propriétés** (aussi appelées **attributs**). Une de ces propriétés est la source des dépendances fonctionnelles avec le reste des propriétés. Elle joue le rôle d'un **identifiant** unique de l'entité.
- Le nom d'une entité est souvent un nom représentant un « objet de gestion ». Exemple : Étudiant, Formation, Article, Fournisseur...
- Une entité est formalisée comme suit :

Nom de l'entité
<u>Identifiant</u>
Propriété 1
Propriété 2
...

- Une entité est aussi l'ensemble des occurrences.

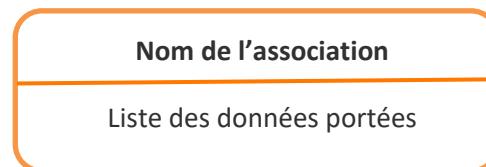
Exemple d'occurrences de l'entité FORMATION :

ID01
Introduction au développement
3
2500

CCP01
C/C++
30
3000

Association :

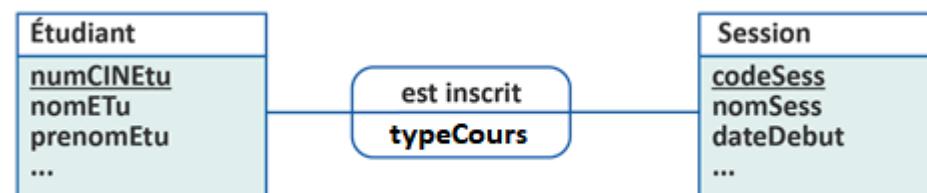
- Une association est un lien entre deux ou plusieurs entités. Ce lien est défini par des règles de gestions non traduites sous forme d'entité simple. Une association porteuse peut avoir des propriétés aussi. Une association est formalisée comme suit :



- Les données du dictionnaire de données qui dépendent de plusieurs entités sont mises dans l'association (porteuse) qui relie ces entités. Une association est dite binaire si elle relie entre deux entités, et tertiaire si elle relie entre trois entités.

Exemple :

- la règle de gestion : « Un étudiant est inscrit dans une session d'une formation » ainsi que « lors de l'inscription l'étudiant choisit le type de cours » sera représentée par l'association « est inscrit ». Cette association porte la donnée: typeCours.



02 - Modélisation des données

Règles de passage du graphe DF au MCD

Les cardinalités d'une association :

- Les cardinalités indiquent le nombre de fois où une entité est concernée par une association. Elles sont déduites des règles de gestion. Il y a trois valeurs typiques : 0, 1 et N (plusieurs).
- Les entités liées par une association possèdent chacune deux cardinalités : minimum et maximum. En effet, pour une association entre deux entités, il y aura quatre cardinalités à définir.
- Les cardinalités sont déduites à partir des règles de gestion. Ces règles sont propres à l'organisation étudiée et expriment des contraintes sur le modèle.

Remarque :

- Il se peut qu'il y ait des règles de gestions qui imposent un nombre précis de cardinalités. Ceci devra être géré par des traitements supplémentaires.

Les cardinalités d'une association :

- Comment définir les cardinalités ?

La cardinalité minimum :

- = 0 si la participation des occurrences d'une entité dans l'association est facultative.

Exemple : on peut avoir des Formations qui n'ont aucune session programmée.

- = 1 si la participation des occurrences d'une entité dans l'association est obligatoire.

Exemple : On ne peut pas avoir une session sans qu'elle soit liée à une formation.

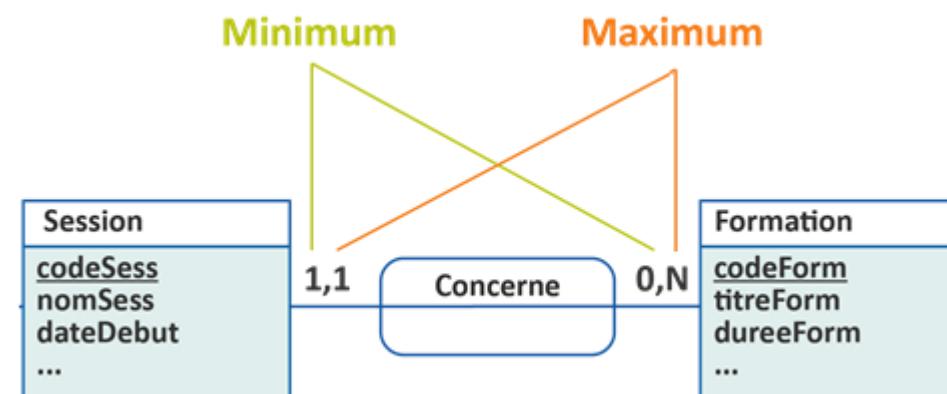
La cardinalité maximum :

- = 1 si la participation des occurrences d'une entité dans l'association est exclusive.

Exemple : une session concerne une et une seule formation.

- = N si la participation des occurrences d'une entité dans l'association est multiple.

Exemple : une formation peut avoir plusieurs sessions.



Les cardinalités d'une association :

- Voici un tableau récapitulatif des types de cardinalités les plus répandues :

Cardinalité	Signification
0, 1	Au plus un : chaque occurrence de l'entité n'est pas obligatoirement concernée par l'association et si elle l'est, c'est au plus une seule fois.
1, 1	Un et un seul : chaque occurrence de l'entité est concernée par l'association exactement une fois.
0, N	Zéro, un ou plusieurs : chaque occurrence de l'entité n'est pas obligatoirement concernée par l'association et si elle l'est, elle peut l'être plusieurs fois.
1, N	Au moins un : chaque occurrence est concerné par l'association et peut l'être plusieurs fois.

02 - Modélisation des données

Règles de passage du graphe DF au MCD



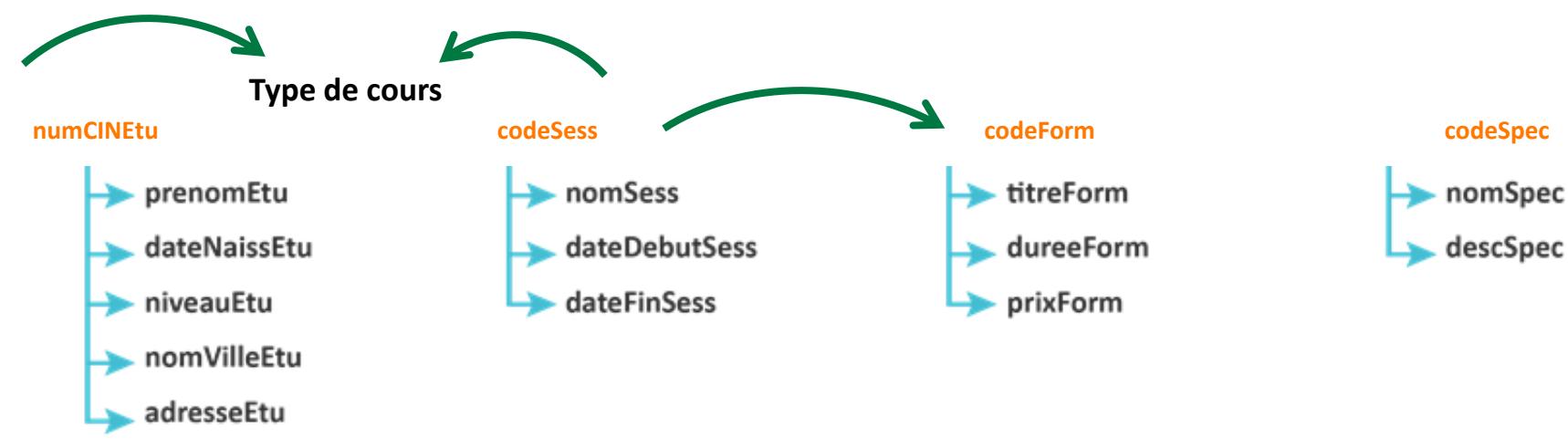
- Le passage du graphe des dépendances fonctionnelles au MCD se fait en respectant les règles suivantes :

	RÈGLES	OBSERVATIONS
N°1	Toute donnée du graphe devient une propriété.	-----
N°2	Chacune des données sources de dépendance fonctionnelle devient l'identifiant d'une entité.	-----
N°3	Une dépendance fonctionnelle entre deux données sources se traduit en association non porteuse de propriétés.	-----
N°4	Une donnée source de DF qui est relevée de l'association de plusieurs données élémentaires se traduit par une association porteuse de propriétés.	Il s'agit d'une association hiérarchique appelée aussi association fonctionnelle ou CIF (Contrainte d'Intégrité Fonctionnelle).
N°5	Des associations (issues de dépendances non fonctionnelles) peuvent exister dans un MCD sans pour autant faire partie du graphe des DF.	Il s'agit d'une association non hiérarchique appelée aussi association non fonctionnelle ou CIM (Contrainte d'Intégrité Multiple).

02 - Modélisation des données

Règles de passage du graphe DF au MCD

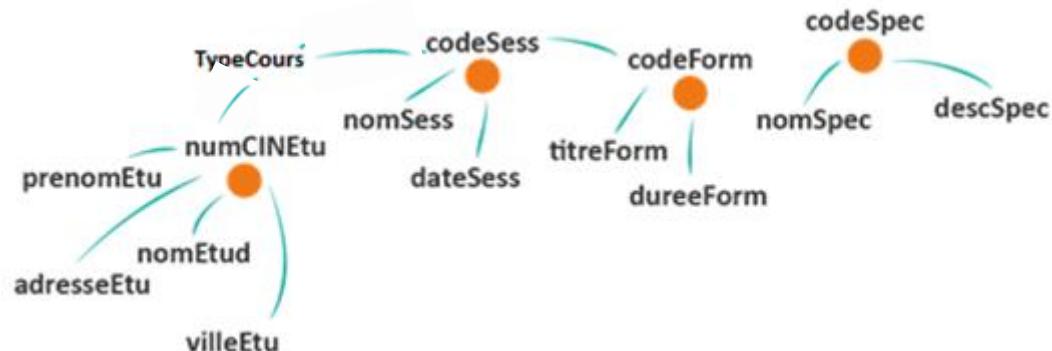
- Le graphe des DF permet de construire un modèle conceptuel de données (MCD) fiable. Même si les dépendances fonctionnelles sont la plupart du temps évidentes et ne nécessitent pas une représentation graphique, le graphe des DF aide toutefois à distinguer entre les futures éléments du MCD



02 - Modélisation des données

Règles de passage du graphe DF au MCD

Exemple : (voir graphe des dépendances fonctionnelles ci-dessous)



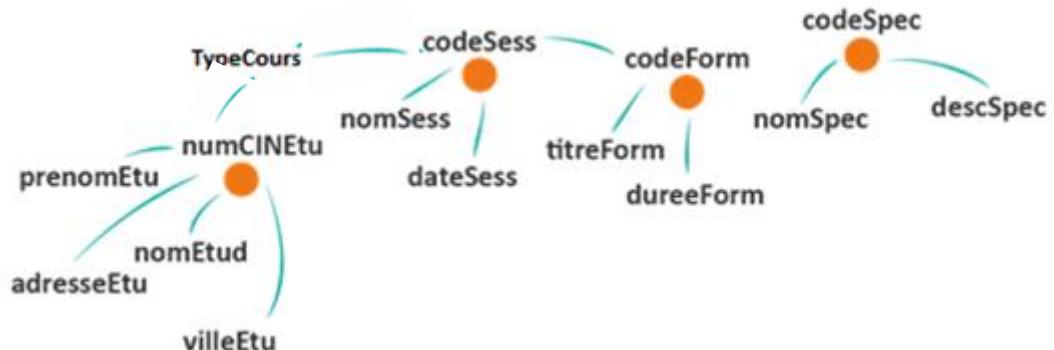
- En appliquant les règles N°1 et N°2, on peut déduire la liste des entités avec leurs propriétés et identifiants :
- ÉTUDIANT** (NumCINETU, nomEtud, prenomEtu, adresseEtu,...)
- FORMATION** (codeForm, titreForm, dureeForm, prixForm)
- SESSION** (codeSess, nomSess, dateDebutSess, dateFinSess)
- SPÉCIALITÉ** (codeSpec, nomSpec, descSpec)

RÈGLES	
N°1	Toute donnée du graphe devient une propriété.
N°2	Chacune des données sources de dépendance fonctionnelle devient l'identifiant d'une entité.
N°3	Une dépendance fonctionnelle entre deux données sources se traduit en association non porteuse de propriétés.
N°4	Une donnée source de DF qui est relevée de l'association de plusieurs données élémentaires se traduit par une association porteuse de propriétés.
N°5	Des associations (issues de dépendances non fonctionnelles) peuvent exister dans un MCD sans pour autant faire partie du graphe des DF.

02 - Modélisation des données

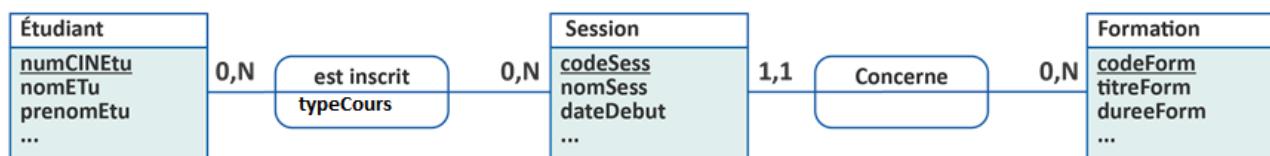
Règles de passage du graphe DF au MCD

Exemple : (voir graphe des dépendances fonctionnelles ci-dessous)



- Les règles N°3 et N°4 permettent de déduire les deux associations fonctionnelles : **Concerne** (règles N°3) - **est Inscrit** (règles N°4)

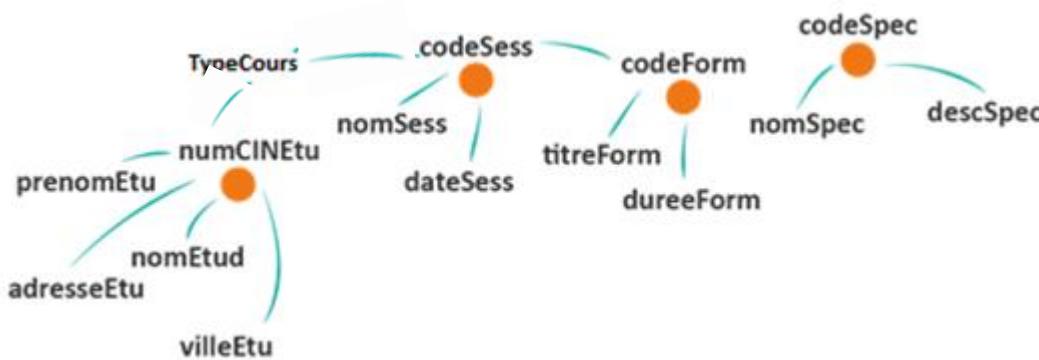
RÈGLES	
N°1	Toute donnée du graphe devient une propriété.
N°2	Chacune des données sources de dépendance fonctionnelle devient l'identifiant d'une entité.
N°3	Une dépendance fonctionnelle entre deux données sources se traduit en association non porteuse de propriétés.
N°4	Une donnée source de DF qui est relevée de l'association de plusieurs données élémentaires se traduit par une association porteuse de propriétés.
N°5	Des associations (issues de dépendances non fonctionnelles) peuvent exister dans un MCD sans pour autant faire partie du graphe des DF.



02 - Modélisation des données

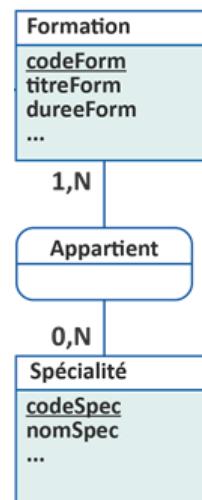
Règles de passage du graphe DF au MCD

Exemple : (voir graphe des dépendances fonctionnelles ci-dessous)



En appliquant la règle N°5, nous définissons l'association non fonctionnelle entre l'entité **FORMATION** et **SPECIALITE** :

Une formation peut appartenir à plusieurs spécialités, chaque spécialité peut contenir plusieurs formations



RÈGLES

	RÈGLES
N°1	Toute donnée du graphe devient une propriété.
N°2	Chacune des données sources de dépendance fonctionnelle devient l'identifiant d'une entité.
N°3	Une dépendance fonctionnelle entre deux données sources se traduit en association non porteuse de propriétés.
N°4	Une donnée source de DF qui est relevée de l'association de plusieurs données élémentaires se traduit par une association porteuse de propriétés.
N°5	Des associations (issues de dépendances non fonctionnelles) peuvent exister dans un MCD sans pour autant faire partie du graphe des DF.



CHAPITRE 2

MODÉLISATION DES DONNÉES

1. Contraintes déduites des règles de gestion
2. Dictionnaire de données
3. Construction du graphe de dépendances fonctionnelles
4. Règles de passage du graphe au modèle conceptuel de données
5. **Construction du modèle conceptuel de données**

02 - Modélisation des données

Construction du modèle conceptuel des données



Quelle démarche pour la construction du MCD ?

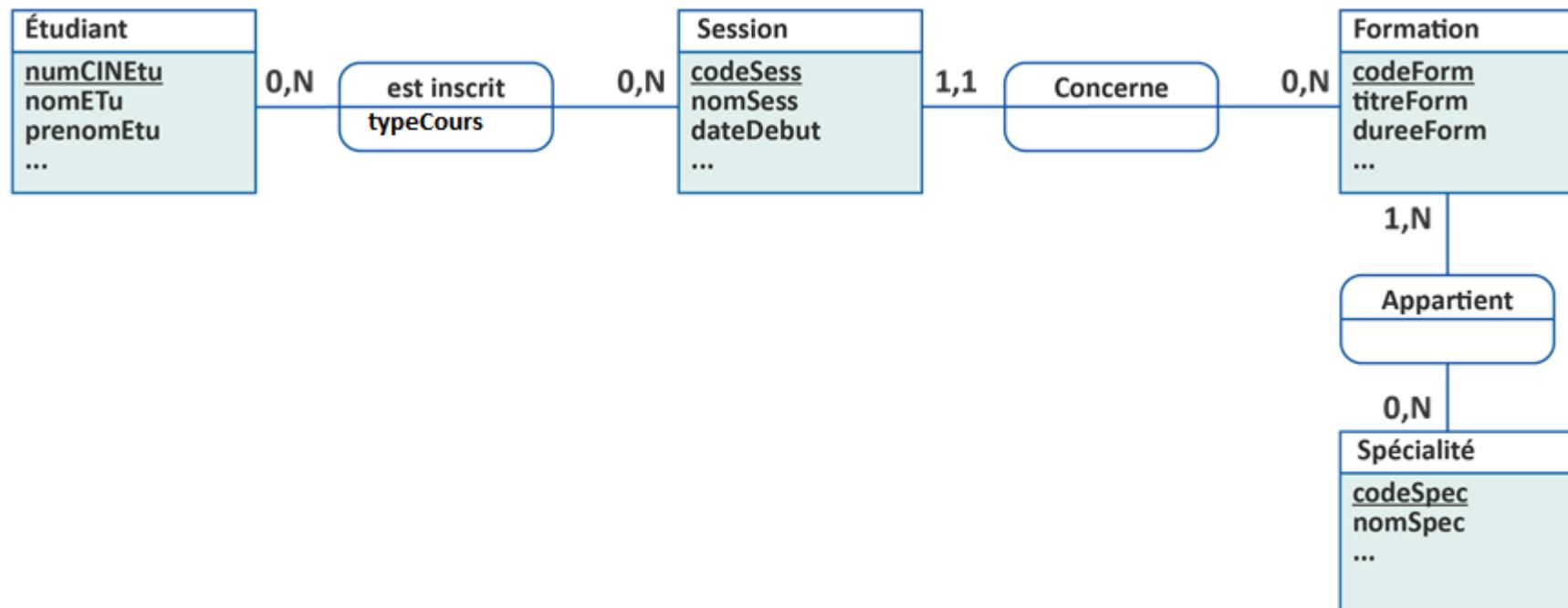
- À ce stade, il est possible d'élaborer le MCD complet à partir des entités et associations ainsi que les données du dictionnaire des données, et ce en suivant la démarche suivante :
 - Afin d'assurer la validité du modèle, il faut observer les points suivants :
 - Toutes les entités du MCD doivent être reliées à, au moins, une association.
 - S'assurer de la conformité du modèle aux contraintes et règles de gestion.
 - Évaluer le modèle contre ce que les utilisateurs comprennent.
 - En cas de modèle complexe, il faut commencer par les entités et associations élémentaires puis itérer en complétant progressivement jusqu'à ce que le modèle semble raisonnablement complet.

02 - Modélisation des données

Construction du modèle conceptuel des données

Exemple :

- MCD correspondant au projet du «Centre de formation» :





CHAPITRE 3

NORMALISATION DES DONNÉES

Ce que vous allez apprendre dans ce chapitre :

- L'identification des différentes formes normales
- La construction du modèle logique des données (MLD) normalisé



05 heures



CHAPITRE 3

NORMALISATION DES DONNÉES

1. **Formes normales**
2. Règles de passage du MCD au MLD normalisé

03 - Normalisation des données

Formes normales



- Les formes normales permettent la décomposition des entités en des relations, sans perdre d'informations, en se basant sur les dépendances fonctionnelles, dans le but de construire un schéma conceptuel représentant de manière correcte les associations canoniques du monde réel.
- Au niveau de la base de données, ce travail permet d'éviter les redondances et facilite la maintenance des données.
- Il existe différents niveaux de formes normales :
 - ✓ Première forme normale (1FN ou 1NF)
 - ✓ Deuxième forme normale (2FN ou 2NF)
 - ✓ Troisième forme normale (3FN ou 3NF)
 - ✓ Forme normale de Boyce-Codd (FNBC ou BCNF)
 - ✓ Quatrième forme normale (4FN – NF4)

03 - Normalisation des données

Formes normales

Première forme normale (1FN ou 1NF) : Attribut élémentaire

- Une relation est en première forme normale si, et seulement si, tous ses attributs sont atomiques et sont en dépendance fonctionnelle avec l'identifiant de cette relation. Ceci dit, dans un attribut, on ne peut avoir qu'une seule valeur. Un attribut est atomique s'il ne contient qu'une seule valeur pour un tuple (c'est-à-dire une ligne de données), c'est à dire qu'il ne regroupe pas un ensemble de plusieurs valeurs.

Exemple :

- La relation : Formation** (**codeForm**, **titreForm**, **codeSpec...**) : Cette relation n'est pas en 1FN car l'attribut « **codeSpec** » n'est pas en dépendance fonctionnelle avec l'identifiant: **codeFom**
- La relation : Étudiant** (**numCINEtu**, **nomEtu**, **prénoms...**) : Cette relation n'est pas en 1FN si on stocke plusieurs valeurs dans l'attribut « **prénoms** ». La forme correcte serait : Étudiant (**numCINEtu**, **nomEtu**, **prenom1**, **prenom2**, **prenom3...**).

numCINEtu	nomEtu	prénoms
G683909	Alaoui	Mohammed, Amine
AB123456	Hilali	Nour, laila
..		



numCINEtu	nomEtu	prénom1	prénom2
G683909	Alaoui	Mohammed	Amine
AB123456	Hilali	Nour	Laila
..			

03 - Normalisation des données

Formes normales



Deuxième forme normale (2FN ou 2NF) : Dépendance fonctionnelle élémentaire

- Une relation est en deuxième forme normale si elle vérifie les deux conditions suivantes :
 - Être en 1FN.
 - Les attributs non clé dépendent de la totalité de la clé, et non d'une partie de la clé.
- Dans le cas échéant, il faut diviser la relation en plusieurs relations regroupants un groupe d'attributs qui vérifieront la dépendance entre chaque morceau de la clé et la clé entière.

Exemple :

- **La relation** : Inscription (numCINETu, codeSess, nomEtu, villeEtu...) **n'est pas en 2FN**.
- Cette relation doit être divisée en deux :
 - Étudiant (numCINETu, nomEtu, villeEtu...)
 - Inscription (numCINETu, codeSess)

03 - Normalisation des données

Formes normales



Troisième forme normale (3FN ou 3NF) : Dépendance fonctionnelle élémentaire directe

- Une relation est en troisième forme normale si elle vérifie les deux conditions suivantes :
 - ✓ Être en 2FN.
 - ✓ Chacun des attributs de la relation ne dépend que de la clé et non pas d'un autre attribut de la relation.
- C'est-à-dire que toutes les dépendances fonctionnelles entre la clé primaire et les autres attributs doivent être directes, et ce pour éliminer les transitivités et les dépendances entre les attributs non clé.
- Dans le cas échéant, diviser la relation en autant de relations que de dépendances entre attributs non clé.

Exemple :

- **La relation** : Formation (codeForm, titreForm, codeSpecialite, nomSpecialite) **n'est pas en 3FN**.
- Cette relation doit être décomposée en deux :
 - Formation (codeForm, titreForm, codeSpecialite#)
 - Spécialité (codeSpecialite, nomSpecialite)

03 - Normalisation des données

Formes normales



Forme normale de Boyce-Codd (FNBC ou BCNF)

- Une relation est en FNBC si elle vérifie les deux conditions suivantes :
 - ✓ Être en 3FN.
 - ✓ Les seules dépendances fonctionnelles élémentaires existantes dans les relations sont celles de la clé vers les attributs non clés.
- Cette règle permet d'éliminer les redondances créées par des dépendances entre parties de clés ainsi que celles déjà éliminées par la 3FN.

Exemple :

- **La relation** : Commune (commune, ville, région, population) **n'est pas en FNBC**.
 - Si "commune + ville" déterminent la région et la population, on a aussi ville qui détermine région.
 - Donc, on doit décomposer cette relation en :
 - Commune (commune, ville, population)
 - Ville (ville, région)

03 - Normalisation des données

Formes normales



WEBFORCE
BE THE CHANGE

Quatrième forme normale (4FN – NF4)

- Les trois premières formes normales se focalisent sur des aspects très conceptuels et évidents, mais ne permettent pas d'éliminer toutes les redondances.
- On fait alors recours aux formes 4FN et 5FN pour ajouter une dimension de traitement de l'information et faciliter la mise à jour des données de la base.
- Une relation est en quatrième forme normale lorsque elle vérifie les deux conditions suivantes :
 - Être en 3FN.
 - Si, et seulement si, les dépendances **multi-valuées** élémentaires sont celles dans lesquelles une clé détermine la valeur d'une colonne.

03 - Normalisation des données

Formes normales

Exemple : 4FN

- **Dépendances multivaluées :** Voici la liste des modèles, couleurs et versions disponibles d'une voiture :

Modèle	Couleur	Version
715	Gris	Enjoy
715	Gris	Excellence
715	blanc	Enjoy
620	noir	Enjoy
620	blanc	Enjoy
620	blanc	Excellence

- Pour le même modèle d'une voiture, il peut exister plusieurs couleurs et plusieurs versions. La table ci-dessus illustre toutes les combinaisons possibles pour chaque modèle en terme de couleur et version. En effet, la dépendance entre Modèle et Couleur d'une part, et Modèle et Version d'autre part est dite dépendance multivaluée.
- La table ainsi modélisée "Disponibilité(modèle, couleur, version)" présente un inconvénient majeur : si pour un modèle X, on veut supprimer une valeur de la colonne version V, il faudra parcourir et supprimer toutes les combinaisons où modèles = X et Version = V. Le recours à la forme 4FN permet d'éviter ce genre de problèmes et ainsi on doit décomposer la table en deux relations :
 - DispoCouleur : (Modèle, Couleur)
 - DispoVersion : (Modèle, Version)



CHAPITRE 3

NORMALISATION DES DONNÉES

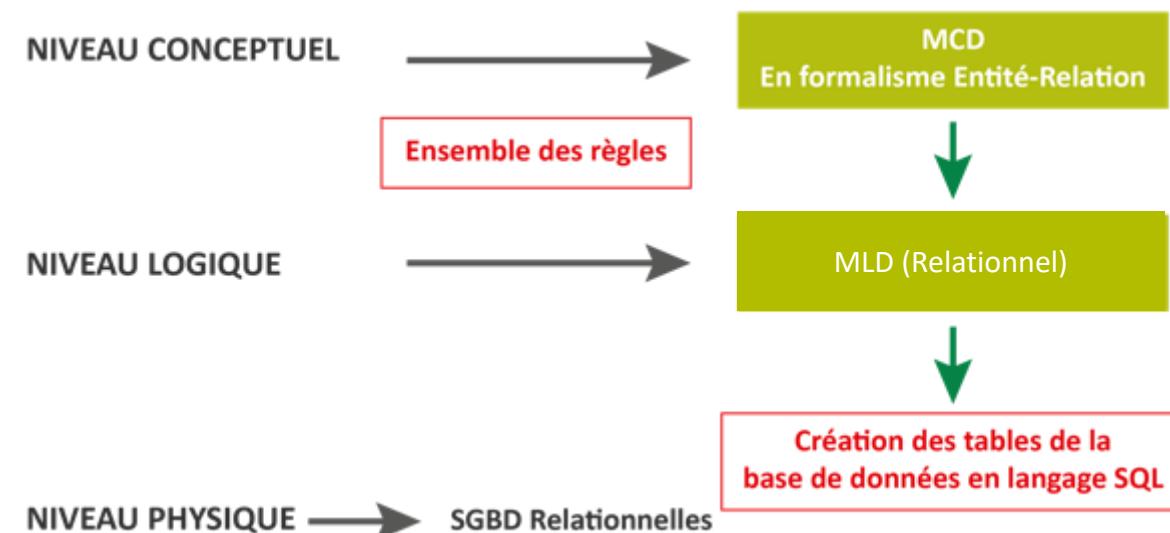
1. Formes normales
2. **Règles de passage du MCD au MLD normalisé**

03 - Normalisation des données

Règles de passage du MCD au MLD normalisé

Modèle Logique de Données (MLD) :

- Le modèle logique de données est une représentation du modèle de données en tables logiques reliées entre elles par des flèches. Il permet de modéliser la structure de la base de données à partir du MCD et est adapté au Systèmes de Gestion de Bases de Données Relationnelles (SGBDR).
- Du MCD AU MLD :



03 - Normalisation des données

Règles de passage du MCD au MLD normalisé

Règles de passage du MCD au MLD normalisé

- **Règle N°1 :** transformation des entités.
- **Règle N°2 :** transformation d'une association sans propriété du type (* ,n)-(1,1).
- **Règle N°3 :** transformation d'une association (1, n)-(*, n).
- **Règle N°4 :** associations ternaires (n-aires).

Règle N°1 : transformation des entités

- Une entité du MCD devient une table portant le même nom.
- Chaque ligne de la table correspond à un enregistrement.
- Chaque colonne correspond à un attribut.
- L'identifiant devient la clé primaire de la table.

Exemple :

- **ÉTUDIANT** (numCINEtu ,prenomEtu, dateNaissEtu, niveauEtu,nomVilleEtu, AdresseEtu)



Étudiant
numCINEtu
nomEtu
prenomEtu
dateNaissEtu
niveauEtu
nomVilleEtu
AdresseEtu

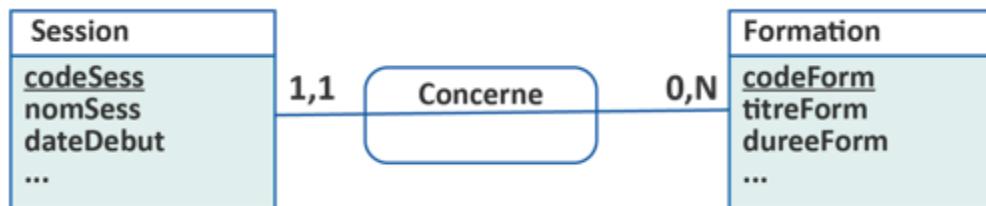
03 - Normalisation des données

Règles de passage du MCD au MLD normalisé

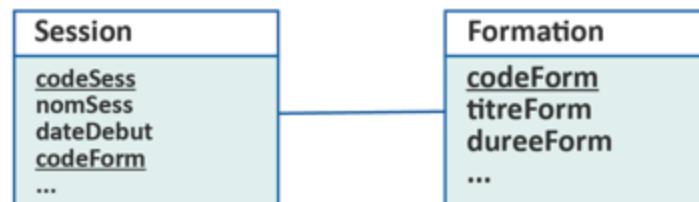
Règle N°2 : transformation d'une association sans propriété du type (* ,n)-(1,1)

- La clé primaire de la table, ayant la cardinalité (*,N), est dupliquée dans la table ayant la cardinalité (1,1).

Exemple :



- Une session concerne une seule formation.
- Une formation peut n'avoir aucune session.
- Formation est dite entité **forte** et Session est dite entité **faible**.
 - La clé primaire codeForm de la table Formation doit être dupliquée dans la table Session.
 - Formation** (codeForm)
 - Session** (codeSess)



03 - Normalisation des données

Règles de passage du MCD au MLD normalisé

Règle N°3 : transformation d'une association (1, n) - (*, n)

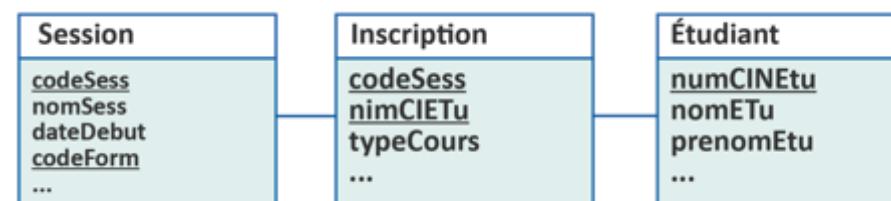
- Concerne les relations ou les cardinalités max des deux côtés de l'association = N.
- La relation est transformée en une entité.
- La clé primaire de cette entité est la combinaison des clés des relations correspondantes aux entités de part et d'autre de la relation.
- Les propriétés de l'association deviennent des attributs de l'entité.

Exemple :



- Un étudiant peut n'être inscrit à aucune session de formations comme il peut être inscrit à plusieurs.
- Une session peut n'avoir aucun étudiant inscrit, comme elle peut avoir plusieurs étudiants inscrits.
- L'association « est inscrit » est transformée en une nouvelle entité « Inscription ».

- Inscription (codeSess, numCINETu, typeCours)



03 - Normalisation des données

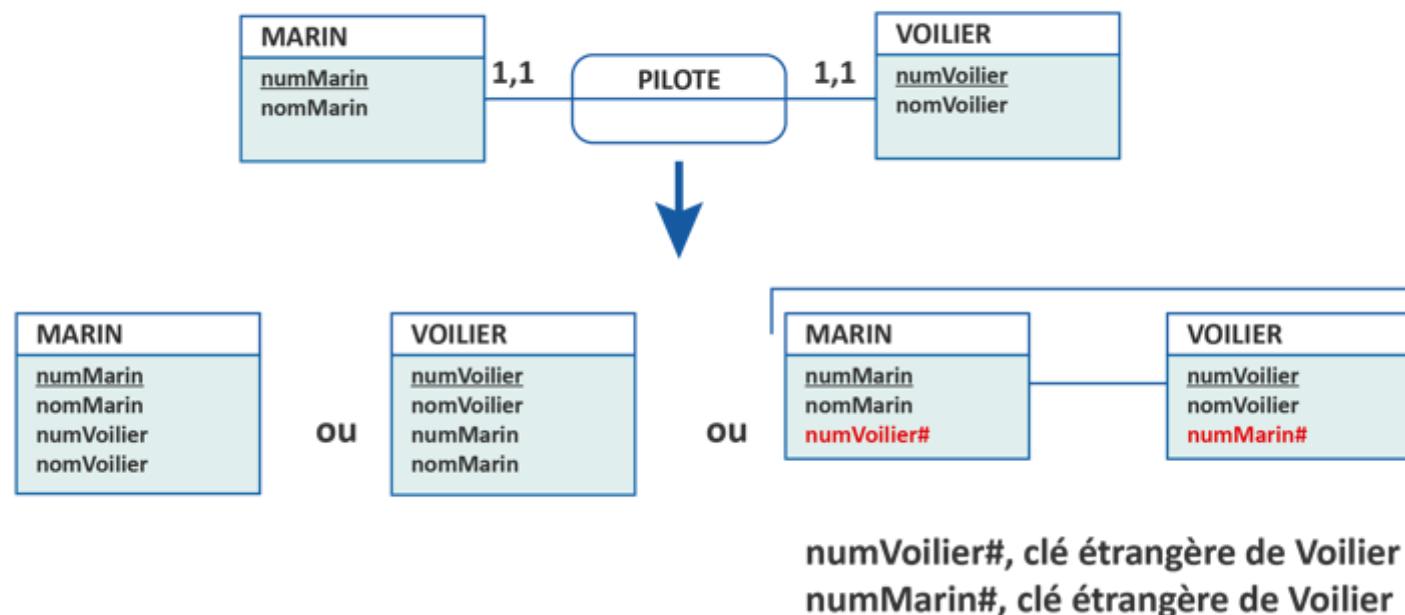
Règles de passage du MCD au MLD normalisé

Règle N°3 : transformation d'une association (1, n) - (*, n)

Cas particuliers :

- Associations 1,1

Exemple : course à la voile : 3 solutions



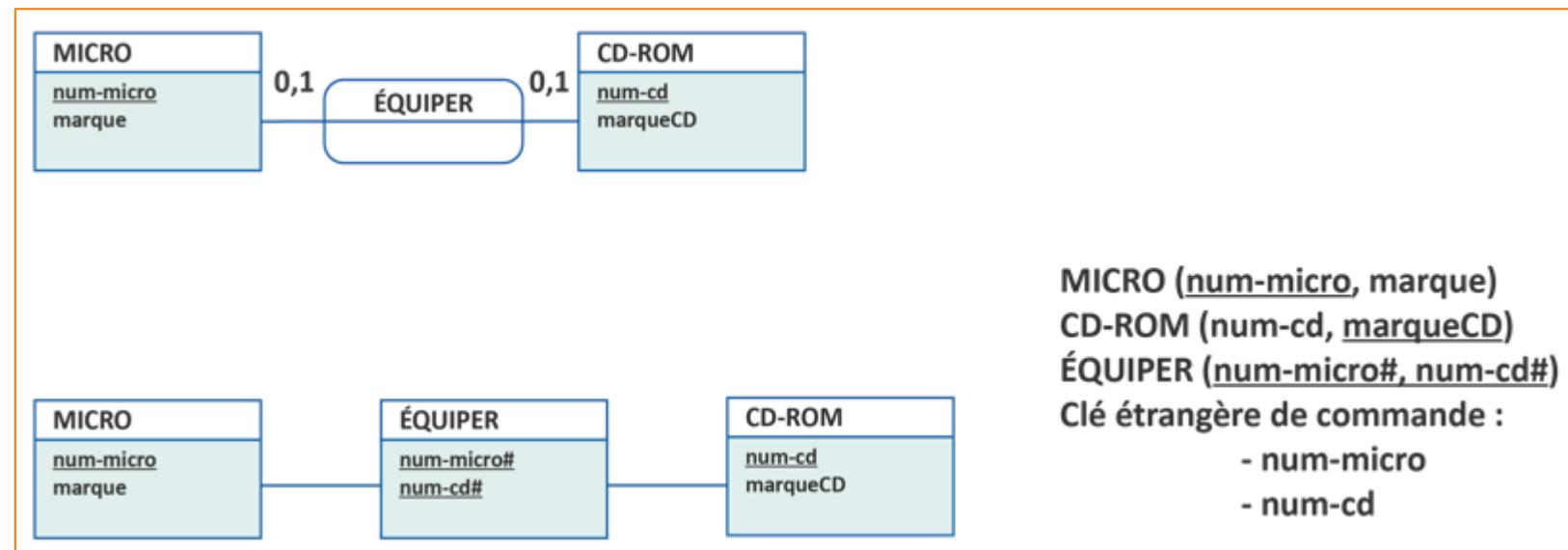
03 - Normalisation des données

Règles de passage du MCD au MLD normalisé

Règle N°3 : transformation d'une association (1, n) - (*, n)

Cas particuliers :

- Associations binaires 01,01



03 - Normalisation des données

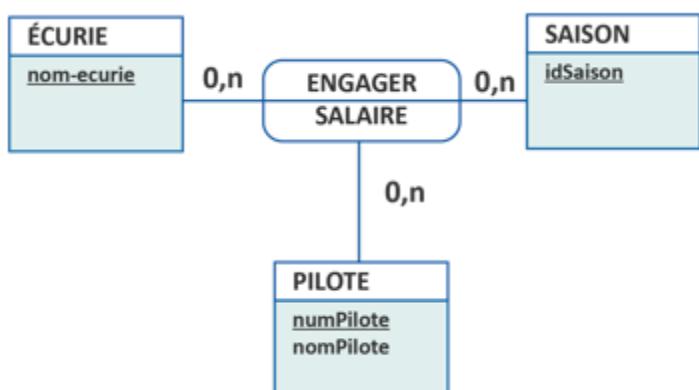
Règles de passage du MCD au MLD normalisé

Règle N°4 : associations ternaires (n-aires)

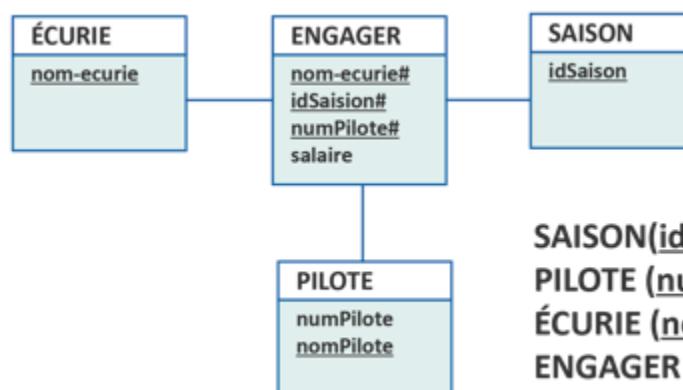
- L'association gère une table, qui reçoit en clé étrangère, les clés primaires des tables associées.
- La composition des clés étrangères devient la clé primaire de la table association.
- Les données éventuelles de l'association deviennent les attributs de la table association.

Exemple :

- L'association suivante :



Est transformée comme suit :

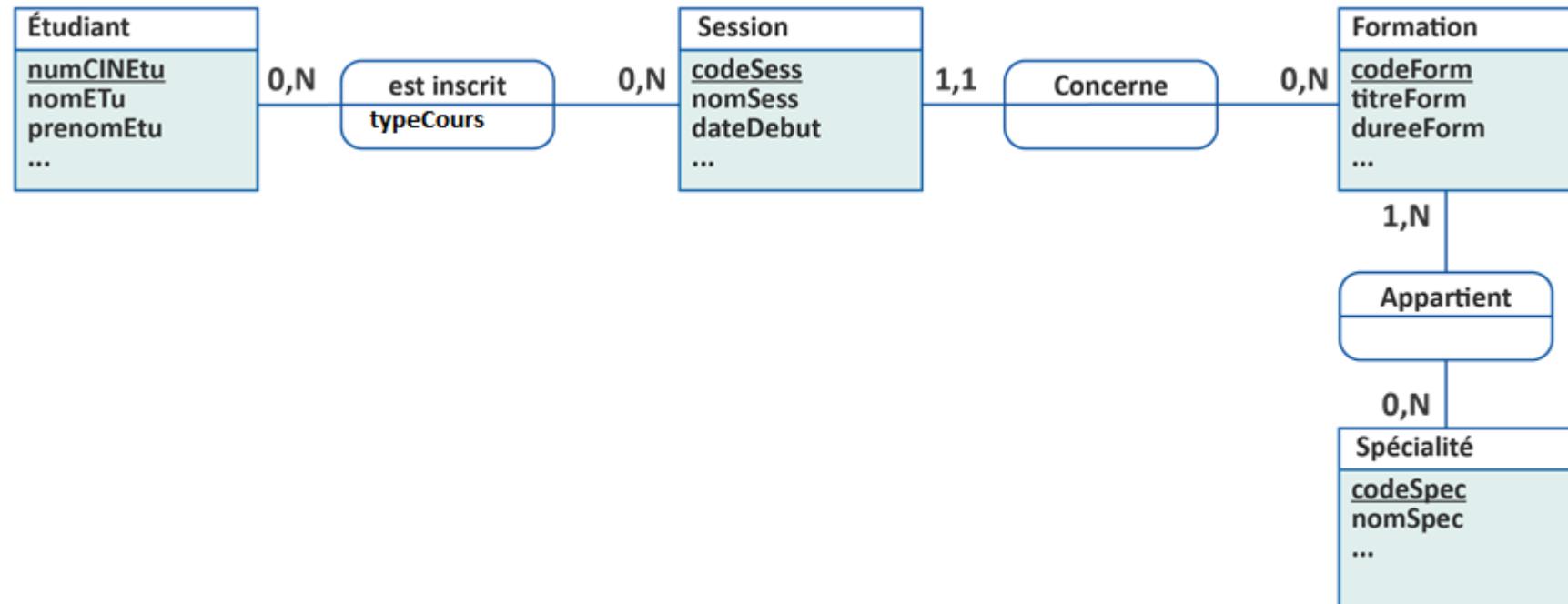


SAISON([idSaison](#))
PILOTE ([numPilote](#), [nomPilote](#))
ÉCURIE ([nom-ecurie](#))
ENGAGER ([idSaison](#), [numPilote](#), [nom-ecurie](#), [salaire](#))

03 - Normalisation des données

Règles de passage du MCD au MLD normalisé

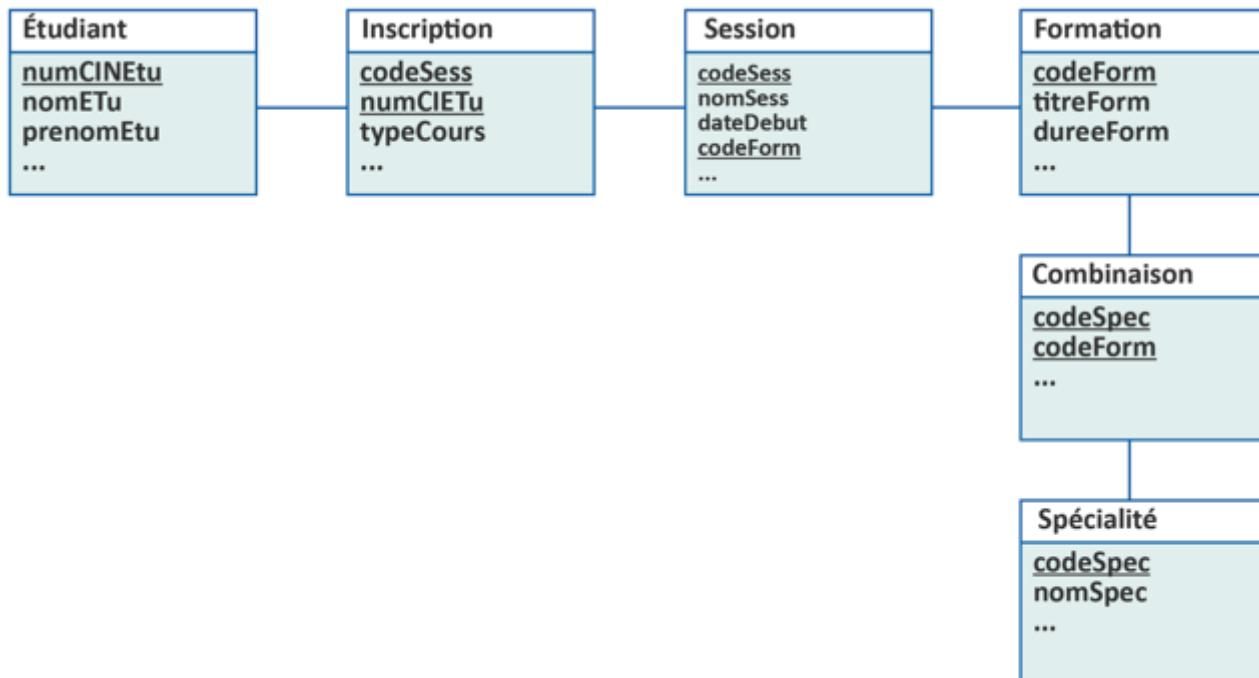
Le MCD de la base de données du centre de formation



03 - Normalisation des données

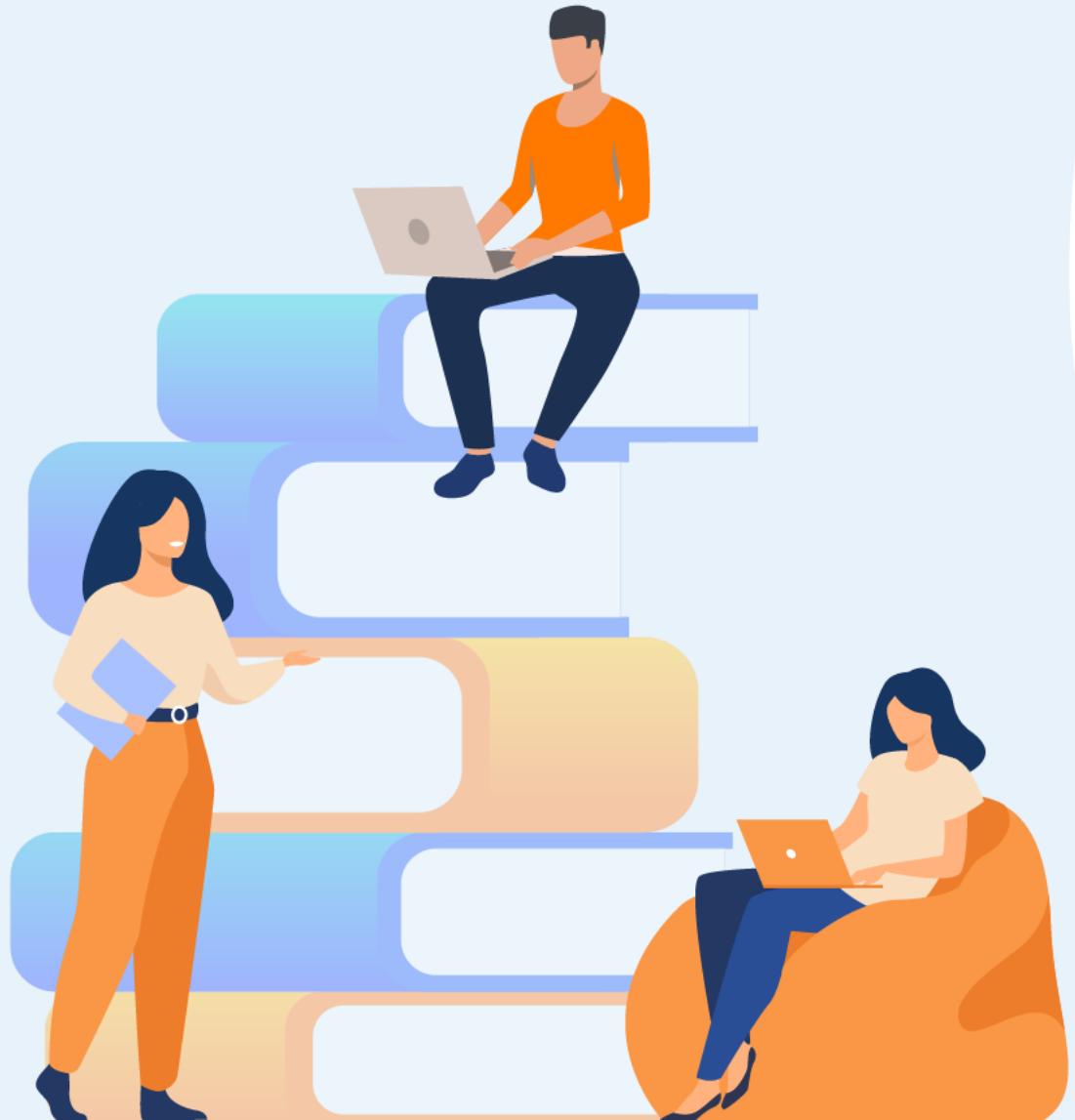
Règles de passage du MCD au MLD normalisé

Le MLD correspondant au MCD du centre de formation



Remarques :

- Deux nouvelles entités ajoutées : Inscription et Combinaison identifiées respectivement par les clés (codeSess,numCINETu) et (codeSpec,codeForm).
- L'association entre Session et Formation est traduite par l'ajout de la clé : codeForm dans l'entité Session.



PARTIE 2

Préparation de l'environnement

Dans ce module, vous allez :

- Apprendre la procédure d'installation d'un outil de modélisation
- Maitriser son utilisation et son exploitation
- Préparer le serveur MySQL





CHAPITRE 1

Exploiter un outil de modélisation

Ce que vous allez apprendre dans ce chapitre :

- Procédure d'installation d'un outil de modélisation
- Utilisation et exploitation de l'outil de modélisation





CHAPITRE 1

Exploiter un outil de modélisation

1. Procédure d'installation d'un outil de modélisation
2. Utilisation de l'outil de modélisation

01 - Exploiter un outil de modélisation

Introduction



- Dans la première partie de ce cours nous avons suivi les étapes de conception d'une base de données, depuis la lecture du cahier des charges jusqu'à l'élaboration des modèles conceptuel et logique de données (MCD et MLD).
- La modélisation a donc pour but de convertir la conception complexe en diagrammes représentants les données de manière simple et facile à comprendre. Ainsi, les outils de modélisation de données nous permettront de dessiner ces diagrammes et créer les structures logiques et physiques de la base de données.
- Il existe différents outils de modélisation souvent adaptés aux besoins de l'utilisateur en termes de systèmes d'exploitation, architecture et serveur de base de données. Parmi les fonctionnalités, on trouve : la création de structure de données à partir de diagrammes, l'ingénierie en amont et en aval, la fonction d'importation et d'exportation, la documentation, la prise en charge de plusieurs bases de données, le reporting, etc...
- Dans ce cours nous allons utiliser l'outil **MySQL Workbench**.

01 - Exploiter un outil de modélisation

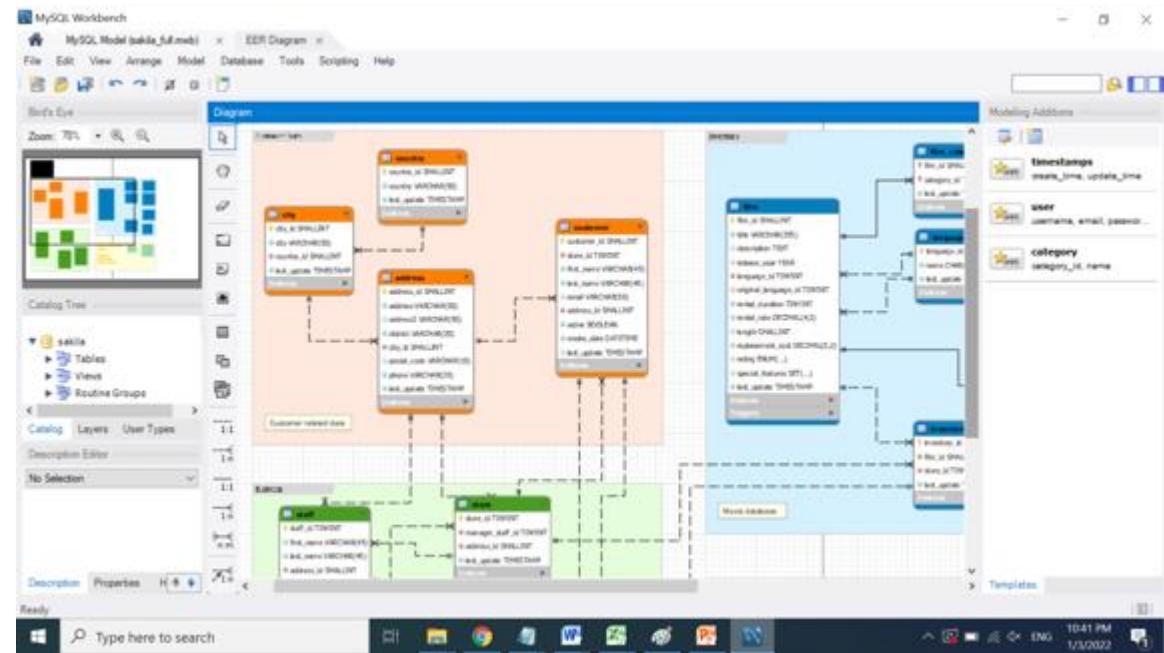
Procédure d'installation d'un outil de modélisation

MySQL Workbench

Il s'agit d'un outil utilisé par les administrateurs, les architectes et les développeurs de bases de données pour la modélisation des données, le développement SQL, la configuration du serveur, l'administration des utilisateurs et la sauvegarde.

Fonctionnalités :

- Modélisation et reverse engineering.
- Création des modèles ER complexes.
- La gestion des bases de données.
- La documentation.
- La création, exécution et optimisation des requêtes SQL via des outils visuels.
- Outils visuels pour la configuration des serveurs, la sauvegarde et la restauration, l'administration des utilisateurs, l'inspection des données d'audit et la visualisation de l'état de la base de données.



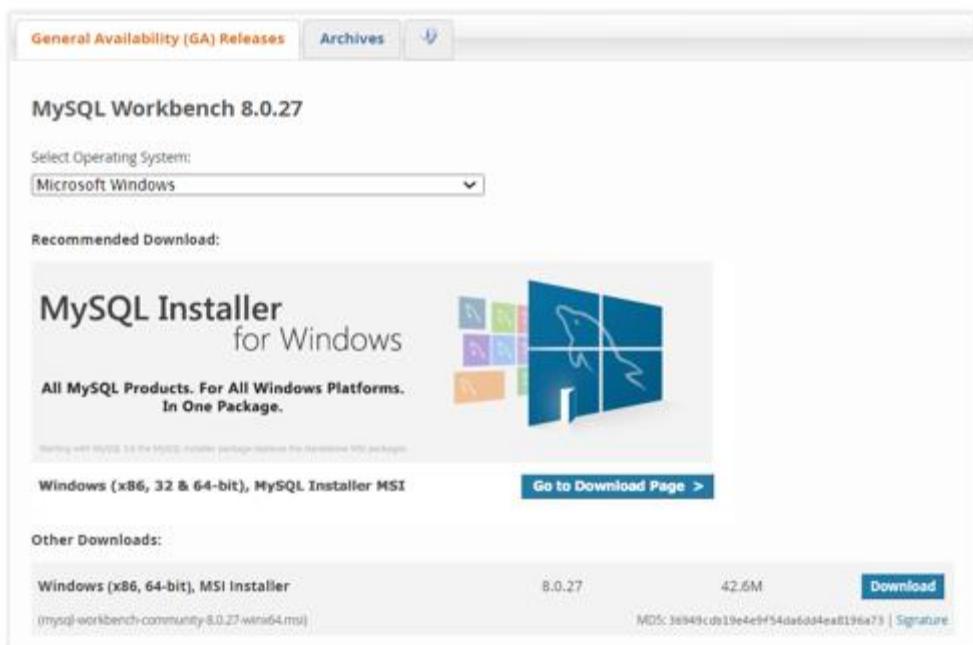
Design des diagrammes sur Workbench

01 - Exploiter un outil de modélisation

MySQL Workbench

Installation

Étape 1 : téléchargez l'outil sur le lien suivant : <https://dev.mysql.com/downloads/workbench/>



The screenshot shows the MySQL Workbench 8.0.27 download page. At the top, there are tabs for 'General Availability (GA) Releases' and 'Archives'. Below that, it says 'MySQL Workbench 8.0.27'. A dropdown menu 'Select Operating System:' is set to 'Microsoft Windows'. Under 'Recommended Download:', there is a large button for 'MySQL Installer for Windows' with the subtext 'All MySQL Products. For All Windows Platforms. In One Package.' and an image of the Windows logo. Below this, there's a note about starting with MySQL 8.0.27. Further down, there's a link 'Windows (x86, 32 & 64-bit), MySQL Installer MSI' and a 'Go to Download Page >' button. Under 'Other Downloads:', there's a link for 'Windows (x86, 64-bit), MSI Installer' with version 8.0.27, size 42.6M, and a 'Download' button.

Étape 2 : afin de commencer l'installation, lancez le fichier : mysql-workbench-community-8.0.27-winx64.msi

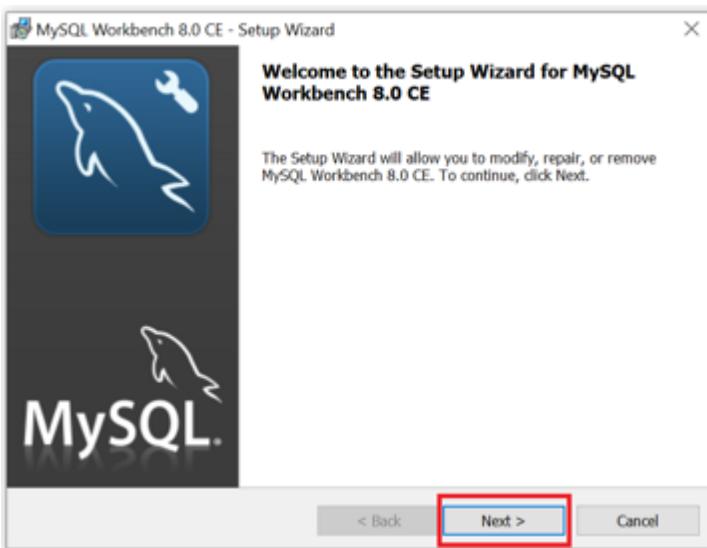
Name	Date modified	Type	Size
 mysql-workbench-community-8.0.27-winx64.msi	12/19/2021 12:33 PM	Windows Installer Pa...	43,636 KB

01 - Exploiter un outil de modélisation

MySQL Workbench

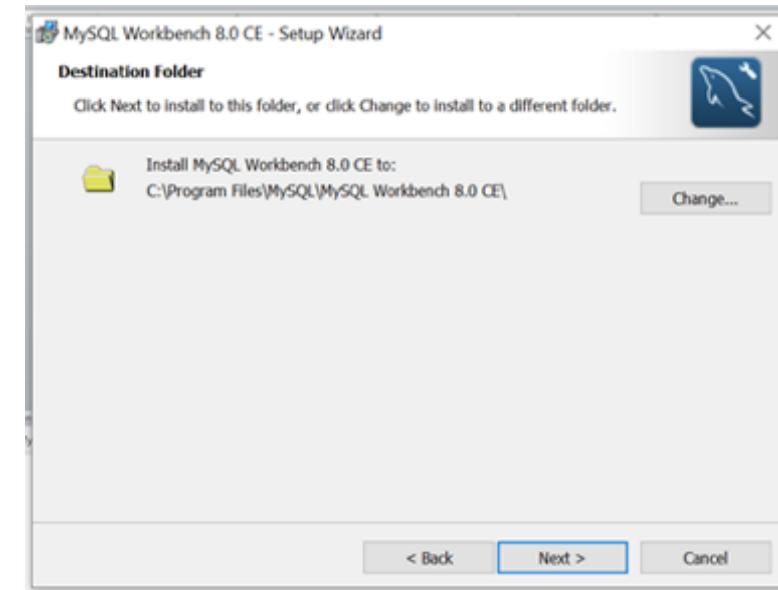
Installation

Étape 3 : une fois l'assistant de configuration lancé, cliquez sur « Next » :



Étape 4 : le système vous prompste à choisir le dossier où vous voulez installer Workbench.

Pour changer le dossier par défaut, on clique sur « Change ». Puis, on valide en cliquant sur « Next ».

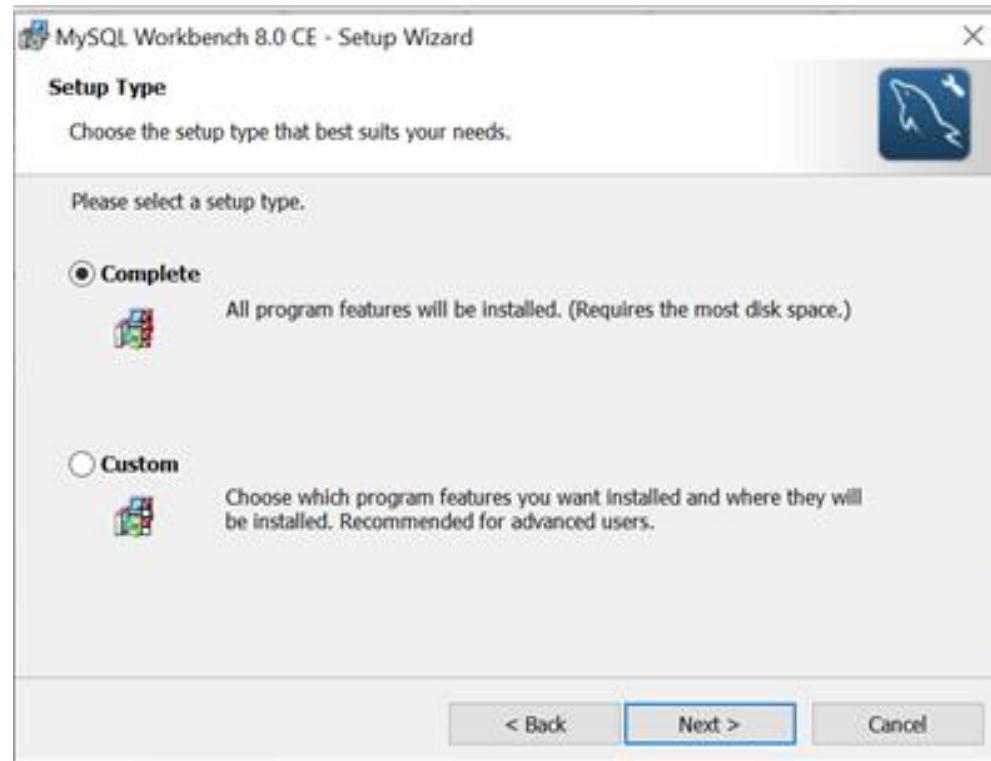


01 - Exploiter un outil de modélisation

MySQL Workbench

Installation

Étape 5 : on choisit le type d'installation (complète ou personnalisée), puis on clique sur « Next » :

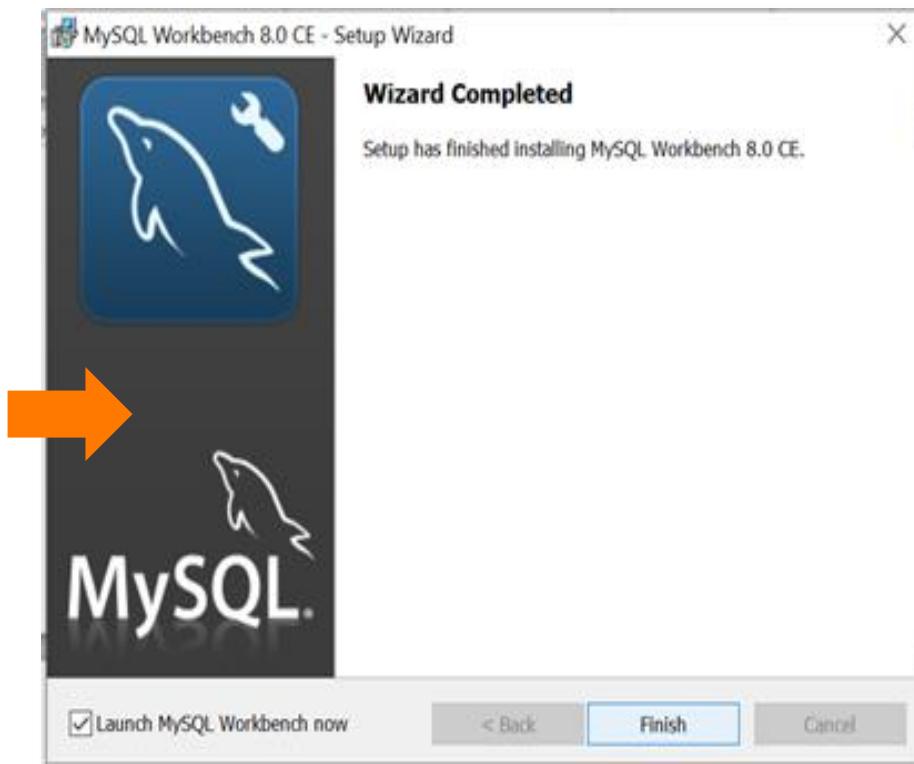
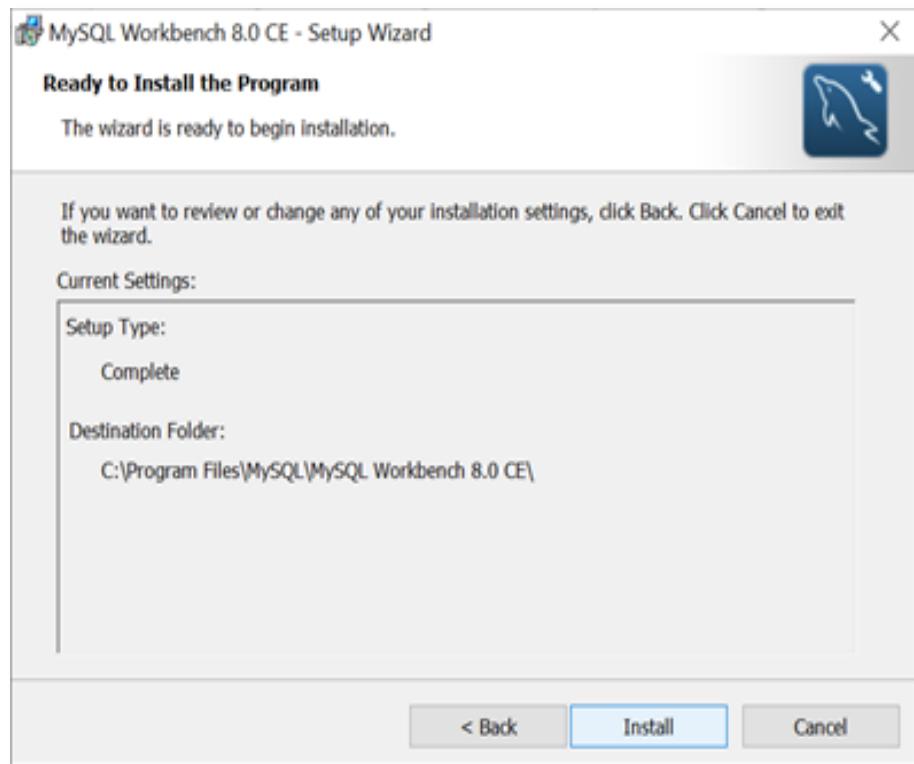


Exploiter un outil de modélisation

MySQL Workbench

Installation

Étape 6 : pour terminer, lancez l'installation :

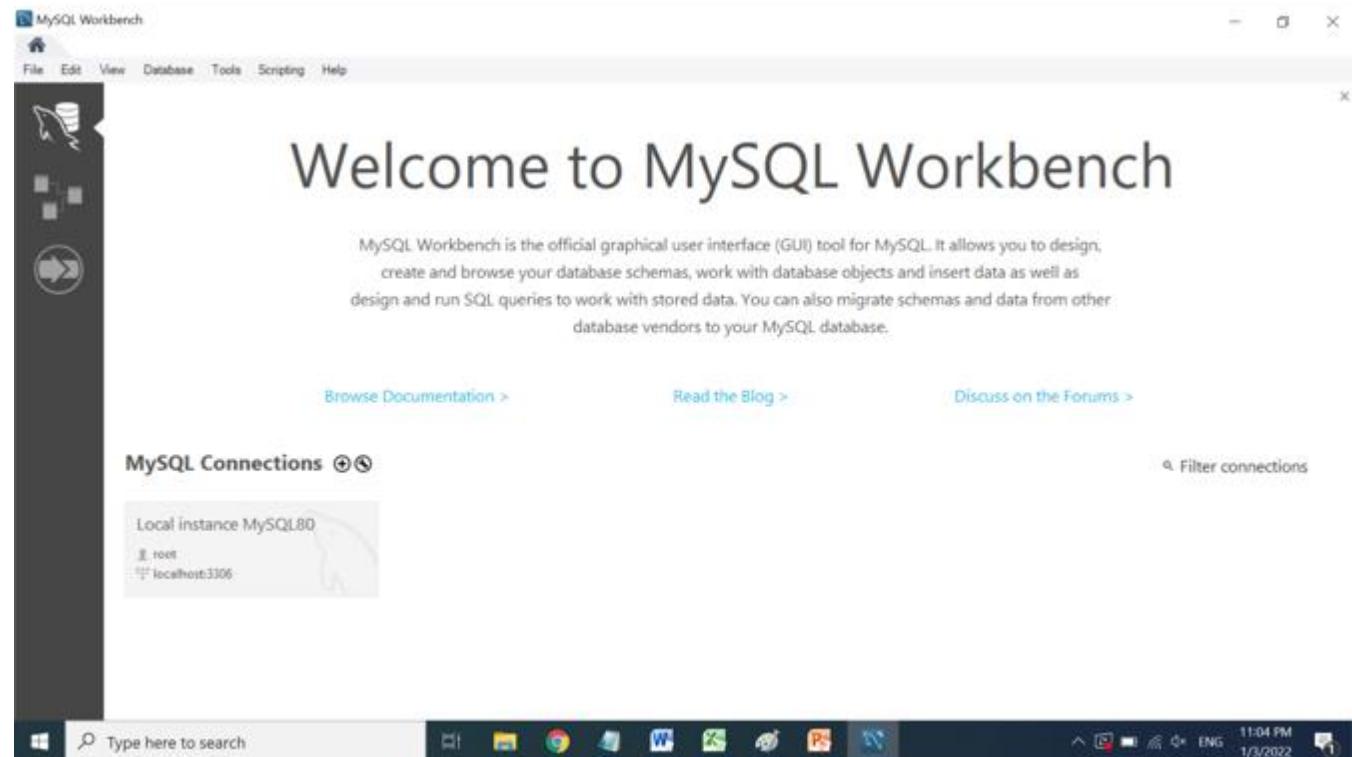


01 - Exploiter un outil de modélisation

MySQL Workbench

Installation

Une fois l'installation terminée, on peut accéder à la page d'accueil de Workbench :





CHAPITRE 1

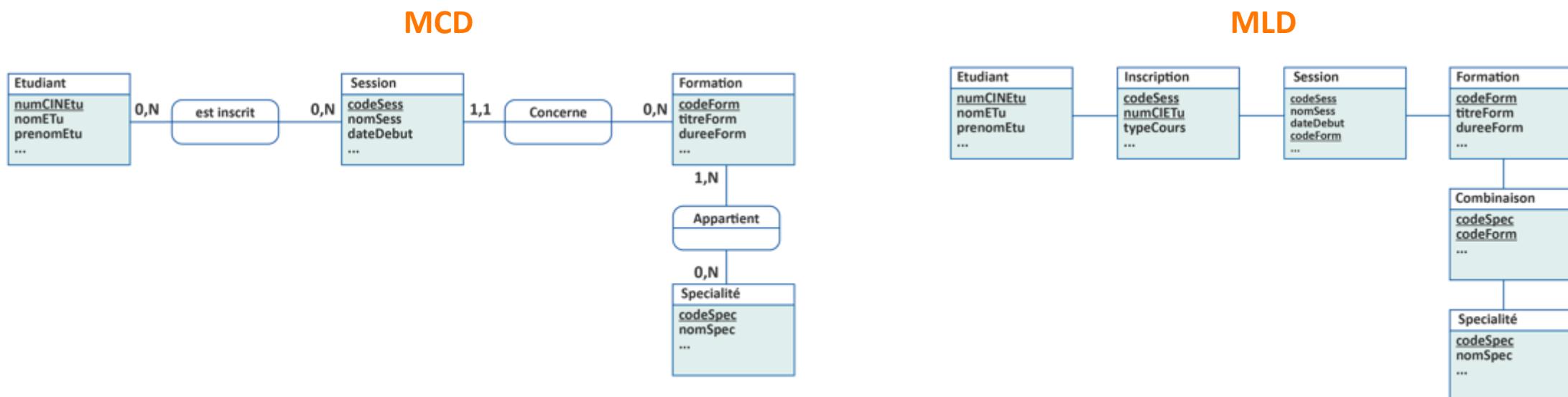
Exploiter un outil de modélisation

1. Procédure d'installation d'un outil de modélisation
2. **Utilisation de l'outil de modélisation**

01 - Exploiter un outil de modélisation

Utilisation de l'outil de modélisation

- Une des plus intéressantes fonctionnalités de l'outil MySQL Workbench est la possibilité de créer et gérer des modèles de données. Dans ce chapitre, nous allons suivre les étapes nécessaires pour concevoir un schéma simple à l'aide de MySQL Workbench. Ce schéma pourrait ensuite être exploité pour générer un script SQL et le transmettre à un serveur de base de données pour créer la base de données physique correspondante.
- Rappelons les MCD et MLD relatifs au centre de formation que nous avons vu dans le chapitre précédent :



01 - Exploiter un outil de modélisation

Utilisation de l'outil de modélisation

Créer un nouveau modèle

- Pour créer un nouveau modèle, démarrez l'outil MySQL Workbench et cliquez sur l'option « New model » située dans la colonne « Modélisation » des données de l'écran d'accueil.



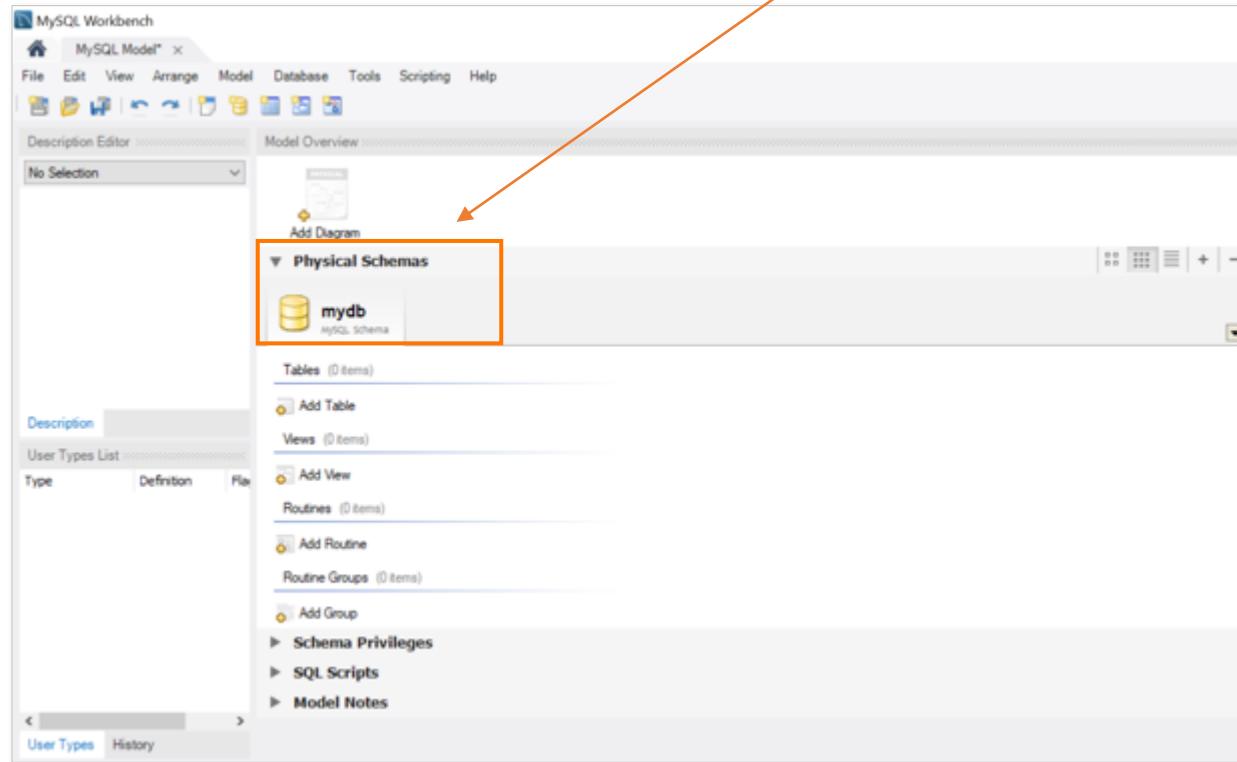
- Un nouveau panneau sera ajouté à l'atelier intitulé Modèle.

01 - Exploiter un outil de modélisation

Utilisation de l'outil de modélisation

Créer un schéma et ses objets

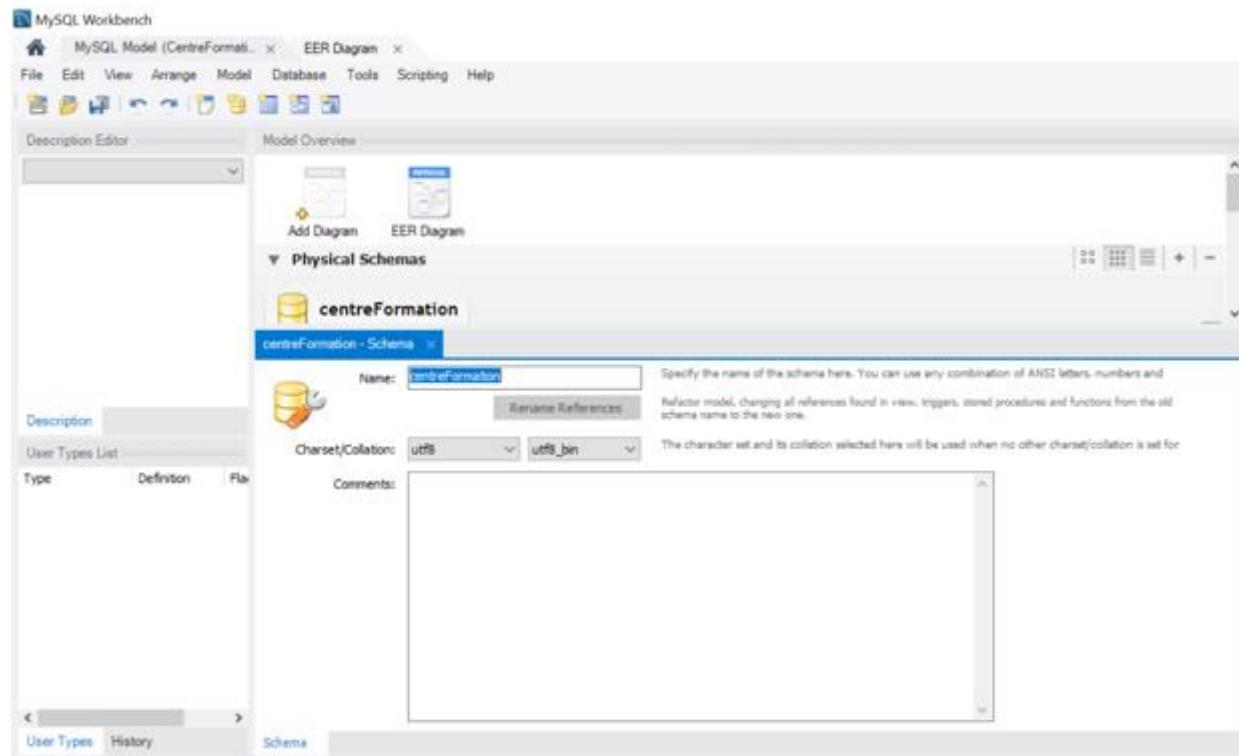
- On commence par définir les propriétés du schéma en double-cliquant sur l'onglet intitulé « mydb MySQL Schéma ».



01 - Exploiter un outil de modélisation

Utilisation de l'outil de modélisation

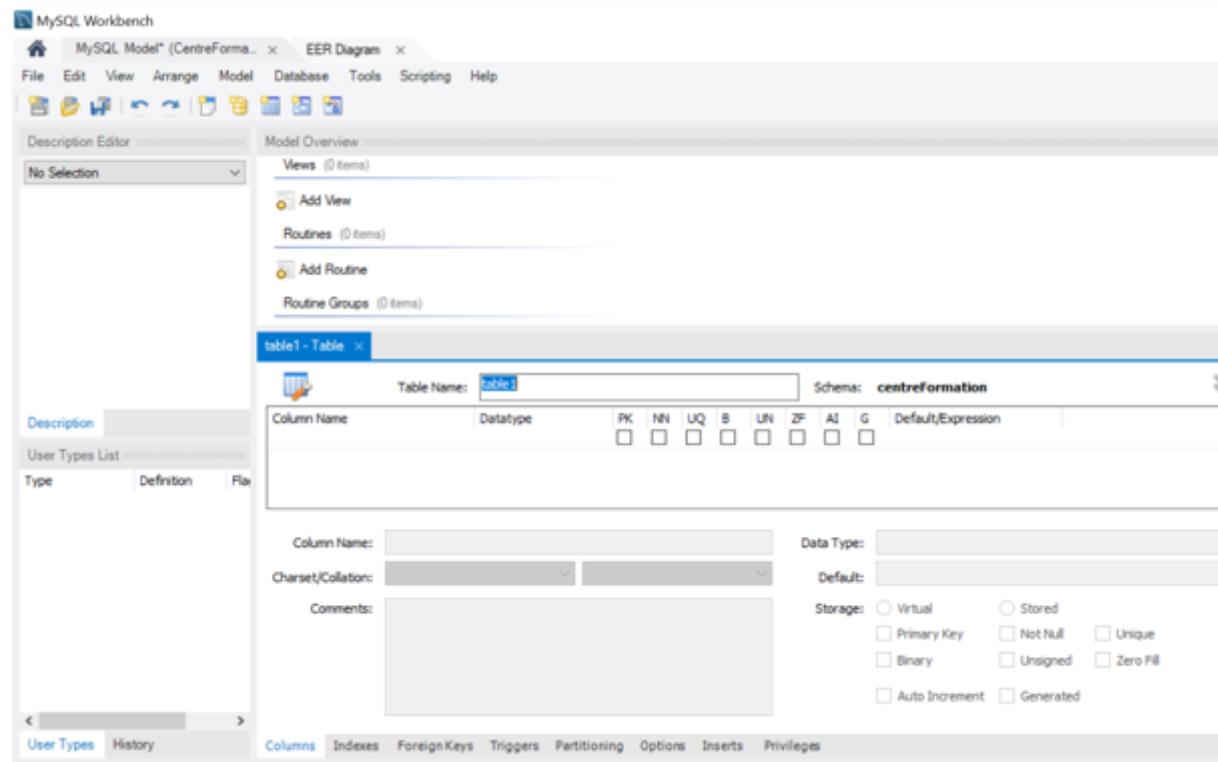
- Remplacez le nom du schéma comme il vous convient, ici on utilise le nom : « **centreFormation** ».
- Fermez le panneau des propriétés du schéma en cliquant sur le petit « x » à côté de l'onglet « Schéma ».



01 - Exploiter un outil de modélisation

Utilisation de l'outil de modélisation

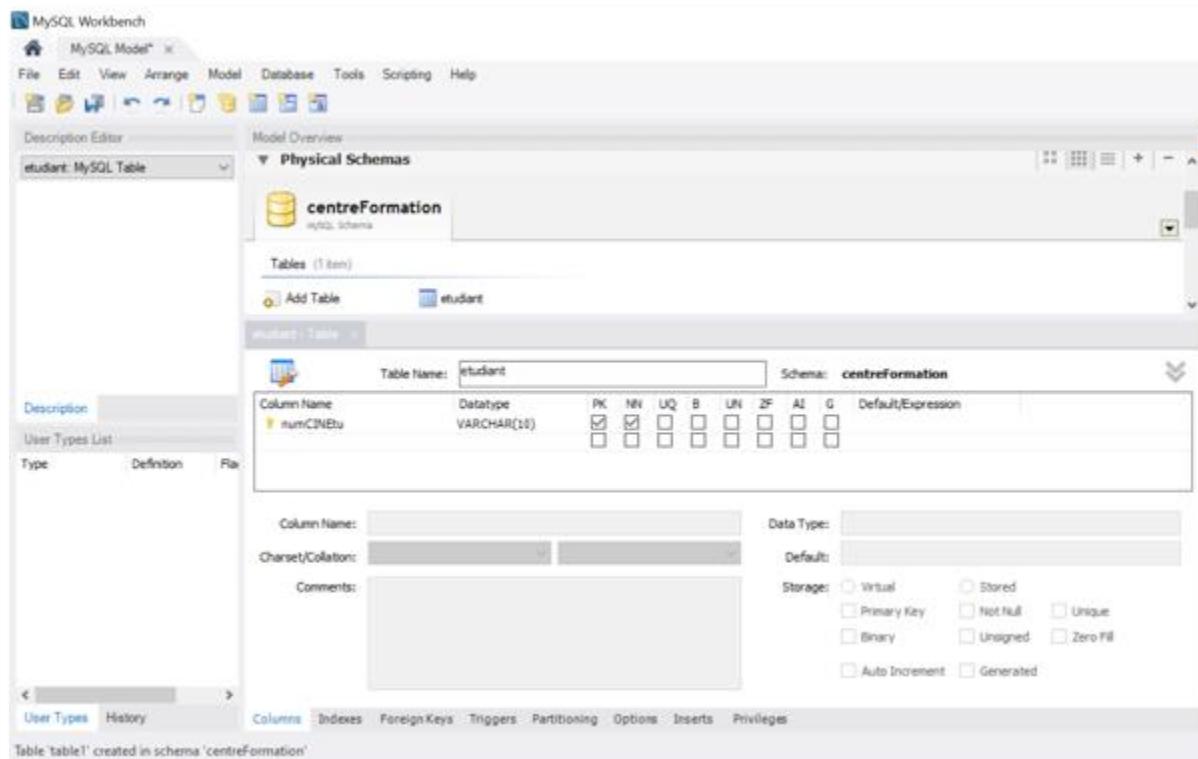
- Après avoir donné un nom à notre schéma, nous pouvons maintenant ajouter une table au modèle. Ceci est réalisé en double-cliquant sur le bouton « Ajouter une table » dans le panneau « Tables ».



01 - Exploiter un outil de modélisation

Utilisation de l'outil de modélisation

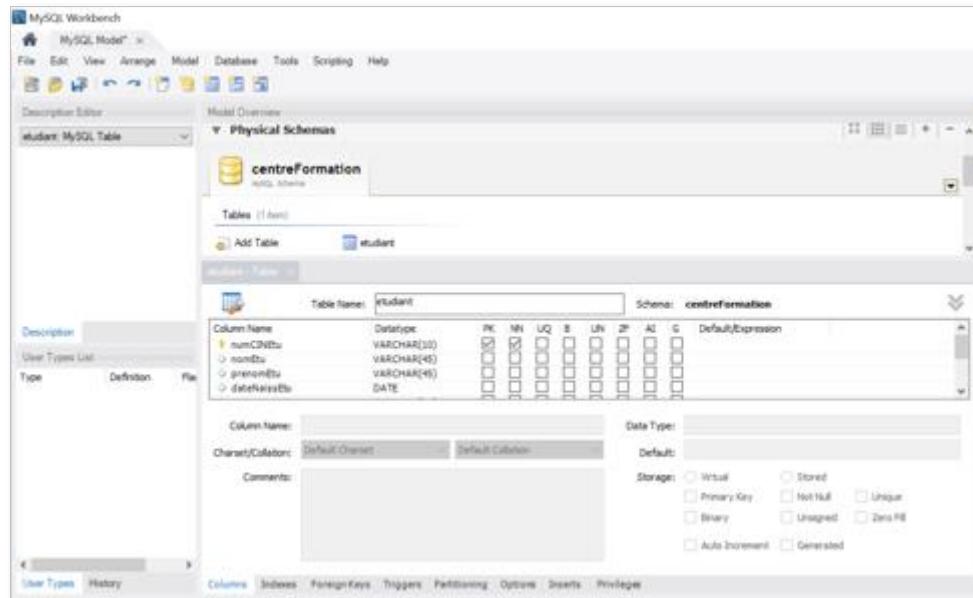
- Renommer la table « etudiant », puis sélectionnez l'onglet « Colonnes » pour commencer le processus d'ajout de colonnes à la table.
- Par défaut, le système va créer une clé primaire non nulle nommée idEtudiant. Renommez cette colonne : numCINETu. La colonne datatype définit le type de données, nous choisissons VARCHAR(10). Cochez la case PK (Primary key) comme c'est la clé primaire de la table, et aussi la case NN pour ne pas accepter des valeurs nulles dans ce champ.



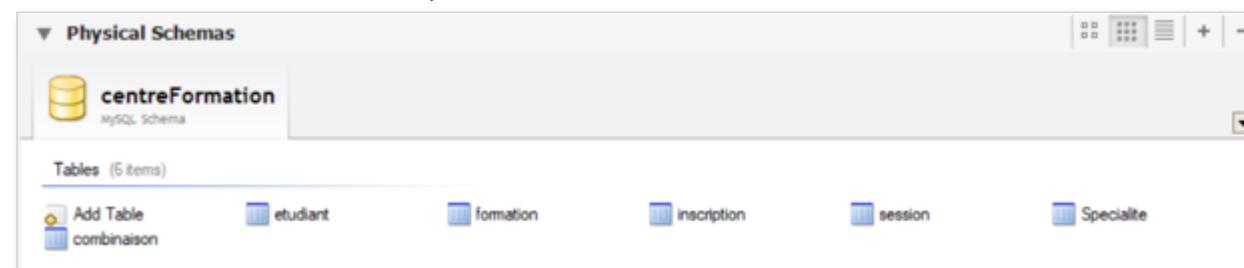
01 - Exploiter un outil de modélisation

Utilisation de l'outil de modélisation

- De la même manière, on continue à créer les autres colonnes de la table :



- Nous créons les autres tables du modèle en suivant les mêmes étapes :

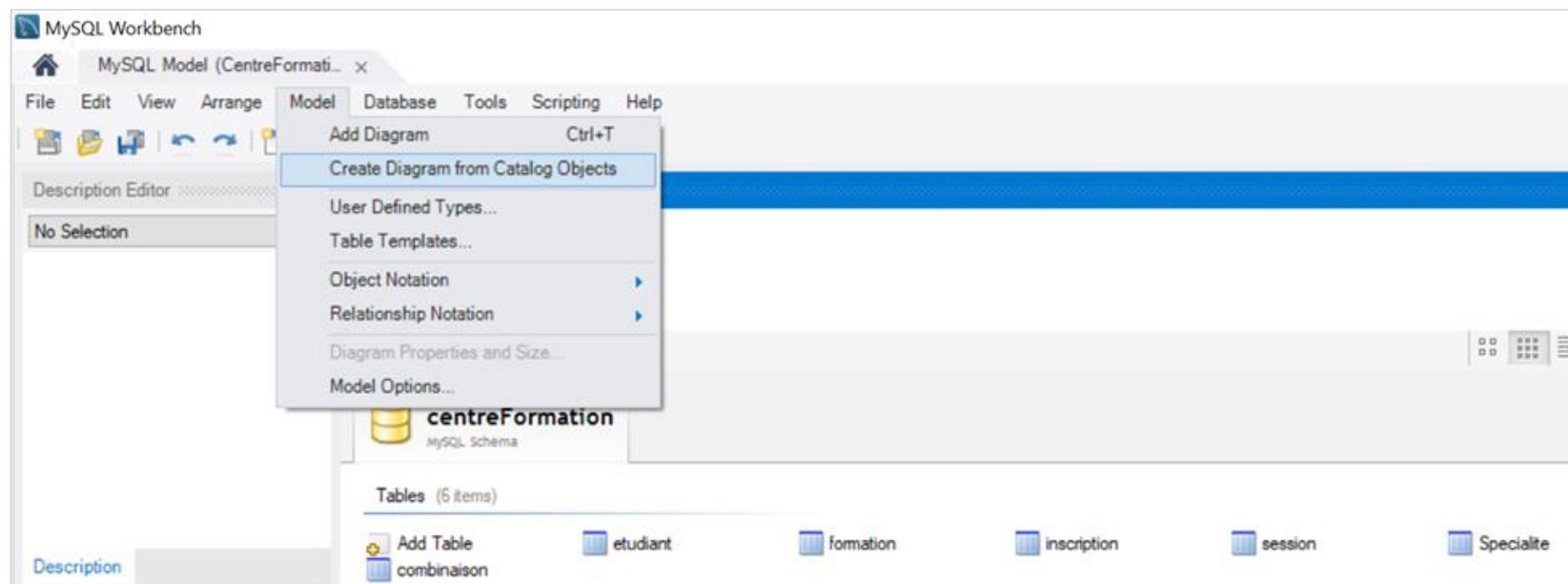


01 - Exploiter un outil de modélisation

Utilisation de l'outil de modélisation

Générer le diagramme

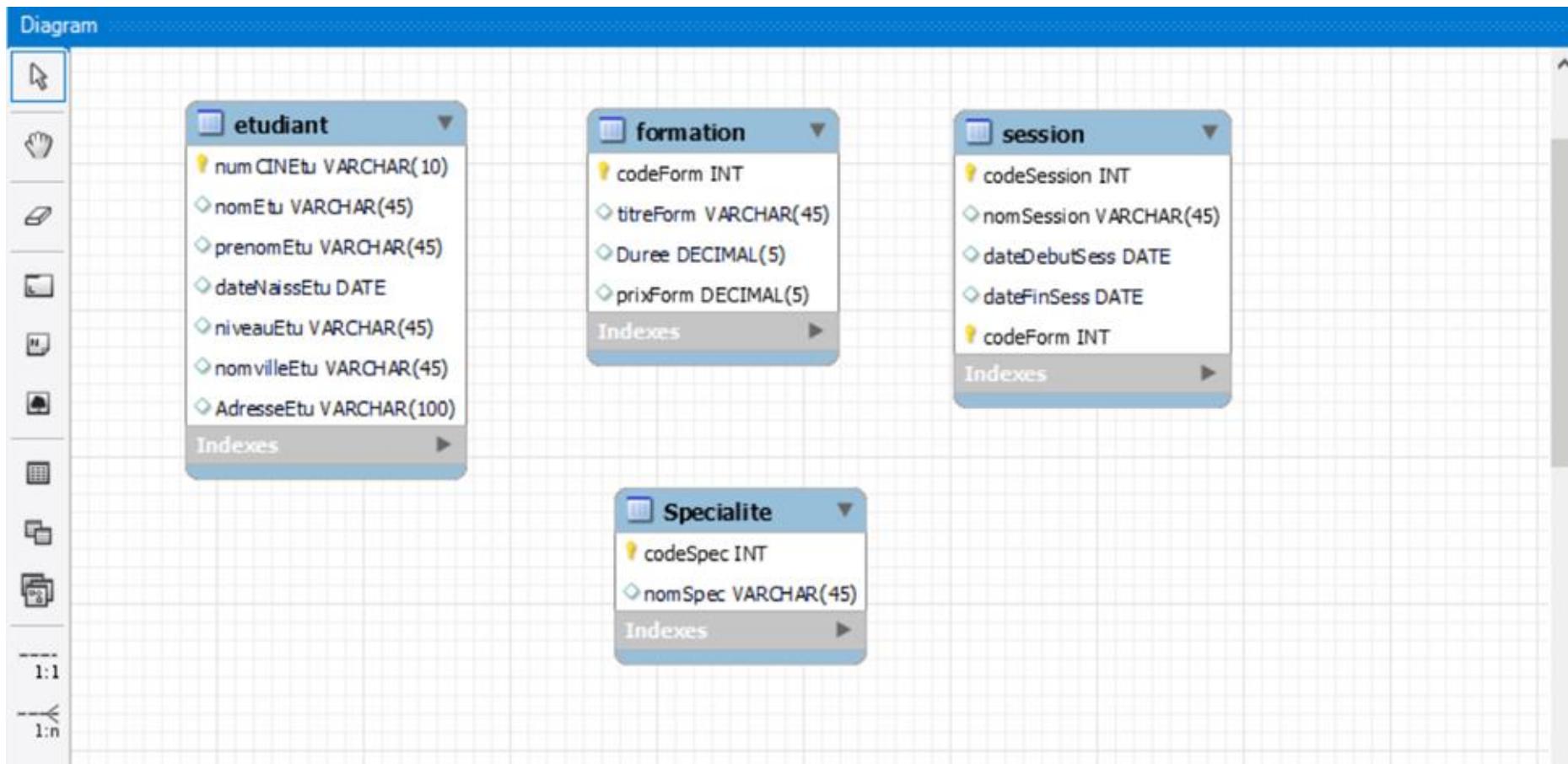
- Après avoir créé toutes les tables, avant de continuer, enregistrez le modèle à partir du menu Fichier -> Enregistrer le modèle ou à l'aide du bouton de barre d'outils approprié.
- Générez le modèle EER en naviguant vers le menu Model -> Create Model from catalogue objects :



01 - Exploiter un outil de modélisation

Utilisation de l'outil de modélisation

- Le diagramme illustré dans la figure suivante sera alors généré :



01 - Exploiter un outil de modélisation

Utilisation de l'outil de modélisation



Créer les associations entre les tables

- Nous passons à la création des relations entre les tables de notre modèle :
 - Les associations (**plusieurs à plusieurs**)
 - Les associations de **un à plusieurs**
- Pour ce, nous pouvons utiliser l'interface graphique du design ou en définissant les clés étrangères.

01 - Exploiter un outil de modélisation

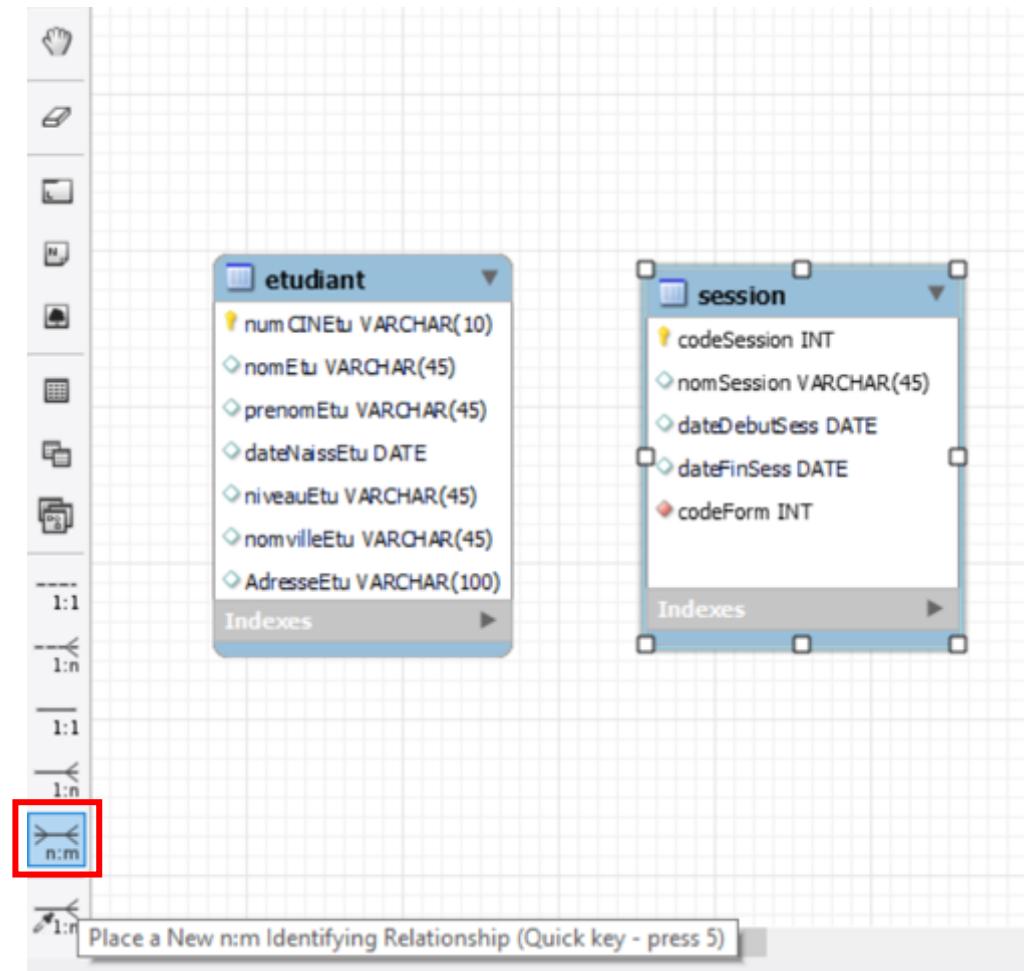
Utilisation de l'outil de modélisation

Les associations (plusieurs à plusieurs)

- La relation entre « Etudiant » et « Session » :
- D'après le MCD :



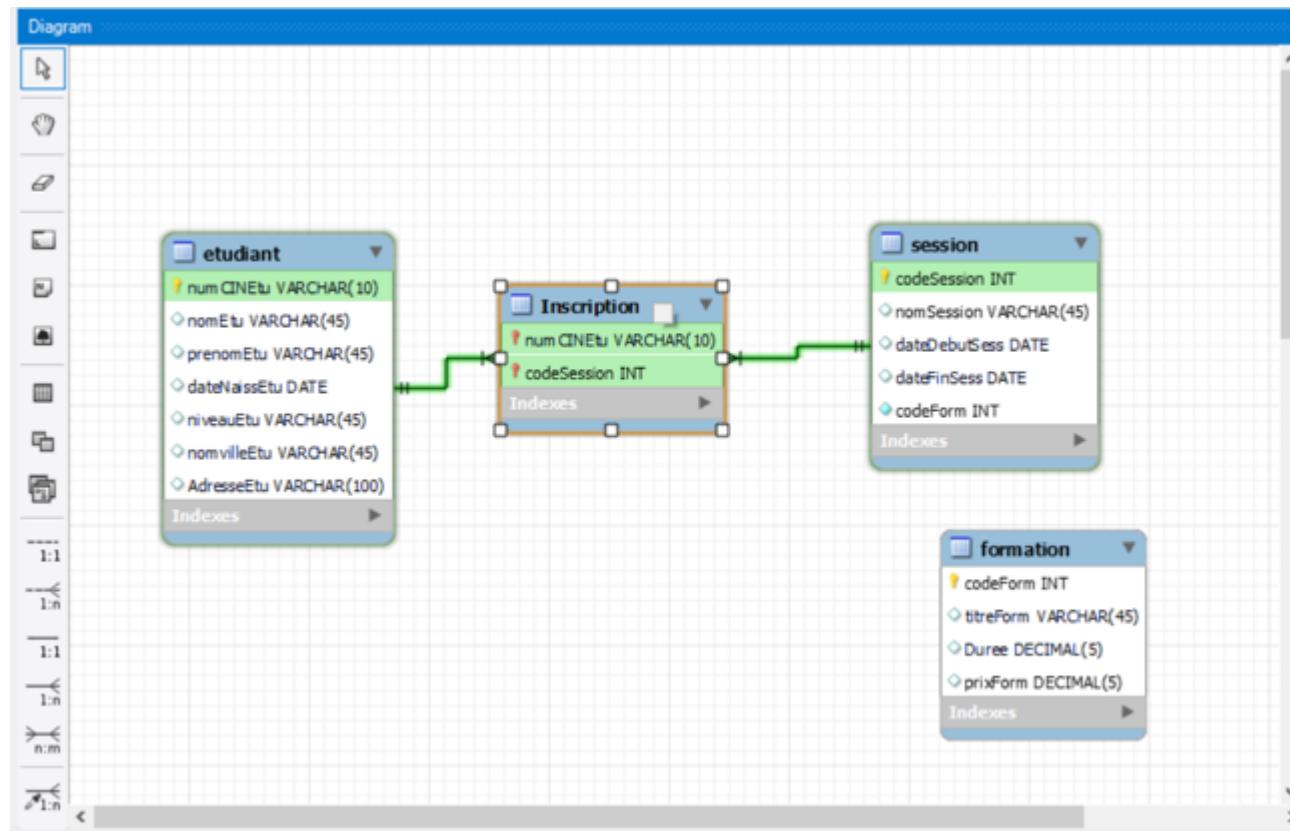
- Pour établir l'association, on clique sur l'icône « n:m Identifying Relationship », puis successivement sur les objets « etudiant » et « session ».



01 - Exploiter un outil de modélisation

Utilisation de l'outil de modélisation

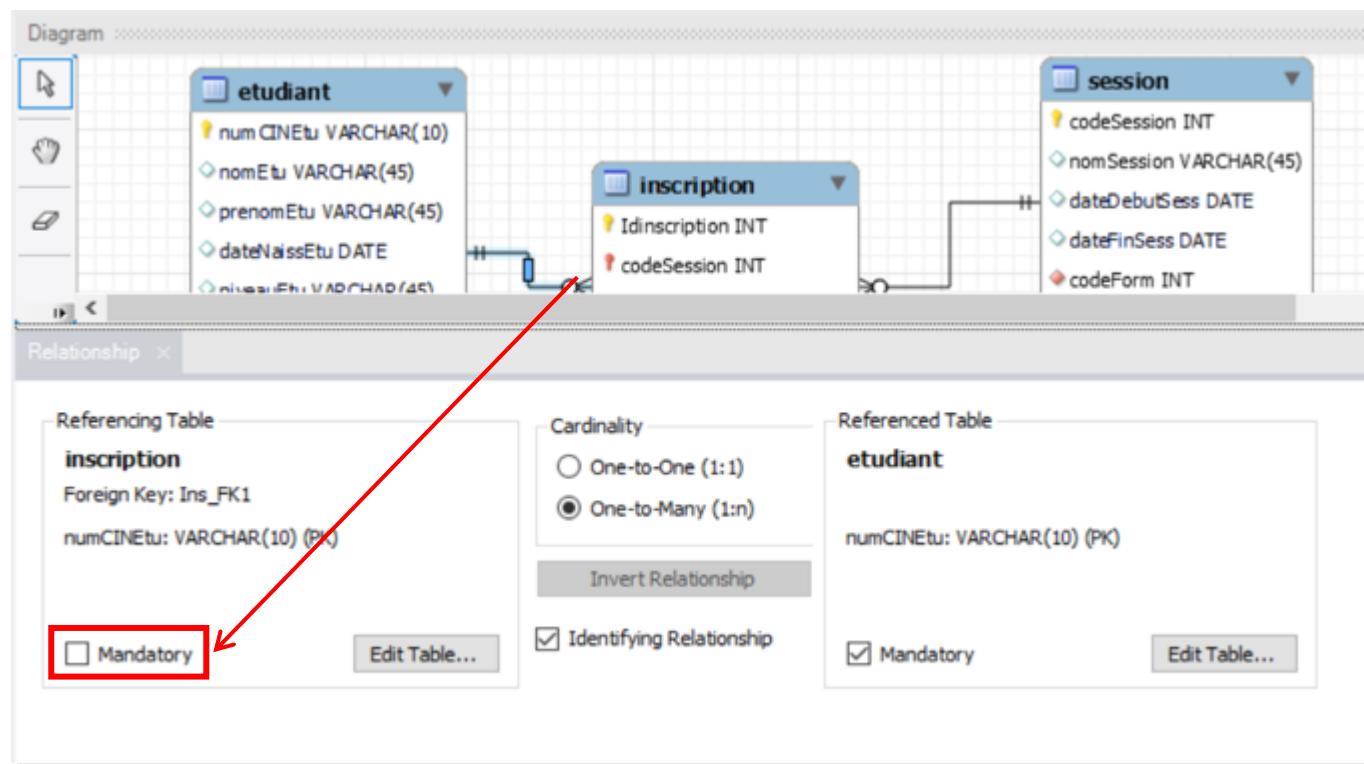
- L'outil crée automatiquement une table d'association ETUDIANT_has_SESSION qu'on peut renommer : « **Inscription** ».
- On remarque que le système a déjà créé une clé primaire (numCINEtu, codeSession).



01 - Exploiter un outil de modélisation

Utilisation de l'outil de modélisation

- Afin de respecter les règles de gestions selon lesquelles un étudiant peut n'être inscrit en aucune session, et une session peut n'avoir aucun étudiant y inscrit.
- On ajuste les cardinalités (0,N) pour les tables « etudiant » et « session », on double-clique sur la patte connectant « inscription » et « etudiant », puis on décoche la case « Mandatory » associée à « inscription » (Referencing Table) :

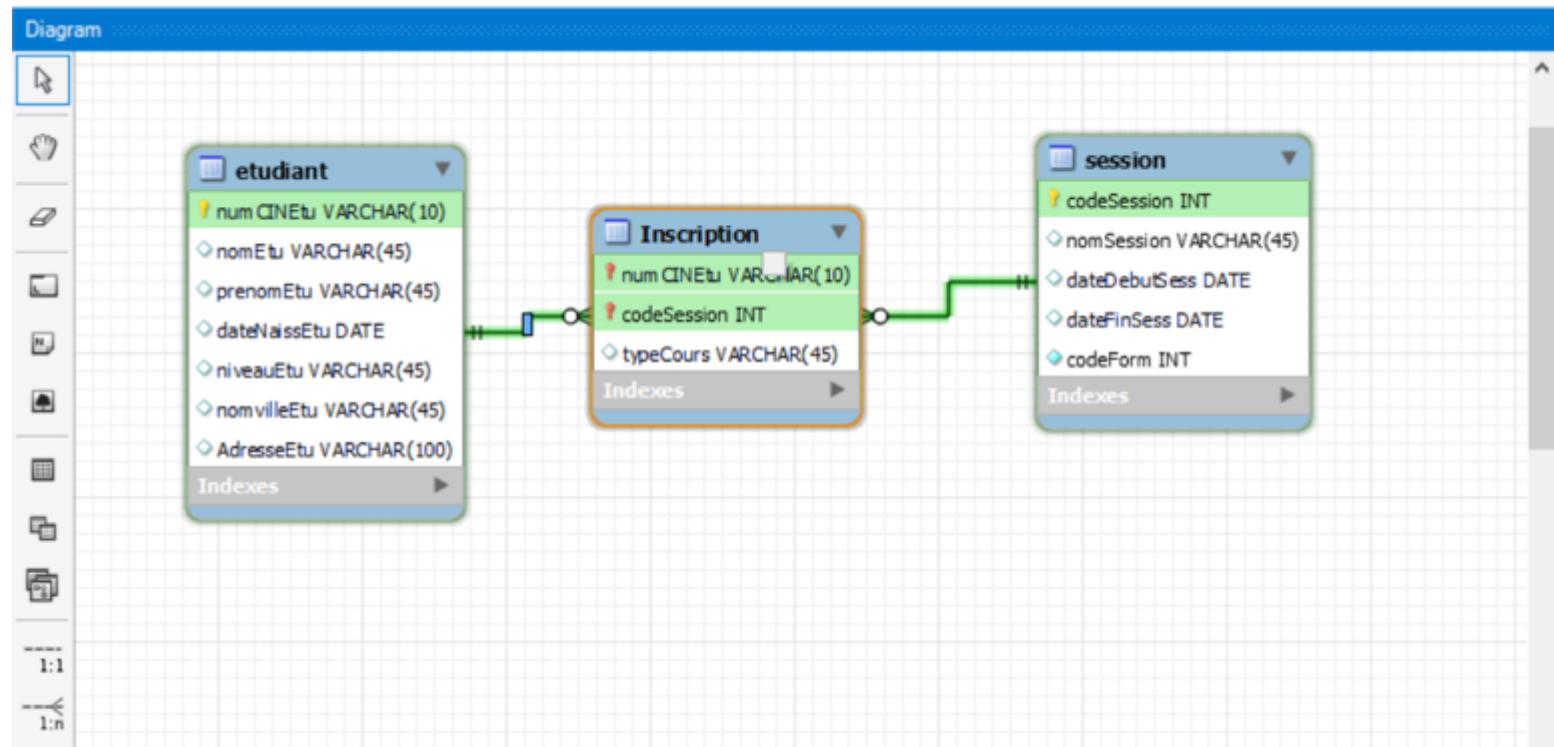


01 - Exploiter un outil de modélisation

Utilisation de l'outil de modélisation

Créer les associations entre les tables

- On refait le même exercice pour la table « **session** ». Et on ajoute la colonne : TypeCours qui est un attribut de la relation : inscription.
- Notre diagramme prend alors cette forme :

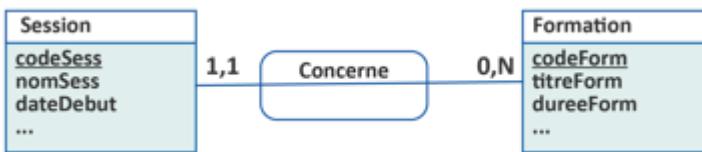


01 - Exploiter un outil de modélisation

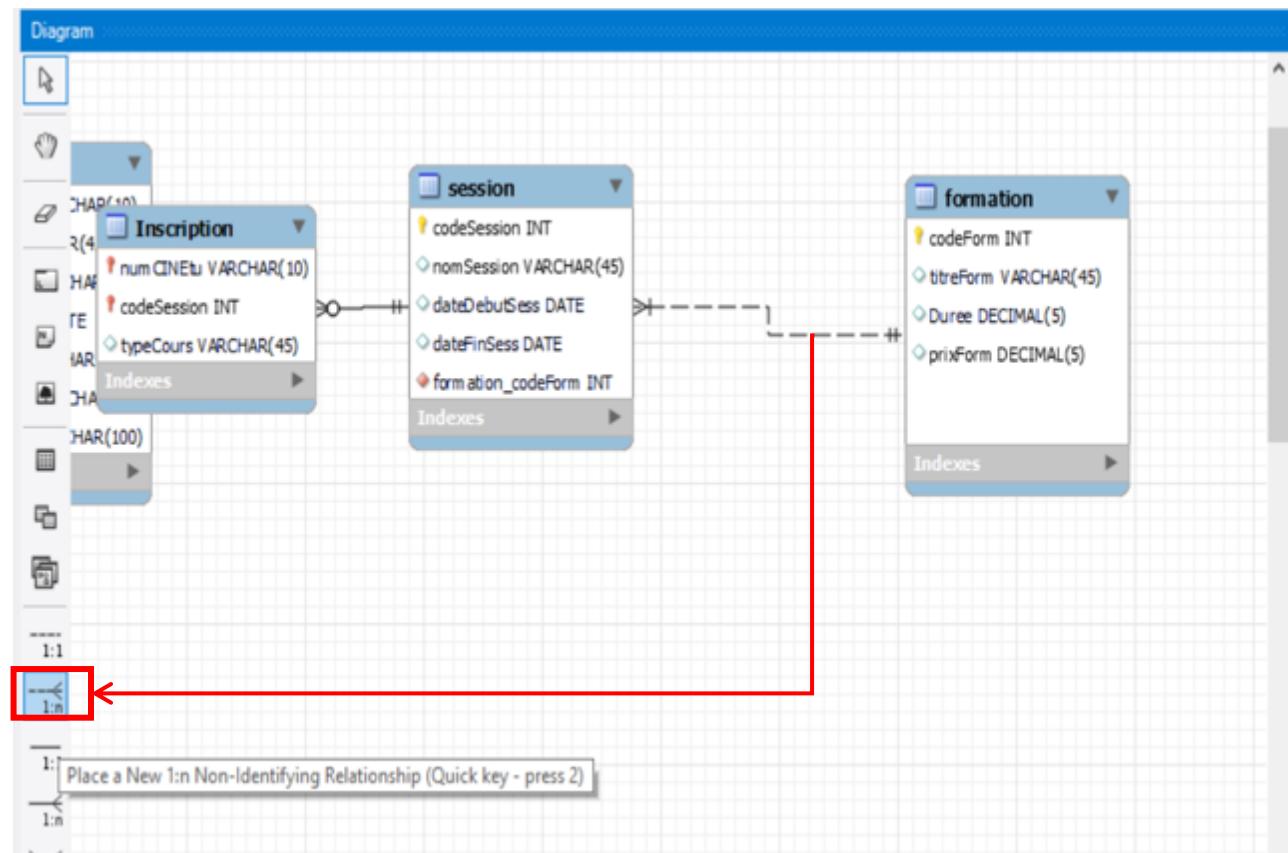
Utilisation de l'outil de modélisation

Les associations de un à plusieurs

- La relation entre Formation et Session :
- D'après le MCD :



- On sélectionne l'icône « 1:n Non-Identifying Relationship », puis on clique successivement sur les objets « session » et « formation » (dans cet ordre, c'est-à-dire le référencant d'abord, puis le référencé). Au résultat :

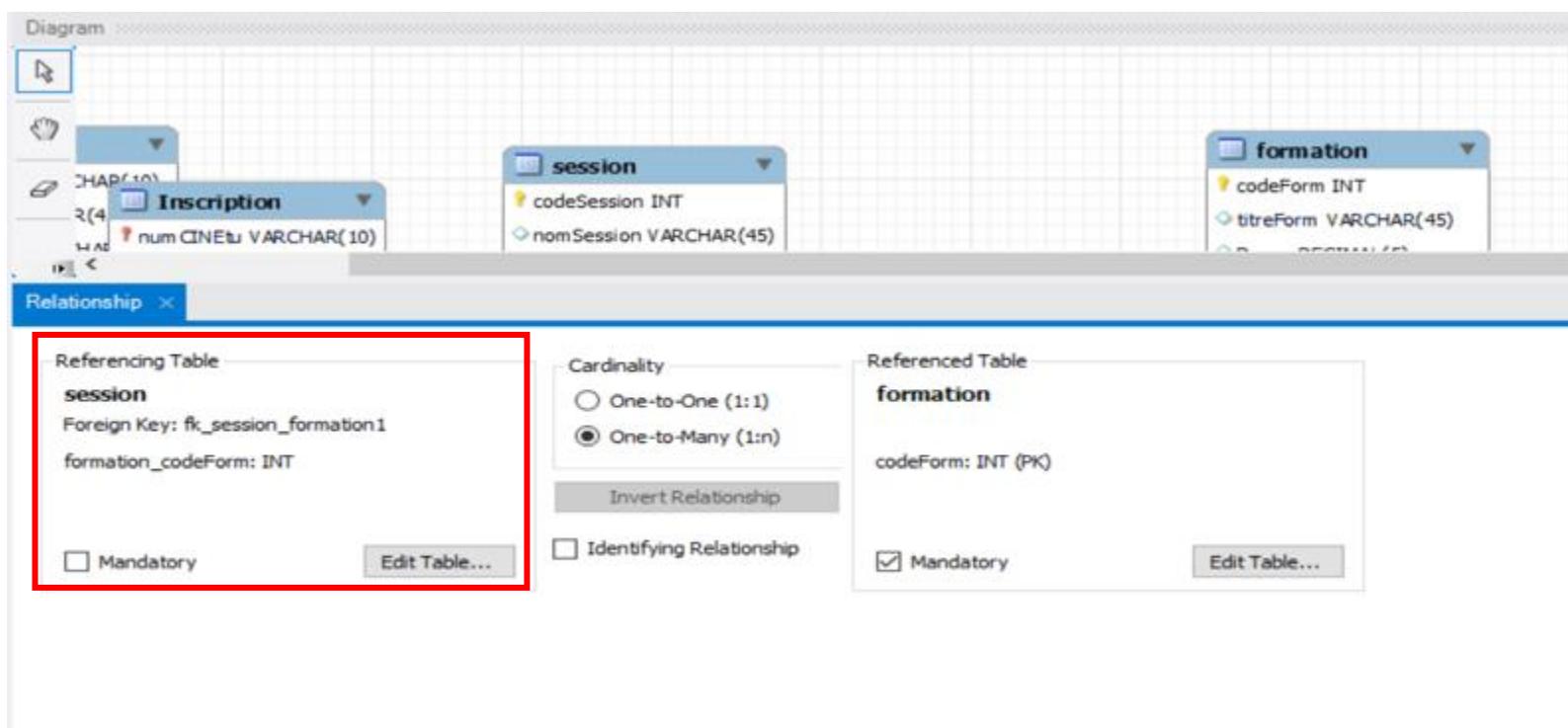


01 - Exploiter un outil de modélisation

Utilisation de l'outil de modélisation

Créer les associations entre les tables

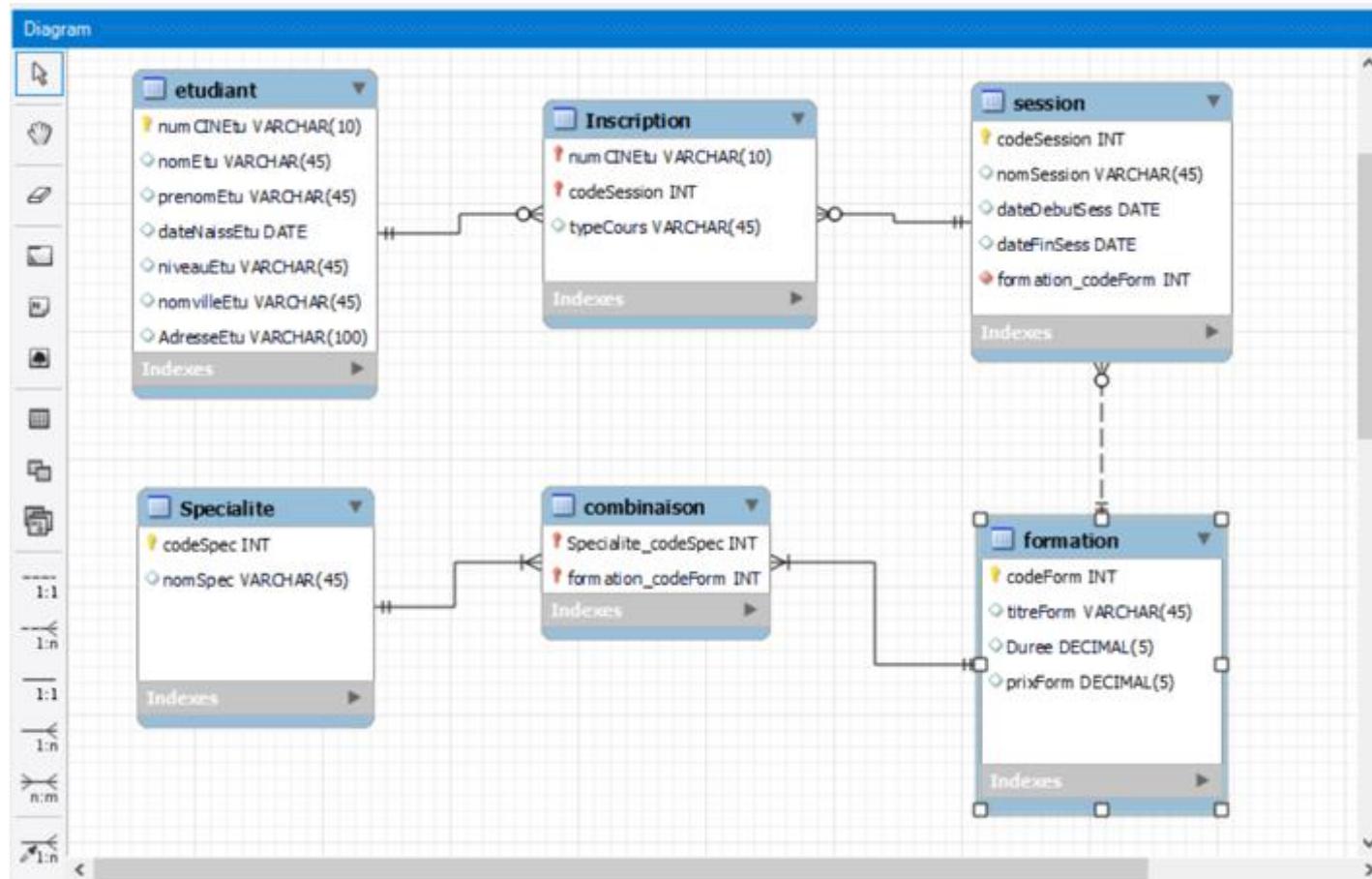
- On remarque qu'une colonne : formation_codeForm a été ajoutée à la table « session ». Il s'agit de la clé étrangère qui référence la table « formation » dans la table « session ».
- Aussi, la patte entre « session » et « formation » est en pointillés : l'outil indique ainsi que « session » n'est pas une propriété (multivaluée) de « formation ».
- Afin de respecter la cardinalité 0,N de la table « formation », il faut décocher la case « Mandatory » du côté « session » (Referencing Table) :



01 - Exploiter un outil de modélisation

Utilisation de l'outil de modélisation

- Après avoir créé toutes les associations du MCD, voici le modèle final :

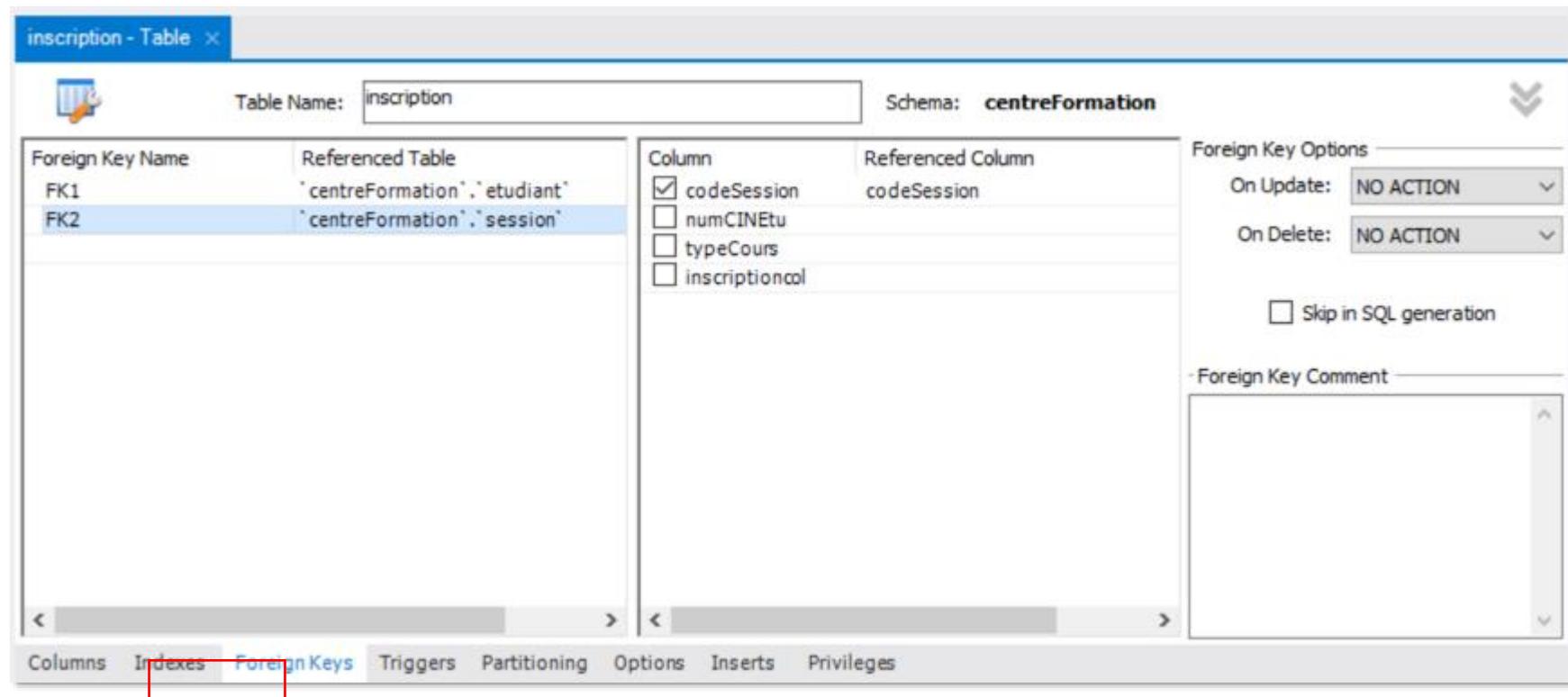


01 - Exploiter un outil de modélisation

Utilisation de l'outil de modélisation

En utilisant les clés étrangères

- Nous pouvons générer les relations automatiquement après avoir ajouté les clés étrangères sur toutes les tables concernées du schéma.
- Pour la table « inscription », par exemple, en navigant sur la tab Foreign Key, nous définissons 2 clés étrangères qui réfèrent aux tables : « etudiant » et « session ».



The screenshot shows the MySQL Workbench interface for the 'inscription' table. The 'Foreign Keys' tab is selected, indicated by a red box at the bottom. The table name is 'inscription' and the schema is 'centreFormation'. Two foreign keys are defined:

Foreign Key Name	Referenced Table
FK1	'centreFormation', 'etudiant'
FK2	'centreFormation', 'session'

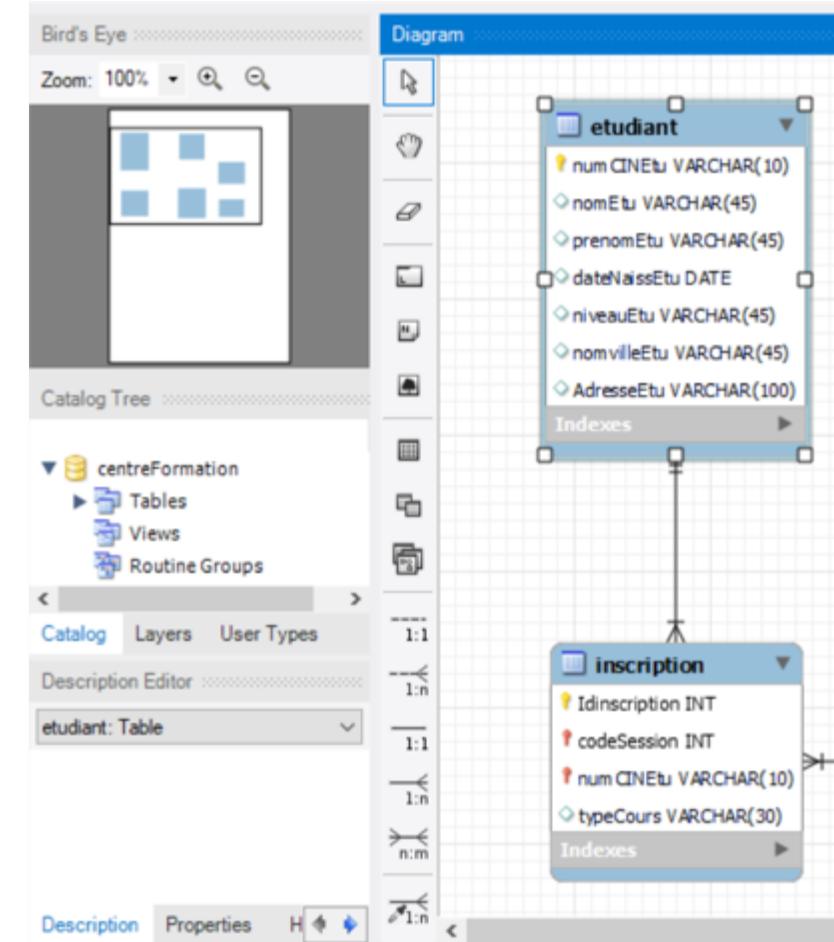
For FK1, the column 'codeSession' is mapped to the referenced column 'codeSession' in the 'etudiant' table. For FK2, columns 'numCINEtu', 'typeCours', and 'inscriptioncol' are mapped to their respective referenced columns in the 'session' table. The 'On Update' and 'On Delete' options are both set to 'NO ACTION'. There is also a checkbox for 'Skip in SQL generation' which is unchecked. A comment section for the foreign key is present but empty.

01 - Exploiter un outil de modélisation

Utilisation de l'outil de modélisation

Créer les associations entre les tables

- Si on génère le diagramme, on remarque que l'outil a déjà créé cette relation :



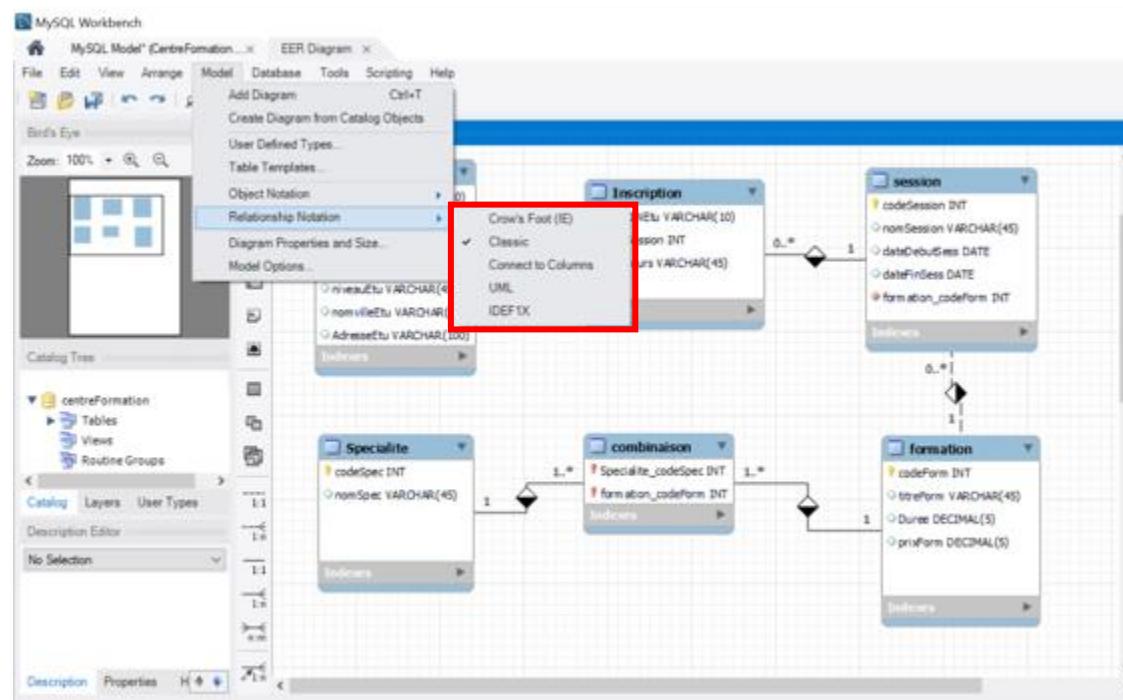
01 - Exploiter un outil de modélisation

Utilisation de l'outil de modélisation

Créer les associations entre les tables

Autres fonctionnalités :

- L'outil offre d'autres fonctionnalités graphiques qui facilitent la lecture du diagramme, par exemple :
 - **Les notations des associations** : afin de refléter la méthode de modélisation utilisée, les notations des associations personnalisables à partir du menu Model -> Relationship Notation.



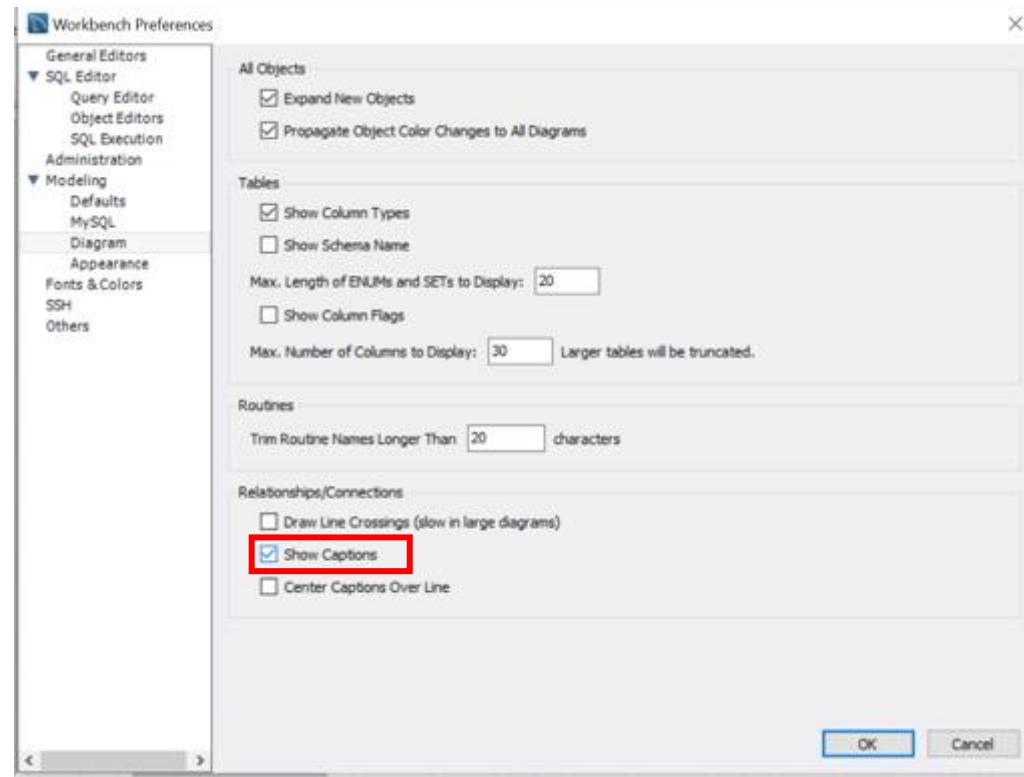
01 - Exploiter un outil de modélisation

Utilisation de l'outil de modélisation

Créer les associations entre les tables

Afficher les noms de relations entre les tables :

- Depuis le menu, selectionner Edit -> Preferences -> Diagram, puis cocher : Show caption.





CHAPITRE 2



Ce que vous allez apprendre dans ce chapitre :





CHAPITRE 2

Préparer le serveur MySQL

1. Installation de **Workbench** serveur MySQL
2. Management des services MySQL
3. Configuration des ports MySQL

02 - Préparer le serveur MySQL

Installation de MySQL Server

- MySQL est un serveur de gestion de base de données relationnelle (SGBDR) open source et multiplateforme développé par la société suédoise « MySQL AB » et acquis plus tard par Oracle Corporation.



(MySQL logo)

- Dans le reste de cette partie du cours, nous allons installer et configurer l'édition du serveur MySQL Community.
- Le version MySQL community server 8.0 peut être téléchargée sur le lien suivant :
<https://dev.mysql.com/get/Downloads/MySQLInstaller/mysql-installer-community-8.0.19.0.msi>
- Cette version comporte aussi l'outil Workbench que nous avons utilisé pour la modélisation des bases de données.
- Workbench sert aussi à assurer l'administration, la gestion, la maintenance, la sécurité et les différentes tâches de développement des base de données.

02 - Préparer le serveur MySQL

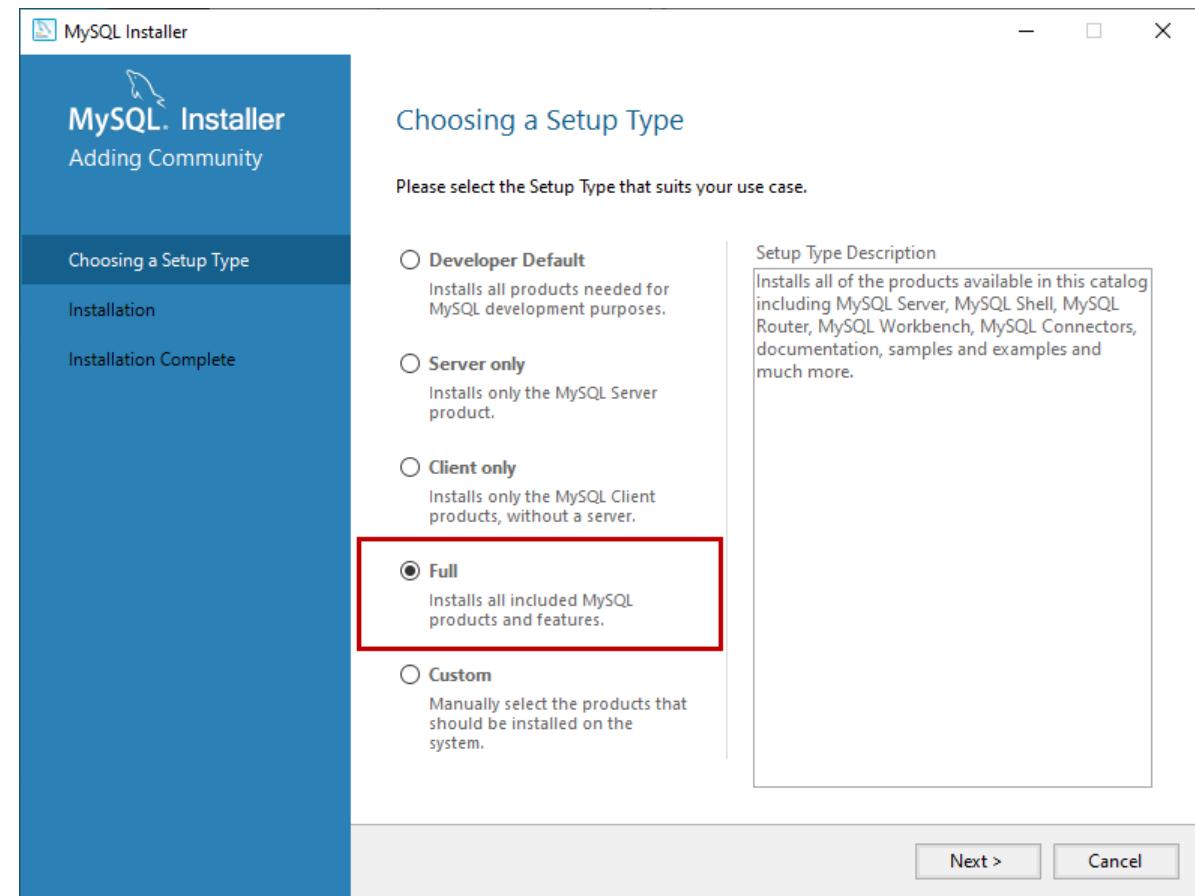
Installation de MySQL Server

- Une fois le téléchargement terminé, lancez le fichier d'installation. Le programme nous prompte à choisir un type d'installation :
 - **Developer default** : si vous souhaitez créer une machine de développement, vous pouvez utiliser cette option. Elle comporte les composants requis pour le développement d'applications, par exemple : MySQL Server, MySQL Shell, MySQL connectors, MySQL.
 - **Server only** : si vous souhaitez créer un serveur de base de données autonome avec des composants spécifiques.
 - **Full** : si vous souhaitez installer MySQL Server avec tous ses composants.
 - **Custom** : si vos besoins se limitent à quelques composants, vous pouvez utiliser cette option.

02 - Préparer le serveur MySQL

Installation de MySQL Server

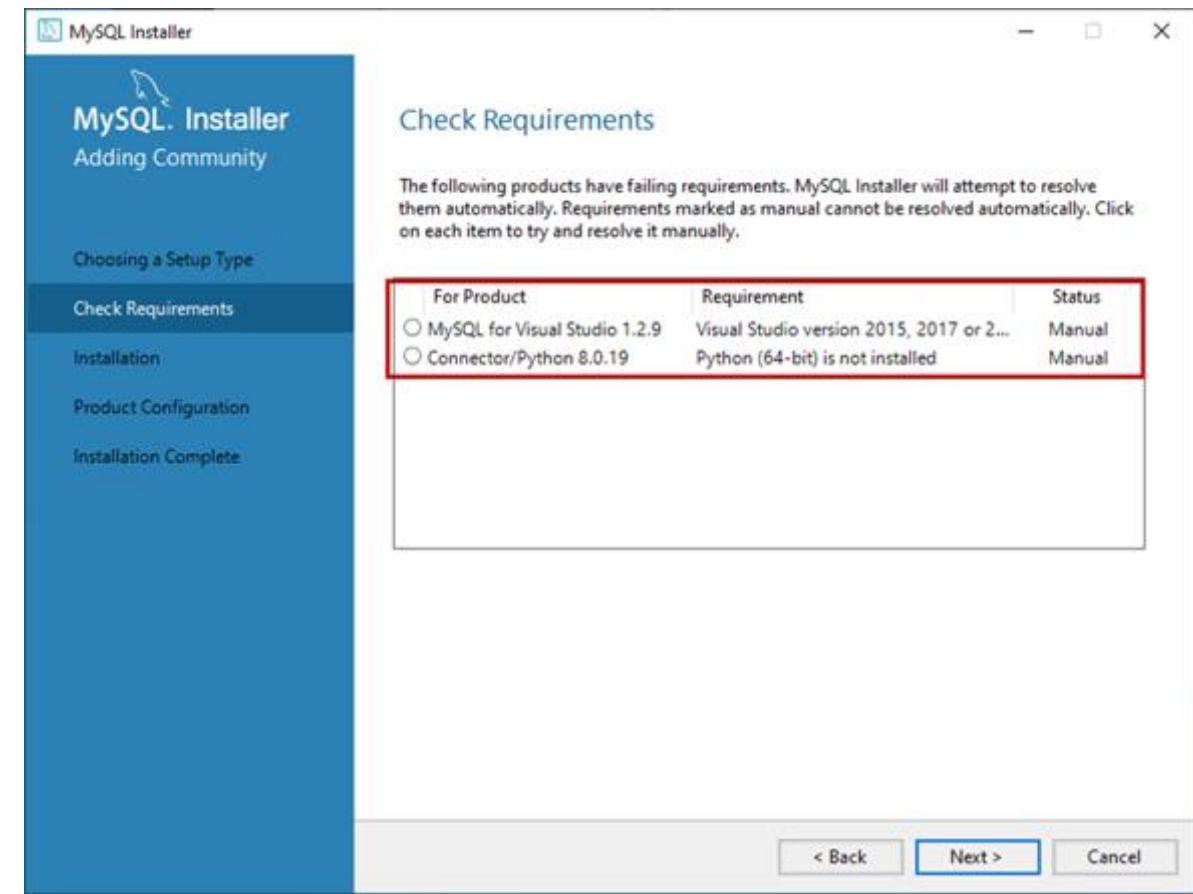
- Nous allons continuer avec l'installation « **Full** ».



02 - Préparer le serveur MySQL

Installation de MySQL Server

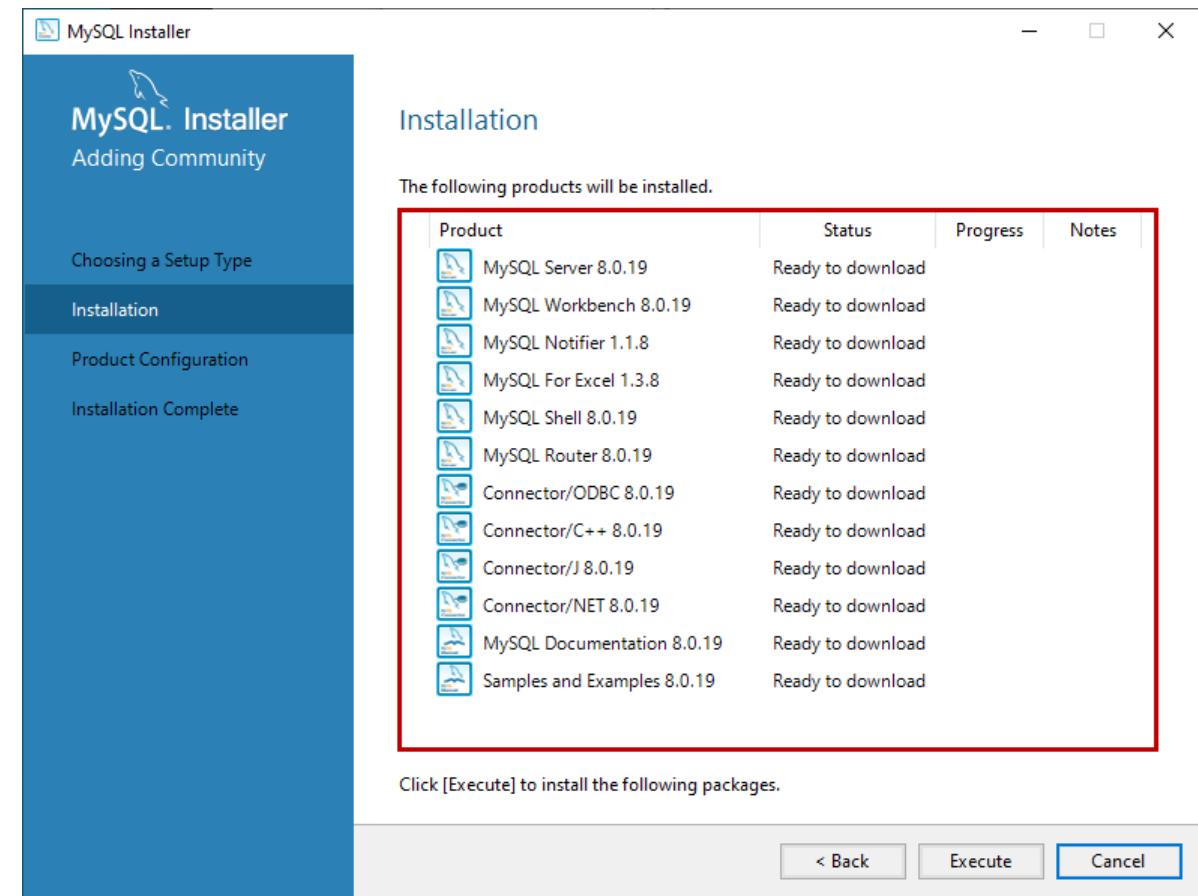
- Le programme d'installation vérifie tous les prérequis nécessaires pour le fonctionnement de tous les composants du serveur de base de données MySQL.
- Vous pouvez consulter les détails des exigences défaillantes sur la partie «**Check Requirements**».
- Cliquez sur Suivant.



02 - Préparer le serveur MySQL

Installation de MySQL Server

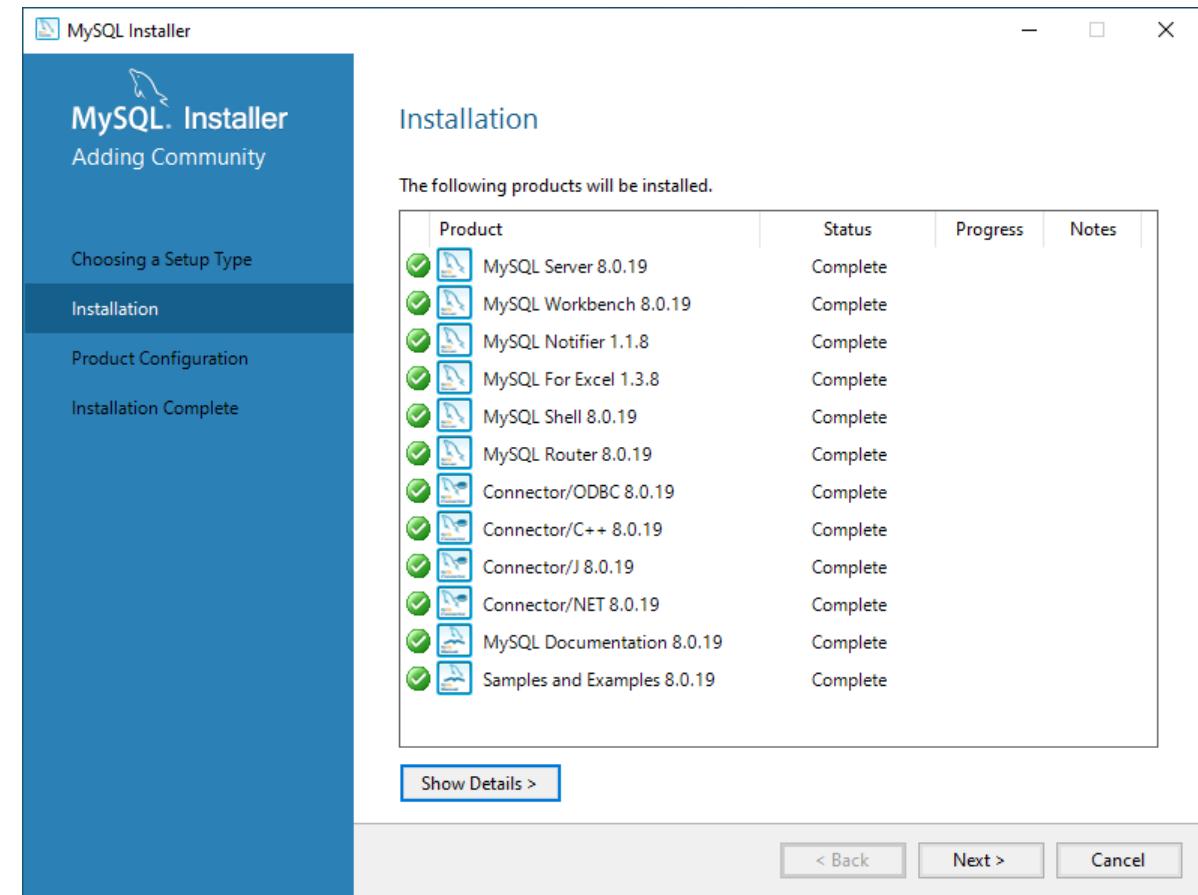
- Sur l'écran d'installation, le programme liste les composants MySQL qui vont être installés sur notre poste de travail. Consultez la liste et cliquez sur « Execute ».



02 - Préparer le serveur MySQL

Installation de MySQL Server

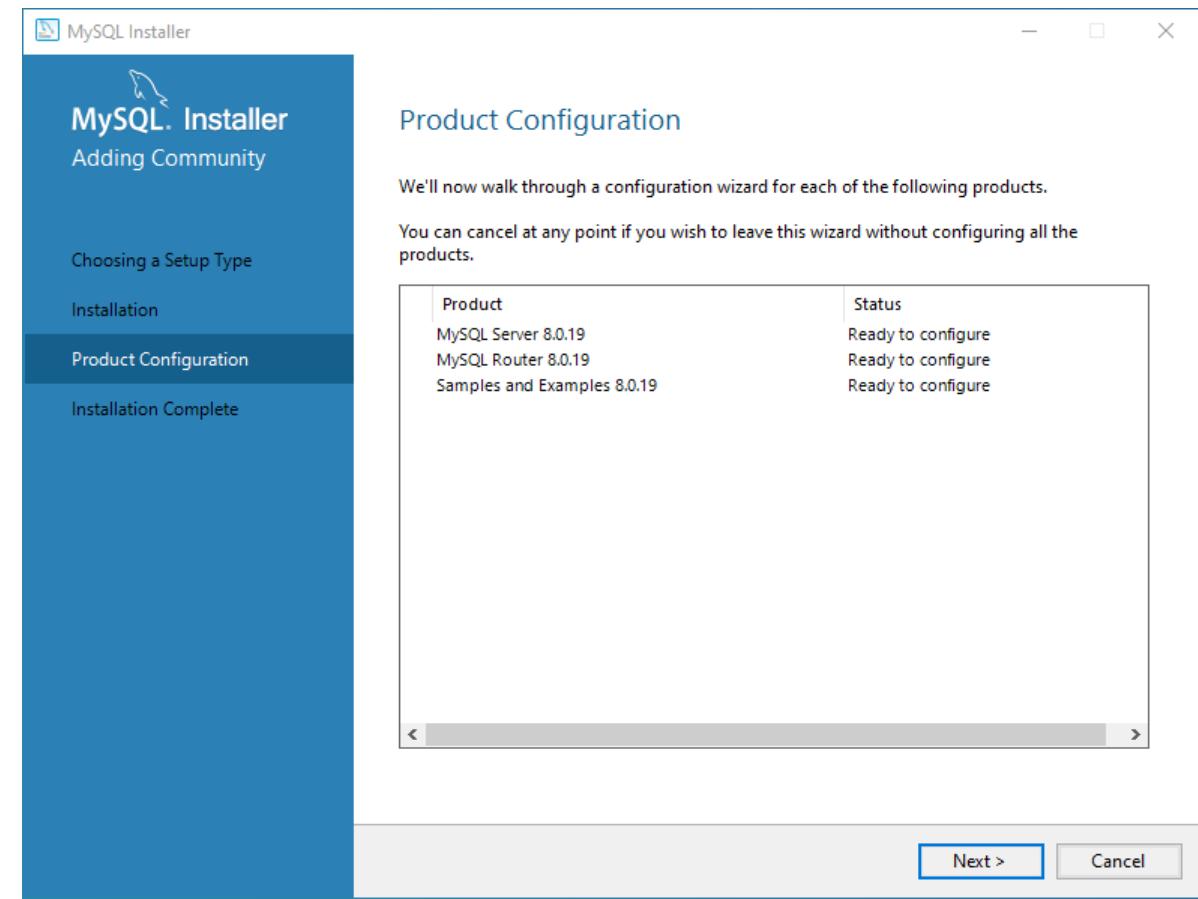
- Le programme télécharge et installe tous les produits.



02 - Préparer le serveur MySQL

Installation de MySQL Server

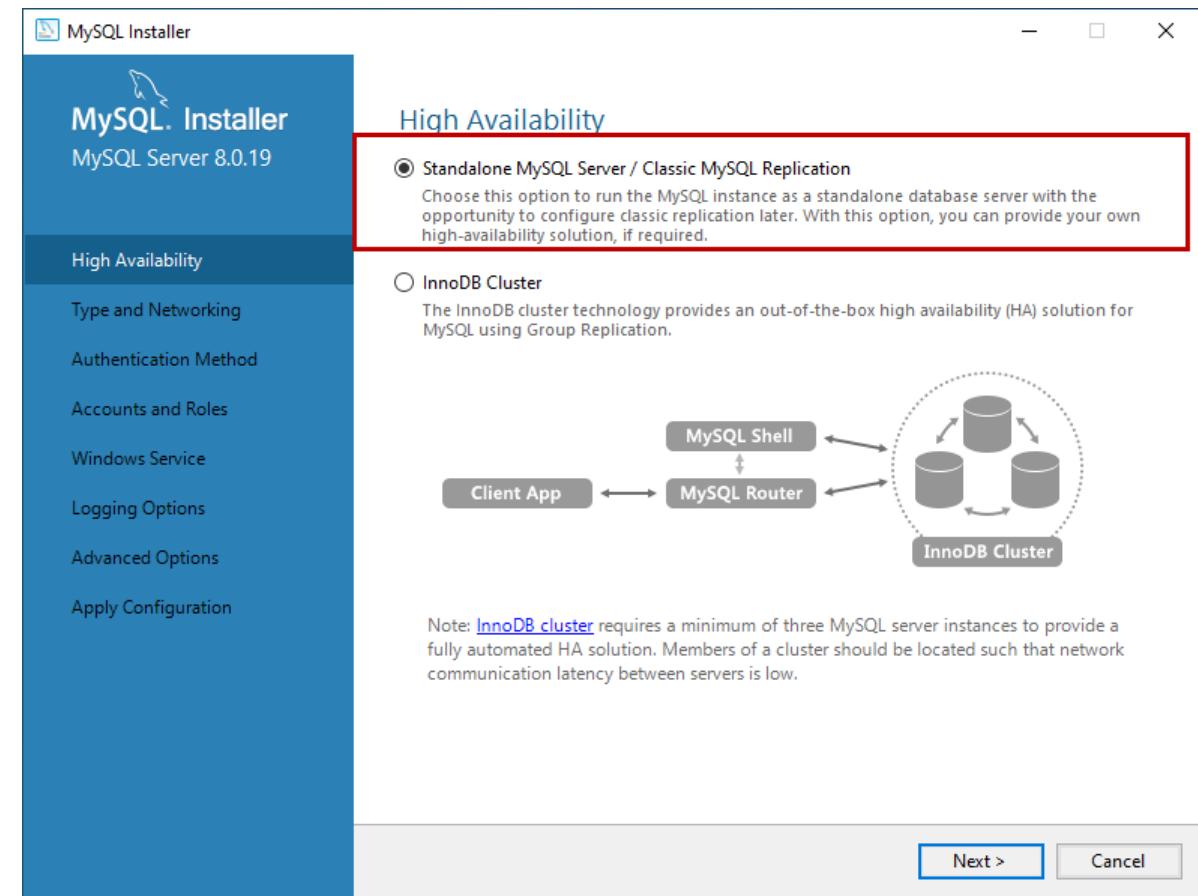
- Sur l'écran de configuration, vous pouvez voir la liste des produits qui doivent être configurés.
- Nous allons commencer par le serveur MySQL. Cliquer sur « **Next** ».



02 - Préparer le serveur MySQL

Installation de MySQL Server

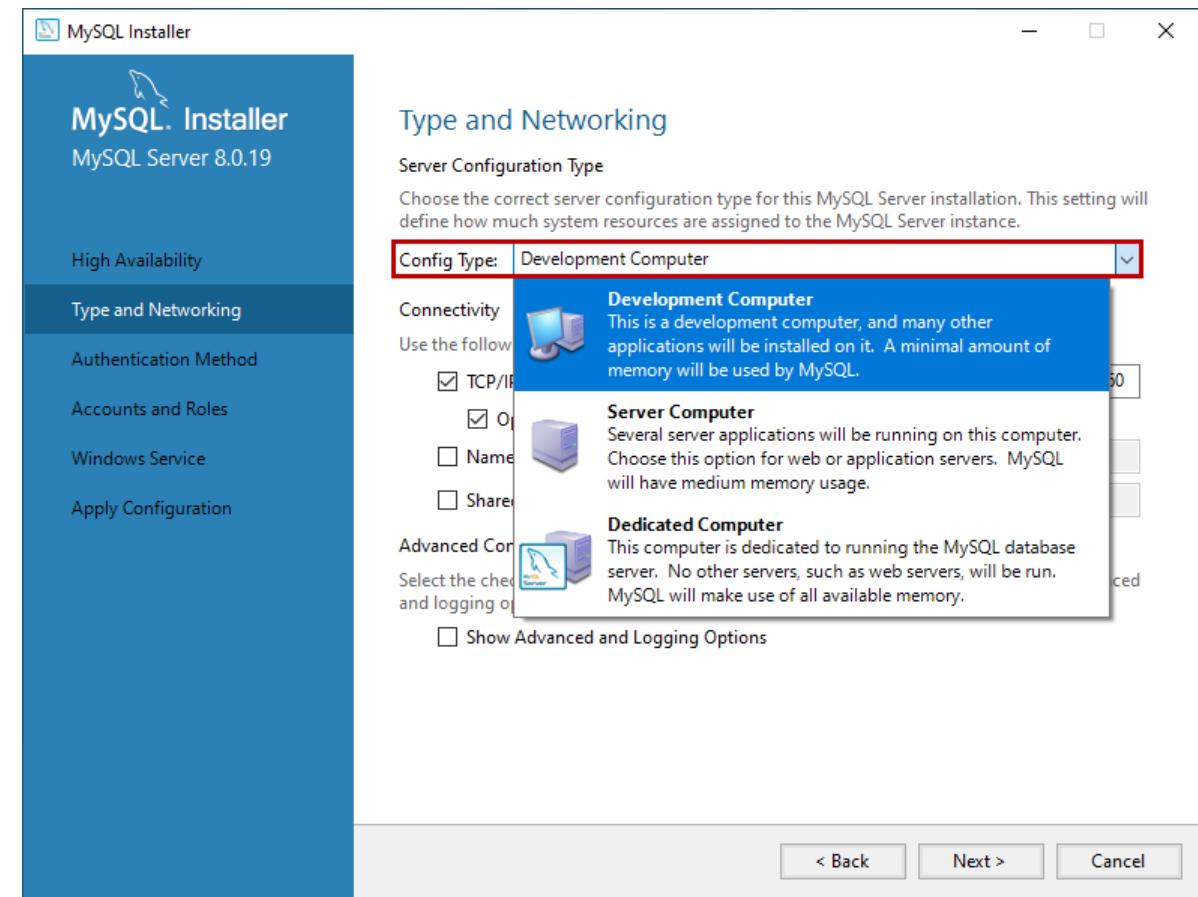
- Sur l'écran « Haute disponibilité », nous allons choisir d'effectuer une installation autonome de MySQL Server, choisissez donc « Standalone MySQL Server / Classic MySQL Replication ».



02 - Préparer le serveur MySQL

Installation de MySQL Server

- Sur le volet « **Type and Networking** », nous pouvons choisir le type de configuration MySQL.
- Il s'agit d'un ensemble prédéfini de paramètres de configuration qui détermine la quantité de ressources à allouer aux services MySQL. On peut choisir entre trois options :
 - Development Computer** : cette configuration utilise une quantité minimale de ressources.
 - Server Computer** : cette option convient lorsque nous installons des serveurs de base de données et des serveurs Web sur la même machine. La configuration alloue une quantité moyenne de ressources au service MySQL.
 - Dedicated Computer** : cette option est utilisée lorsque nous voulons créer un serveur MySQL dédié. La configuration alloue une grande quantité de ressources au service MySQL.
- On opte pour l'option « **Development Computer** ».

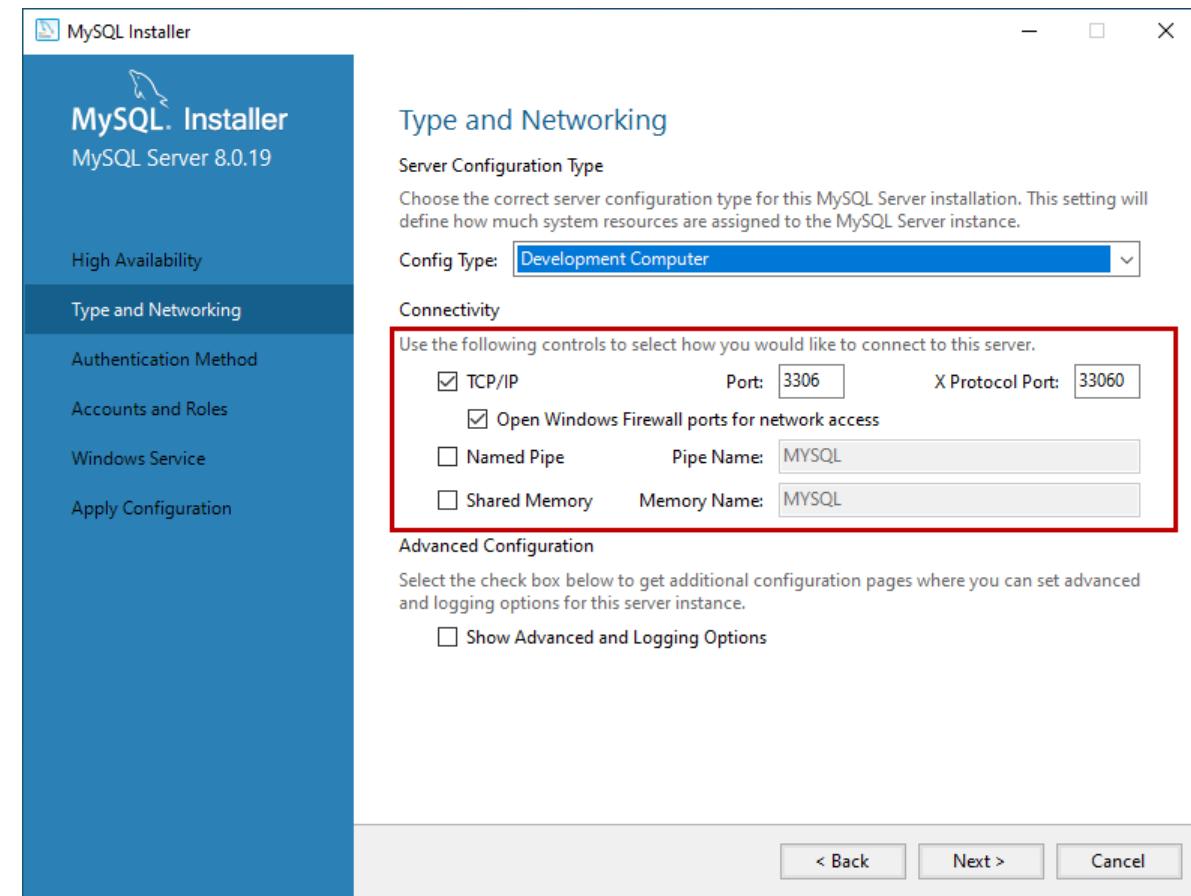


02 - Préparer le serveur MySQL

Installation de MySQL Server

Network Connectivity

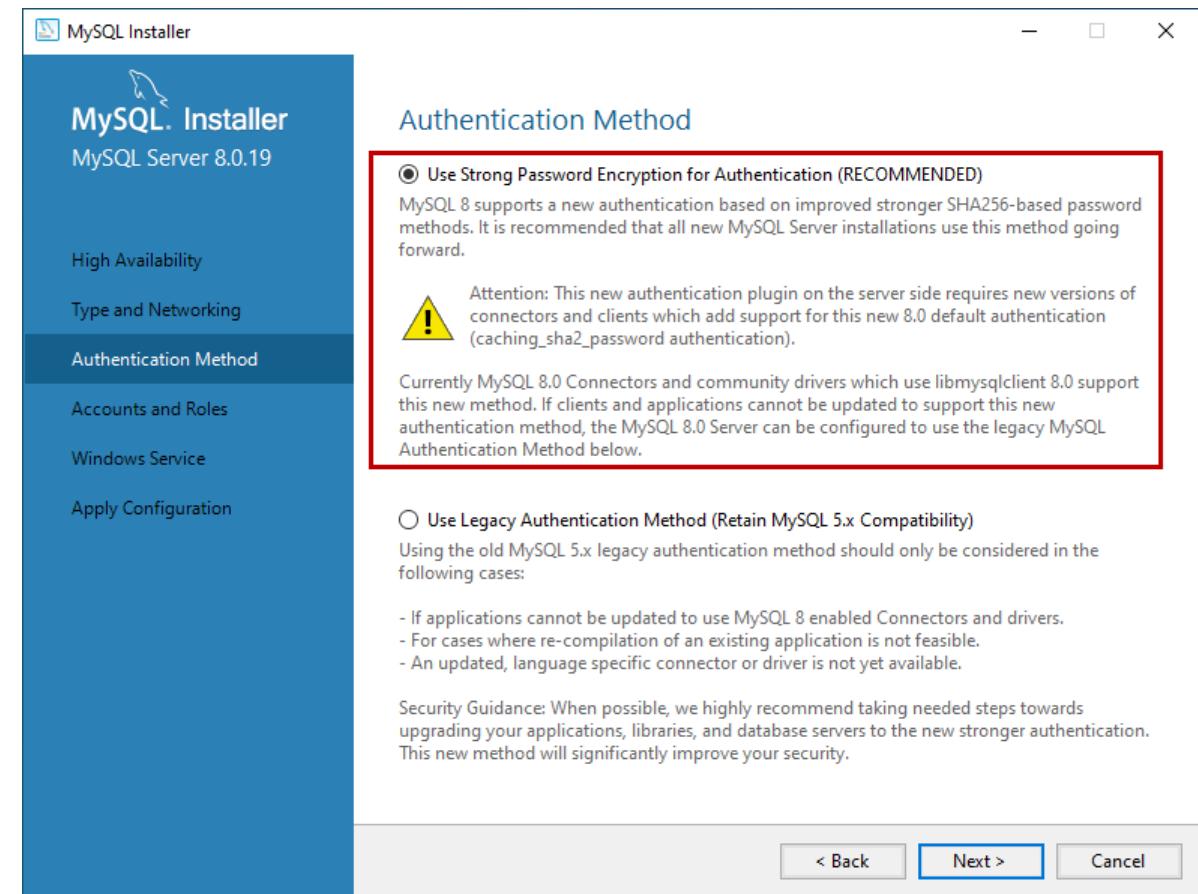
- Dans cette section, nous pouvons contrôler la façon dont les clients peuvent se connecter aux bases de données MySQL. Nous pouvons utiliser le protocole TCP/IP ou « Named Pipe » ou « Shared Memory ». Si nous souhaitons configurer un Named Pipe/Shared Memory, nous devons fournir le nom du Pipe ou de la Memory.
- Nous pouvons aussi spécifier le port par défaut pour nous connecter au serveur de base de données et choisir d'autoriser le numéro de port spécifié dans la zone de texte « Port » dans le pare-feu. Voir l'image suivante :



02 - Préparer le serveur MySQL

Installation de MySQL Server

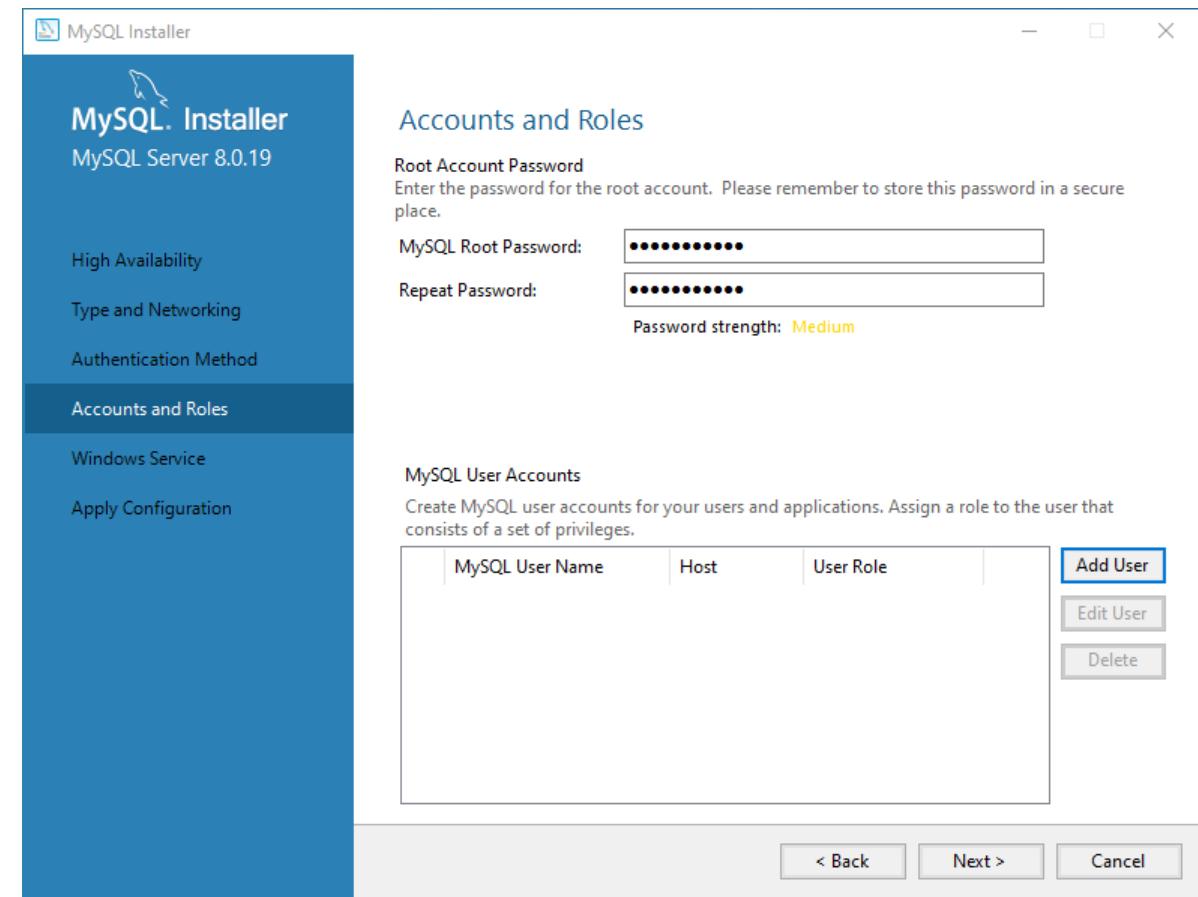
- Sur l'écran « Authentication Method », choisissez l'option « Strong Password Encryption for Authentication » si vous avez installé les dernières versions des connectors.



02 - Préparer le serveur MySQL

Installation de MySQL Server

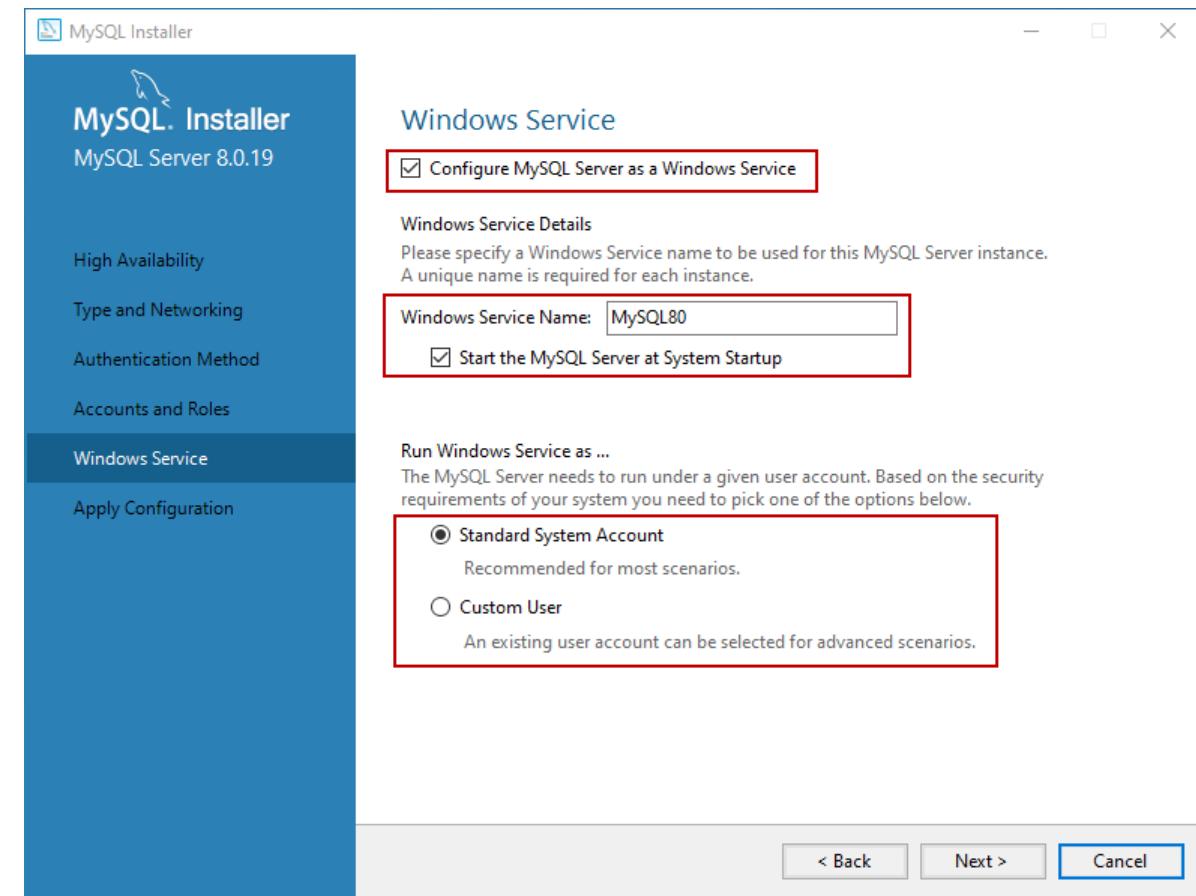
- Sur la partie « Accounts and Roles » vous pouvez spécifier le mot de passe du compte root. Le compte MySQL Root est un compte sysadmin par défaut et il doit être désactivé.
- Vous pouvez également créer d'autres utilisateurs en cliquant sur « Add User ». Dans la boîte de dialogue du compte d'utilisateur MySQL, fournissez un nom d'utilisateur, un nom d'hôte, le rôle de l'utilisateur, le type d'authentification et un mot de passe.



02 - Préparer le serveur MySQL

Installation de MySQL Server

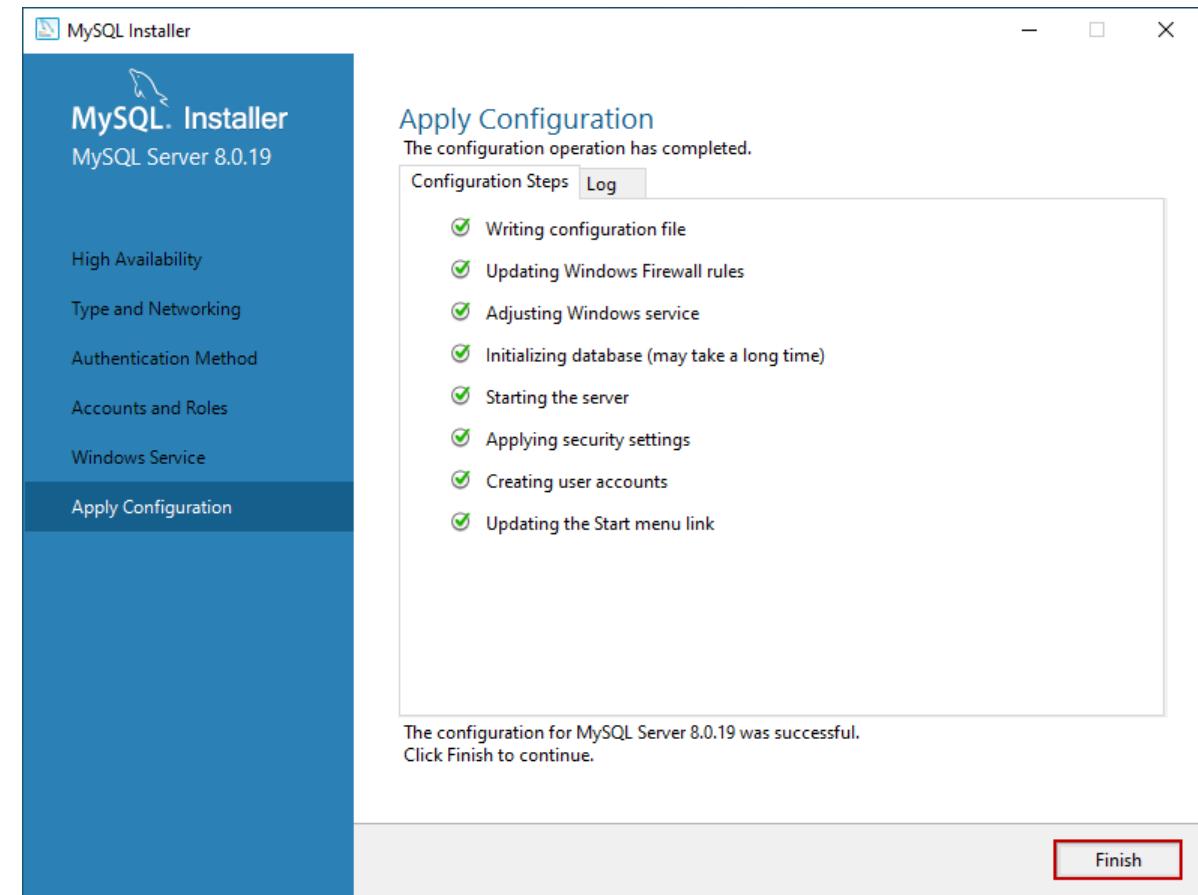
- Sur la partie « Windows Service », vous pouvez configurer le serveur MySQL pour qu'il s'exécute en tant que service Windows.
- Vous pouvez fournir le nom souhaité et le configurer pour démarrer automatiquement le service lorsque le système redémarre.
- Vous pouvez choisir le compte système standard ou fournir un utilisateur spécifique.



02 - Préparer le serveur MySQL

Installation de MySQL Server

- Sur le volet « **Apply Configuration** », vous pouvez consulter les configurations choisies. Une fois tous les paramètres de configuration vérifiés, cliquez sur « Exécuter ».
- L'installation et la configuration sont donc réussies.





CHAPITRE 2

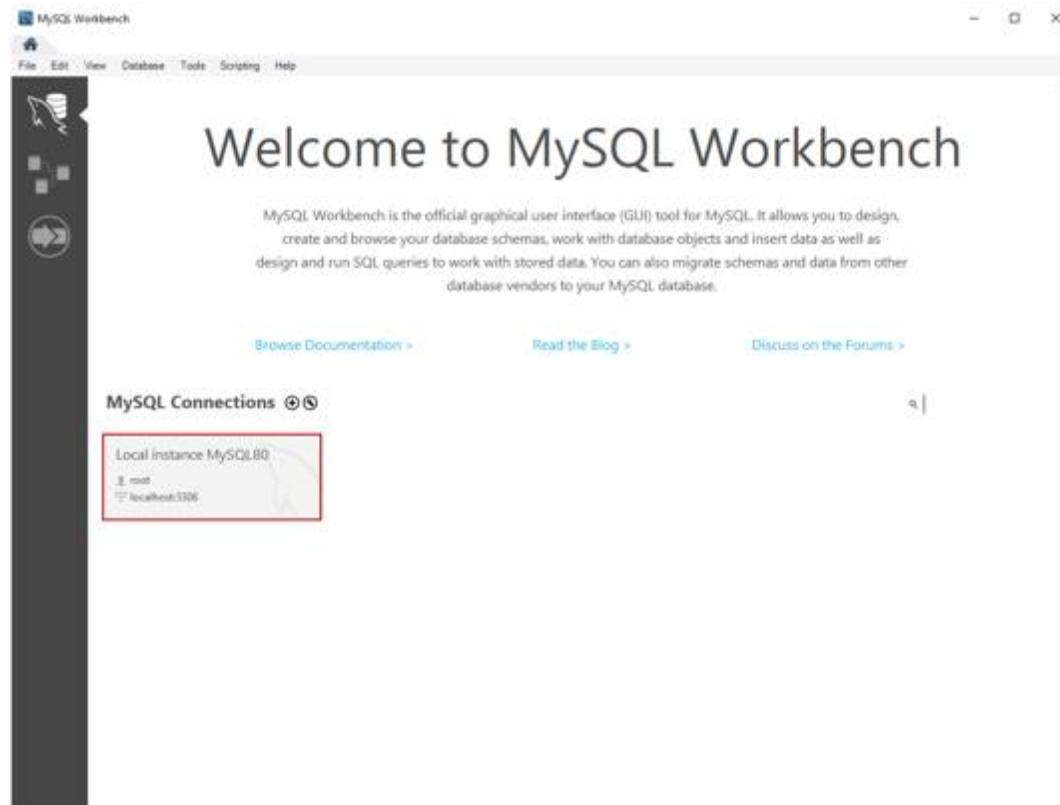
Préparer le serveur MySQL

1. Installation serveur MySQL
2. **Management des services MySQL**
3. Configuration des ports MySQL

02 - Préparer le serveur MySQL

Management des services MySQL

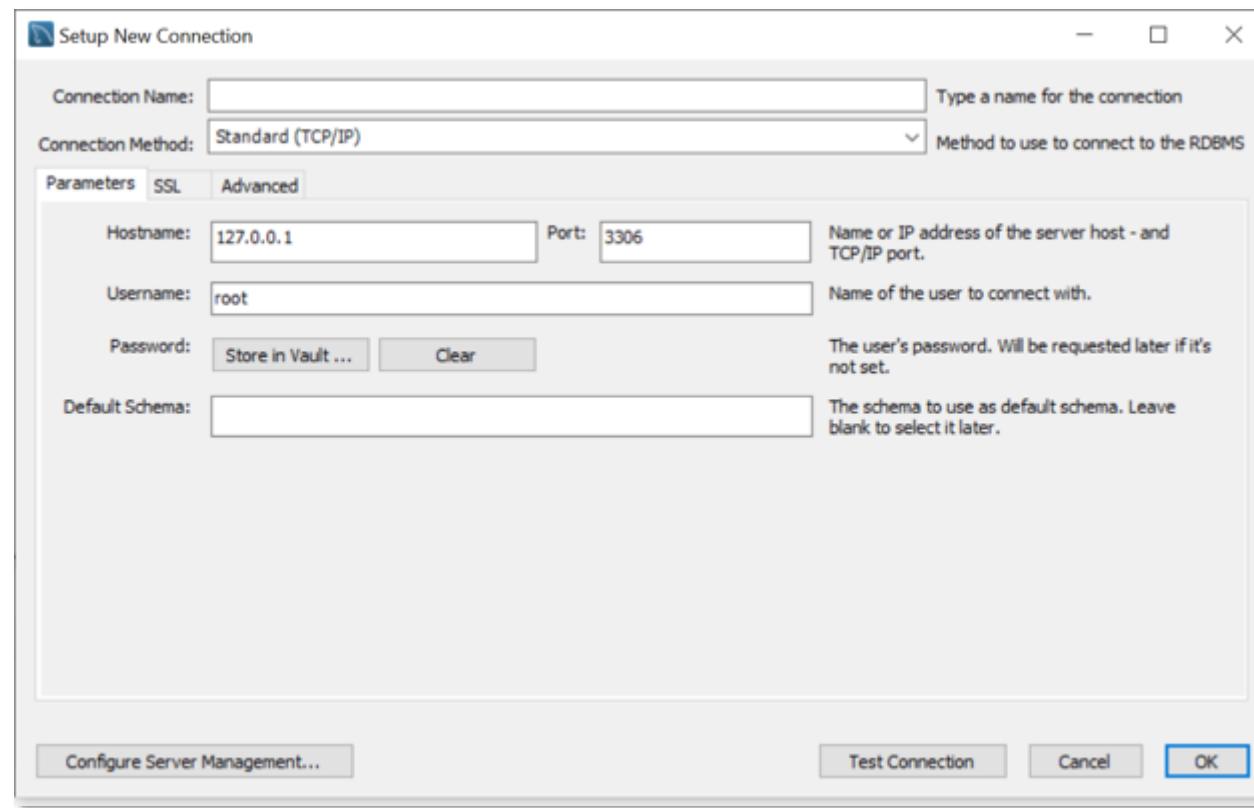
- Sur l'écran d'accueil de MySQL Workbench, vous pouvez voir la liste des connexions MySQL. On y trouve la liste des connexions MySQL configurées.
- Dans notre exemple, nous avons configuré un service local : « **local instance MySQL80** ».



02 - Préparer le serveur MySQL

Management des services MySQL

- On peut ajouter de nouvelles connexions à d'autres serveurs de bases de données en cliquant sur le bouton 
- Puis entrer les détails nécessaires.

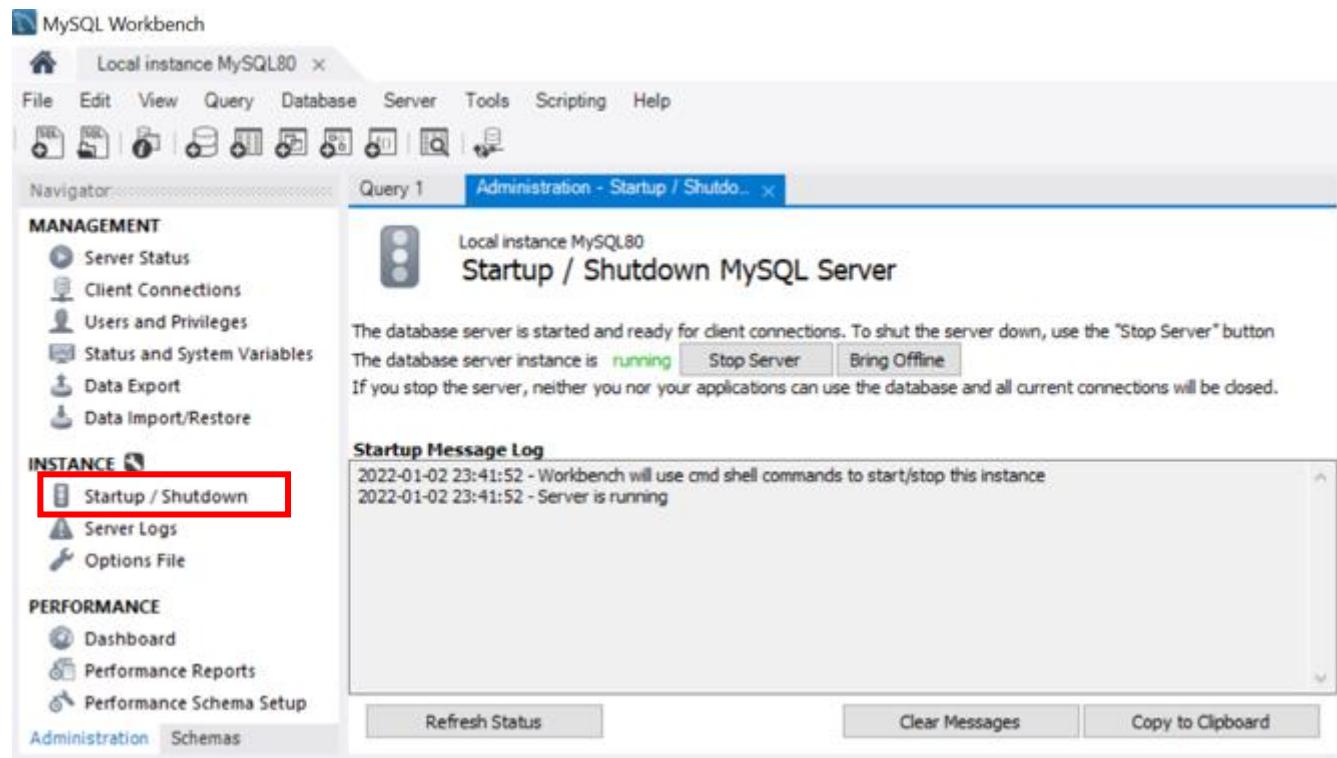


02 - Préparer le serveur MySQL

Management des services MySQL

Manager les services à partir de Workbench

- Afin de démarrer, arrêter ou redémarrer un service, nous cliquons sur la connexion relative pour avoir accès au menu de gestion de ce service. Sur le volet « INSTANCE », on trouve « Startup/Shutdown ».
- Cette page nous permet de manager les services.

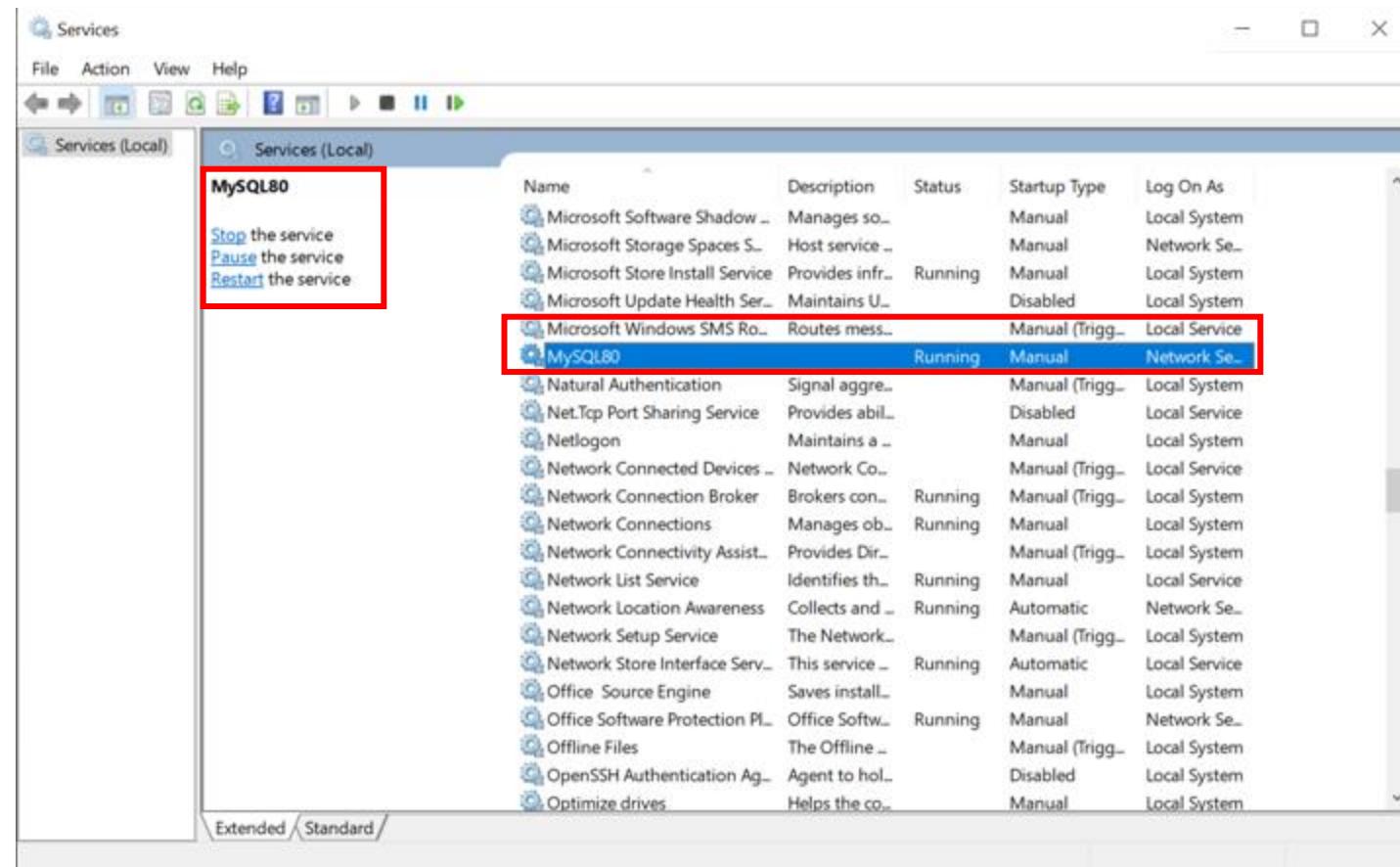


02 - Préparer le serveur MySQL

Management des services MySQL

Manager les services sur Windows

- Comme on a vu sur le volet configuration, MySQL server est configuré comme un service Windows.
- Afin de démarrer, arrêter ou redémarrer ce service, nous devons suivre les étapes suivantes :
 - Ouvrir « Exécuter » depuis le menu Windows ou en utilisant le raccourci bouton Windows + R.
 - Taper « services.msc ».
 - Chercher le service MySQL sur la liste des services Windows.
 - Vous pouvez cliquer sur « stop », « start » or « restart ».



02 - Préparer le serveur MySQL

Management des services MySQL



Manager les services sur Windows

- Il est aussi possible de faire de même à partir du command prompt.
- **Démarrer le service :**

```
C:\> "C:\Program Files\MySQL\MySQL Server 8.0\bin\mysqld"
```

- **Arrêter le service :**

```
C:\> "C:\Program Files\MySQL\MySQL Server 8.0\bin\mysqladmin" -u root shutdown
```

02 - Préparer le serveur MySQL

Management des services MySQL

Manager les services sur Linux

- Afin de démarrer, arrêter ou redémarrer les services, nous pouvons utiliser les commandes suivantes :

```
/etc/init.d/mysql start
```

```
/etc/init.d/mysql stop
```

```
/etc/init.d/mysql restart
```

- Sur les versions Linux qui utilisent la « service command » :

```
service mysql start
```

```
service mysql stop
```

```
service mysql restart
```



CHAPITRE 2

Préparer le serveur MySQL

1. Installation serveur MySQL
2. Management des services MySQL
3. **Configuration des ports MySQL**

02 - Préparer le serveur MySQL

Configuration des ports MySQL



Ports des connexions Client - Serveur MySQL

- Le port **3306** est le port par défaut du protocole classique MySQL (port). Il est utilisé par le client MySQL, les connecteurs MySQL et d'autres utilitaires.
- Le port du X Protocol ([mysqlx_port](#)), utilisé par des composants clients comme MySQL Shell, MySQL Connectors et MySQL Router, est calculé en multipliant le port utilisé pour le protocole MySQL classique par 10.

Port/Protocole par défaut	Description	SSL ou autres cryptages	Obligatoire	Direction
3306/TCP	Clients MySQL vers le serveur MySQL (protocole MySQL classique)	Oui	Oui, sauf si vous utilisez uniquement le protocole X	Du client au serveur MySQL
33060/TCP	Clients MySQL vers le serveur MySQL (protocole X)	Oui	Oui, sauf si vous utilisez uniquement le port 3306	Du serveur au client MySQL

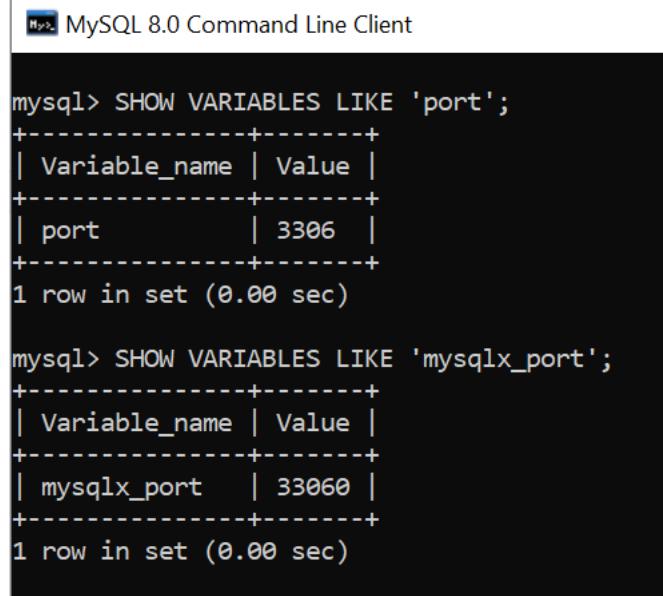
02 - Préparer le serveur MySQL

Configuration des ports MySQL

- On peut vérifier les valeurs des ports en utilisant les commandes suivantes sur la ligne de commande MySQL :

- mysql> **SHOW VARIABLES LIKE 'port';**
- mysql> **SHOW VARIABLES LIKE 'mysqlx_port'**

- Exemple :



MySQL 8.0 Command Line Client

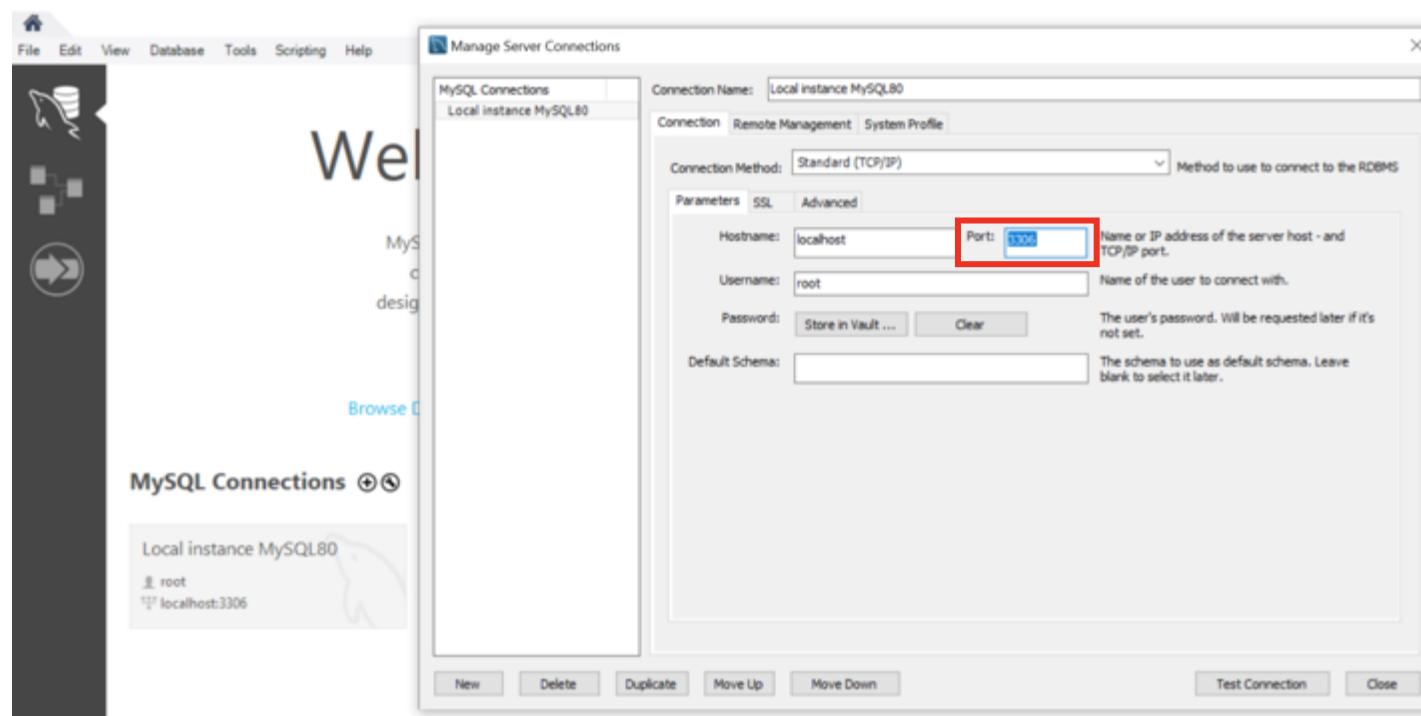
```
mysql> SHOW VARIABLES LIKE 'port';
+-----+-----+
| Variable_name | Value |
+-----+-----+
| port          | 3306  |
+-----+-----+
1 row in set (0.00 sec)

mysql> SHOW VARIABLES LIKE 'mysqlx_port';
+-----+-----+
| Variable_name | Value |
+-----+-----+
| mysqlx_port   | 33060 |
+-----+-----+
1 row in set (0.00 sec)
```

02 - Préparer le serveur MySQL

Configuration des ports MySQL

- Afin de changer le port par défaut :
- Sur Workbench : cliquez sur l'icône  pour avoir accès au gestionnaire de la connexion et modifier le port.



02 - Préparer le serveur MySQL

Configuration des ports MySQL

Sur Windows

- Trouvez le fichier de configuration « my.ini » dans le répertoire suivant :

C:\ProgramData\MySQL\MySQL Server 8.0\ (ProgramData est un dossier caché)

- Parcourez le fichier jusqu'à trouver l'expression suivante :

```
# The TCP/IP Port the MySQL Server will listen on
port=3306
```

- Modifiez le port et enregistrer le fichier.



Notez qu'il faut redémarrer les services après le changement des ports.



PARTIE 3

Manipuler les données

Dans ce module, vous allez :

- Vous initier à créer des Bases de Données
- Réaliser les différentes requêtes SQL
- Administrer une BDD





CHAPITRE 1

Créer une Base de Données

Ce que vous allez apprendre dans ce chapitre :

- Création d'une base de données
- Création des tables et des colonnes
- Intégration des contraintes d'intégrité sur les colonnes
- Manipulation des objets d'une Base de données





CHAPITRE 1

Créer une Base de Données

1. **Création des Bases de Données**
2. Choix de moteur
3. Création des tables
4. Définition des colonnes
5. Typage des colonnes
6. Contraintes d'intégrité
7. Manipulation d'objet table

01 - Créer une Base de Données

Création des Bases De Données



Rappel: Définition d'une base de données

Une **base de données** est une structure permettant de stocker un grand nombre d'informations afin d'en faciliter l'utilisation.

- Le fait de structurer les données a pour but d'assurer les fonctions fondamentales suivantes :
 - La fiabilité du stockage de l'information : La possibilité de restituer l'information stockée dans la base de données ;
 - La massification : Le traitement de grands volumes de données ;
 - L'optimisation : En terme de temps de traitement et espace de stockage ;
 - La sécurisation des accès aux données ;
 - La qualité des données : Vérification des règles de gestion et la conformité avec les modèles de conception ;
 - Le partage des données entre plusieurs acteurs (utilisateurs, applications...) et la gestion des concurrences d'accès.
- Ces fonctions sont assurées au moyen des SGBD : Système de Gestion de Bases de Données

01 - Crée une Base de Données

Création des Bases De Données



Les objets d'une base de données:

- **Les tables** : contenant des données ;
- **Les index** : servant à retrouver, trier, regrouper rapidement les données ;
- **Les déclencheurs (triggers)** : permettant d'exécuter des opérations particulières lors de l'insertion, la modification ou la suppression de données ;
- **Les types de données définis par l'utilisateur (UDT)** : servant de référentiel à plusieurs tables ;
- **Les valeurs par défaut (Defaults)** : autorisant le système à insérer des valeurs dans les colonnes non renseignées par l'utilisateur ;
- **Les vues, ou pseudo-tables (Views)** : offrant une vue particulière des données aux utilisateurs ;
- **Les fonctions définies par l'utilisateur (UDF)** : permettant de renvoyer soit une valeur, soit une table ;
- **Les procédures stockées** : exécutées par l'utilisateur pour produire un résultat donné ;
- **Les diagrammes (Diagrams)** : qui visualisent les relations entre les tables.

01 - Crée une Base de Données

Création des Bases De Données

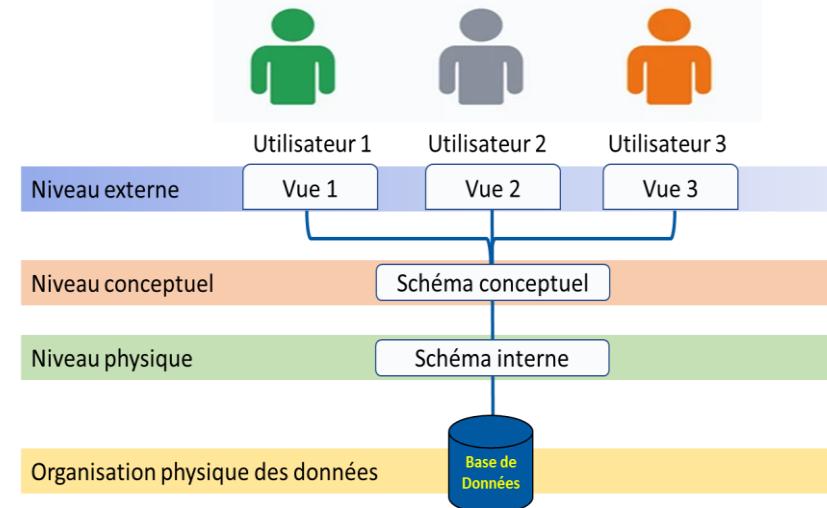
SGBD (Système de Gestion de Bases de Données)

Un SGBD (Système de Gestion de Bases de Données) est un ensemble logiciel qui permet la structuration, le stockage, et la manipulation d'une base de données.

En termes d'architecture, et selon le modèle ANSI/SPARC d'un SGBD comporte 3 niveaux :

- **Niveau conceptuel :**
 - Le niveau central d'un SGBD. Il correspond à une vue générale de toutes les données existant dans l'entreprise.
- **Niveau interne ou physique :**
 - Spécification du stockage physique des données (fichiers, disques, etc.) et des méthodes d'accès (index, etc.).
- **Niveau externe :**
 - Il s'agit d'une vue externe pour chaque groupe d'utilisateurs sur un sous ensemble de la base ;
 - Chaque schéma externe est généralement un sous schéma du schéma conceptuel mais peut contenir parfois des informations supplémentaires.

Le modèle ANSI/SPARC d'un SGBD selon trois niveaux



01 - Créer une Base de Données

Création des Bases De Données



Types de SGBD :

- **SGBD hiérarchique :**
 - Les données sont représentées dans la base sous la forme d'un arbre dont le parcours se fait du père vers le fils à l'aide de pointeurs.
- **SGBD relationnel :**
 - Les SGBDR Dominent le marché des SGBD. La théorie derrière ce type de systèmes est fondée sur la théorie mathématique des relations. Il s'agit de représentation simple des données sous forme de tables constituées de lignes et de colonnes.
- **SGBD objet :**
 - Les SGBDOO enregistrent les données sous forme d'objets ; les données sont enregistrées avec les procédures et les fonctions qui permettent de les manipuler.

01 - Créer une Base de Données

Création des Bases De Données



Exemples de SGBD :

- **Oracle** est un SGBD relationnel et relationnel-objet très utilisé pour les applications professionnelles.
- **PostgreSQL** est un SGBD relationnel puissant qui offre une alternative libre (licence BSD) aux solutions commerciales comme Oracle ou IBM.
- **Access** est un SGBD relationnel Microsoft, qui offre une interface graphique permettant de concevoir rapidement des applications de petite envergure ou de réaliser des prototypes.
- **MongoDb** est un SGBD non-relationnel libre (licence Apache) orienté document. Il permet de gérer facilement de très grandes quantités de données - dans un format arborescent JSON - réparties sur de nombreux ordinateurs.
- **MySQL** est un système de gestion de bases de données relationnelles (SGBDR) open source. Ce SGBDR d'Oracle est basé sur le langage SQL (Structured Query Language) et fonctionne sur pratiquement toutes les plates-formes comme Linux, UNIX et Windows.

Remarque : Nous utiliserons pour les TD/TP le SGBD MySQL.

01 - Créer une Base de Données

Création des Bases De Données



Le langage de requêtes SQL (Structured Query Language) :

- Il s'agit d'un langage d'interrogation des bases de données relationnelles qui permet d'effectuer les tâches suivantes :
 - Définition et modification de la structure de la base de données
 - Interrogation et modification non procédurale (c'est à dire interactive) de la base de données
 - Contrôle de sécurité et d'intégrité de la base
 - Sauvegarde et restauration des bases
- Le langage SQL n'est pas sensible à la casse, cela signifie que l'on peut aussi bien écrire les instructions en minuscules qu'en majuscule.
- Le langage de commandes SQL peut être réparti en quatre catégories :
 - **LDD (Langage de définition des données)** : langage de manipulation des structures de la base.
 - **LMD (Langage de manipulation des données)** : il permet de consulter et de modifier le contenu de la base de données.
 - **LQD (Langage des queries des données)** : langage de requêtes sur les données.
 - **LCD (Langage de contrôle des données)** : il permet de gérer les priviléges et les différents droits d'accès à la base de données.

01 - Créer une Base de Données

Création des Bases De Données



Création d'une base de données

Les interfaces des SGBD offrent la possibilité de créer des bases de données. Cette opération est aussi possible à partir de la ligne de commande.

- **Syntaxe :**
 - Pour créer une base de données nommée « nom_base », il suffit d'utiliser la requête suivante : 'CREATE DATABASE nom_base'
- **Options :**
 - La syntaxe utilisée pour chacune des options qui peuvent accompagner la commande CREATE DATABASE dépendent du SGBD utilisé. Il convient alors de vérifier la documentation du SGBD pour avoir une idée sur les différentes options possibles : définition des jeux de caractères, le propriétaire de la base, les limites de connexion...
- **Exemple :**
 - Création d'une base de données sur MySQL (Page suivante)

01 - Créer une Base de Données

Création des Bases De Données



Exemple : Création d'une base de données sur MySQL

Pour créer une nouvelle base de données sur MySQL, nous utilisons la commande CREATE DATABASE avec la syntaxe suivante :

```
CREATE DATABASE [IF NOT EXISTS] database_name  
[CHARACTER SET charset_name]  
[COLLATE collation_name]
```

Il faut noter que :

- Le nom de la base de données doit être unique. Si une autre base existe avec le même nom, MySQL retourne une erreur.
- Utiliser l'option **IF NOT EXISTS** pour créer conditionnellement une base de données si elle n'existe pas.
- On peut spécifier les options **CHARACTER SET** et **COLLATE** et le classement de la nouvelle base de données. Sinon MySQL utilisera les valeurs par défaut pour la nouvelle base de données.

01 - Créer une Base de Données

Création des Bases De Données



Sur MySQL command line Client :

- Connectez-vous au serveur MySQL avec un compte utilisateur disposant du privilège CREATE DATABASE.
- Exécutez la commande **CREATE DATABASE dbTest** ; et appuyez sur Entrée :

```
CREATE DATABASE testdb;
```

- MySQL retourne le résultat suivant :

```
Query OK, 1 row affected (0.04 sec)
```

- On peut utiliser la commande **SHOW CREATE DATABASE** pour examiner la base de données créée :

```
SHOW CREATE DATABASE testdb;
```

01 - Créer une Base de Données

Création des Bases De Données



- Voici le résultat de cette commande :

```
mysql> SHOW CREATE DATABASE dbTest;
+-----+-----+
| Database | Create Database |
+-----+-----+
| dbTest   | CREATE DATABASE `dbTest` /*!40100 DEFAULT CHARACTER SET utf8mb4 COLLATE utf8mb4_0900_ai_ci */ /*!80016 DEFAULT ENCRYPTION='N' */ |
+-----+-----+
1 row in set (0.00 sec)
```

- Enfin, sélectionnez la base de données nouvellement créée avec laquelle on veut travailler en utilisant la commande USE :

```
USE testdb;
```

- Le système confirme par le message suivant :

```
Database changed
```

- Utiliser la commande EXIT pour quitter le programme.



CHAPITRE 1

Créer une Base de Données

1. Création des Bases de Données
2. **Choix de moteur**
3. Création des tables
4. Définition des colonnes
5. Typage des colonnes
6. Contraintes d'intégrité
7. Manipulation d'objet table

01 - Créer une Base de Données

Choix du moteur



Moteurs de stockage

Le moteur de stockage d'un Système de Gestion de Base de Données (SGBD), aussi appelé **moteur de table**, est l'ensemble d'algorithmes qui permettent de stocker et d'accéder aux données. En général, les SGBD utilisent un seul moteur de stockage qui est optimisé au mieux pour la lecture, l'écriture et la suppression de données.

MySQL se distingue des autres SGBD par le fait de donner à l'utilisateur le libre choix d'utiliser un moteur de table parmi plusieurs moteurs différents. Ces moteurs de stockage peuvent être transactionnels ou non-transactionnels.

On se retrouve ainsi avec des bases où plusieurs moteurs peuvent coexister. C'est un choix de conception qui a ses avantages et inconvénients.

01 - Créer une Base de Données

Choix du moteur



Moteurs de stockage

Afin de consulter la liste des engins mis à notre disposition par MySQL, on peut exécuter la commande :

- SHOW ENGINES sur la ligne de commande MySQL comme suit :

Engine	Support	Comment	Transactions	XA	Savepoints
MEMORY	YES	Hash based, stored in memory, useful for temporary tables	NO	NO	NO
MRG_MYISAM	YES	Collection of identical MyISAM tables	NO	NO	NO
CSV	YES	CSV storage engine	NO	NO	NO
FEDERATED	NO	Federated MySQL storage engine	NULL	NULL	NULL
PERFORMANCE_SCHEMA	YES	Performance Schema	NO	NO	NO
MyISAM	YES	MyISAM storage engine	NO	NO	NO
InnoDB	DEFAULT	Supports transactions, row-level locking, and foreign keys	YES	YES	YES
BLACKHOLE	YES	/dev/null storage engine (anything you write to it disappears)	NO	NO	NO
ARCHIVE	YES	Archive storage engine	NO	NO	NO

9 rows in set (0.01 sec)

01 - Créer une Base de Données

Choix du moteur



Moteurs de stockage

- Chacun de ces moteurs de table possèdent des caractéristiques propres qui peuvent représenter des atouts ou des inconvénients selon le type d'application qui aura besoin d'une base de données.
- A partir de la version MySQL 5.5 , le moteur par défaut est InnoDB.
- On peut choisir le moteur de stockage au moment de la création d'une table.

La syntaxe est la suivante :

```
CREATE TABLE <nomTable> (...) ENGINE <nomMoteur>
```

01 - Créer une Base de Données

Choix du moteur



Quel moteur choisir?

Les moteurs MySQL les plus utilisés sont MyISAM et InnoDB et Memory, le tableau suivant présente une comparaison entre ces moteurs :

Moteur	Avantages	Inconvénients
MyISAM : un moteur non transactionnel assez rapide en écriture et très rapide en lecture.	<ul style="list-style-type: none">Très rapide en lectureExtrêmement rapide pour les opérations COUNT() sur une table entièreLes index FULLTEXT pour la recherche sur des textes	<ul style="list-style-type: none">Pas de gestion des relationsPas de gestion des transactionsBloque une table entière lors d'opérations d'insertions, suppressions ou mise à jour des données
InnoDB : un moteur relationnel performant faisant partie de la famille des moteurs transactionnels.	<ul style="list-style-type: none">Gestion des relationsGestion des transactionsVerrouillage à la ligne et non à la table	<ul style="list-style-type: none">Plus lent que MyISAMOccupe plus de place sur le disque durOccupe plus de place en mémoire vive
Memory : un moteur de stockage permettant de créer des tables directement dans la mémoire vive, sans passer par le disque dur pour stocker les données. Ceci en fait le moteur de stockage le plus rapide que propose MySQL, mais aussi le plus dangereux.	<ul style="list-style-type: none">Le moteur le plus rapideNe consomme pas de place sur le disque dur	<ul style="list-style-type: none">Les données sont volatiles : un arrêt du serveur et elles disparaissentPas de champs BLOB ou TEXT

Afin de faciliter le choix, pour une petite base, sans liens entre les tables, et dont la cohérence des données n'est pas vitale, on peut opter pour MyISAM .

Dans les autres cas, choisissez InnoDB.



CHAPITRE 1

Créer une Base de Données

1. Création des Bases de Données
2. Choix de moteur
- 3. Crédit des tables**
4. Définition des colonnes
5. Typage des colonnes
6. Contraintes d'intégrité
7. Manipulation d'objet table

01 - Créer une Base de Données

Création des tables



La création des tables est possible à partir de la ligne de commande.

- **Syntaxe :** Pour créer une table « nom_table », il suffit d'utiliser la requête suivante:

```
CREATE TABLE nom_table
```

- **Exemple :** Création d'une table dans une base de données MySQL
 - Pour créer une nouvelle base de données sur MySQL, nous utilisons la commande CREATE DATABASE avec la syntaxe suivante :

```
CREATE TABLE [IF NOT EXISTS] table_name(  
    column_1_definition,  
    column_2_definition,  
    ...,  
    table_constraints  
) ENGINE=storage_engine;
```

Sur cette syntaxe, on définit :

- Le nom de la table. « IF NOT EXISTS » est une commande optionnelle, et permet de vérifier si une table avec le même nom existe déjà.
- La liste des colonnes de cette table, séparées par des virgules.
- La liste des contraintes comme UNIQUE, CHECK, PRIMARY KEY et FOREIGN KEY.
- Le moteur de stockage avec la clause : ENGINE. Si on n'utilise pas cette option, MySQL va utiliser le moteur défini par default : InnoDB.



CHAPITRE 1

Créer une Base de Données

1. Création des Bases de Données
2. Choix de moteur
3. Création des tables
4. **Définition des colonnes**
5. Typage des colonnes
6. Contraintes d'intégrité
7. Manipulation d'objet table

01 - Créer une Base de Données

Définition des colonnes



La syntaxe de définition d'une colonne est la suivante :

```
nom_colonne type_donnee Liste_contraintes;
```

Avec :

- Nom_colonne : le nom qu'on va donner à cette colonne.
- Type_donnee : le type et taille de données qui vont être stockées dans cette colonne (numériques, caractères, date..).
- Liste_contraintes : la liste des contraintes sur cette colonne.



CHAPITRE 1

Créer une Base de Données

1. Création des Bases de Données
2. Choix de moteur
3. Création des tables
4. Définition des colonnes
5. **Typage des colonnes**
6. Contraintes d'intégrité
7. Manipulation d'objet table

01 - Créer une Base de Données

Typage des colonnes



Les types de données qui représentent la nature des valeurs stockées dans une colonne appartiennent à trois catégories : Numérique, date/heure et chaîne (caractères). Voici les listes des types supportés par MySQL.

1 - Type de données numériques

Type	Taille	Utilisation
TINYINT	1 octet	petites valeurs entières/ Boolean
SMALLINT	2 octets	valeur entière
MEDIUMINT	3 octets	valeur entière
INT ou INTEGER	4 octets	valeur entière
BIGINT	8 octets	Valeur maximale entier
FLOAT	4 octets	Simple précision valeurs à virgule flottante
DOUBLE	8 octets	Double-précision valeurs à virgule flottante
DECIMAL	De DECIMAL (M, D), si M> D, M + 2 est par ailleurs D + 2	valeur décimale

01 - Crée une Base de Données

Type des colonnes



2 - Type de données caractères

Type	Taille (octets)	Utilisation
CHAR	0-255	chaîne longueur fixe
VARCHAR	0-65535	chaînes de longueur variable
TINYBLOB	0-255	Pas plus de 255 caractères dans une chaîne binaire
TINYTEXT	0-255	Courtes chaînes de texte
BLOB	0-65535	données textuelles longues sous forme binaire
TEXTE	0-65535	Longue données de texte
MEDIUMBLOB	0-16777215	forme binaire de longueur moyenne des données de texte
MEDIUMTEXT	0-16777215	longueur moyenne des données de texte
LONGBLOB	0-4294967295	Grands données de texte sous forme binaire
LONGTEXT	0-4294967295	Grande données de texte



01 - Crée une Base de Données

Typage des colonnes



3 - Type de données date/heure

Type	Taille (Byte)	Format	Utilisation
DATE	3	AAAA-MM-JJ	Les valeurs de date
TIME	3	HH: MM: SS	Valeur temps ou la durée
ANNÉE	1	AAAA	Année Valeur
DATETIME	8	AAAA-MM-JJ HH: MM: SS	Mixage valeurs date et heure
TIMESTAMP	4	AAAAMMJJ HHMMSS	Date de mélange et la valeur temps, un horodatage



CHAPITRE 1

Créer une Base de Données

1. Création des Bases de Données
2. Choix de moteur
3. Création des tables
4. Définition des colonnes
5. Typage des colonnes
6. **Contraintes d'intégrité**
7. Manipulation d'objet table

Définition

- Une contrainte d'intégrité est une règle qui définit la cohérence d'une donnée ou d'un ensemble de données de la BD. Les contraintes peuvent avoir une portée sur une colonne ou sur une table lorsque la contrainte porte sur plusieurs colonnes.
- Il existe trois types de contraintes d'intégrité : (STP sous forme de schéma)
 - Intégrité de domaine : (NOT NULL, DEFAULT, UNIQUE...)**
 - Spécifie un ensemble de valeurs pour une colonne
 - Détermine si les valeurs nulles sont autorisées
 - Implémentation par contrôle de validité
 - Intégrité des entités : (PRIMARY KEY)**
 - Précise la clé primaire de chaque table
 - Intégrité référentielle : (FOREIGN KEY/REFERENCES)**
 - Assure l'intégrité des relations entre les clés primaires et les clés étrangères.

Les contraintes d'intégrité MySQL :

1 - PRIMARY KEY

- Une clé primaire est une colonne ou un ensemble de colonnes qui identifie de manière unique chacune des lignes de la table. Cette colonne doit vérifier les conditions suivantes :
 - Une clé primaire doit contenir des valeurs uniques. Si la clé primaire se compose de plusieurs colonnes, la combinaison de valeurs dans ces colonnes doit être unique.
 - Une colonne de clé primaire ne peut pas avoir de valeurs NULL. Toute tentative d'insertion ou de mise à jour de NULL dans les colonnes de clé primaire entraînera une erreur. Notez que MySQL ajoute implicitement une contrainte **NOT NULL** aux colonnes de clé primaire.
 - Une table ne peut avoir qu'une et une seule clé primaire.
- Lorsque vous définissez une clé primaire pour une table, MySQL crée automatiquement un index appelé **PRIMARY**.

1 - PRIMARY KEY (suite)

En général, la clé primaire d'une table est définie dans l'instruction **CREATE TABLE**.

Si la clé primaire est une seule colonne, **PRIMARY KEY** (contrainte de colonne) est utilisée avec la syntaxe suivante :

```
CREATE TABLE nom_table(
    primary_key_colonne Type_donnee PRIMARY KEY,
    ...
);
```

1 - PRIMARY KEY (suite)

Si la clé primaire est composée de plusieurs colonnes, **PRIMARY KEY** (contrainte de table) est utilisée avec la syntaxe suivante :

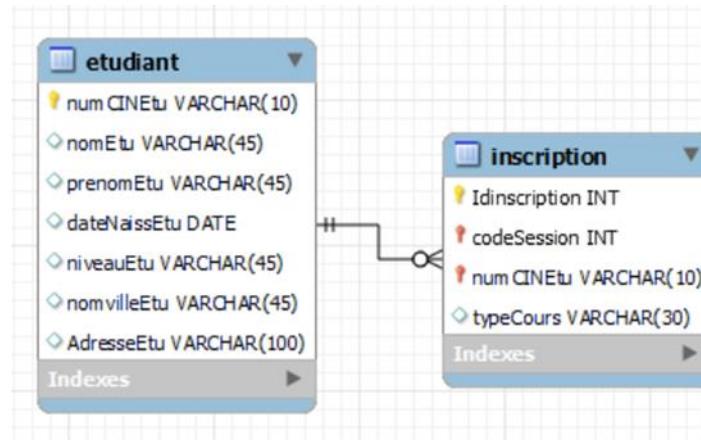
```
CREATE TABLE nom_table(  
    primary_key_colonne1 type_donnée,  
    primary_key_colonne type_donnée,  
    ...  
    PRIMARY KEY(liste_colonnes)  
);
```

On sépare les colonnes de liste_colonnes par des virgules (,).

2 - FOREIGN KEY

Une clé étrangère est une colonne ou un groupe de colonnes d'une table qui renvoie à une colonne ou à un groupe de colonnes d'une autre table. La clé étrangère impose des contraintes sur les données des tables associées, ce qui permet à MySQL de maintenir l'intégrité référentielle.

Exemple : Rappelons la relation entre les tables **étudiant** et **inscription** de notre base de données **CentreFormation**.



2 - FOREIGN KEY (suite)

Chaque étudiant peut avoir aucune ou plusieurs inscriptions, et chaque inscription appartient à un étudiant unique.

Cette relation est traduite au niveau du MLD par une clé étrangère dans la table inscription (référencée) qui renvoie à la colonne identifiante de la table etudiant: numCINETu.

La table etudiant est appelée table parent ou table référencée, et la table inscription est appelée table enfant ou table de référence.

Afin de Créer une contrainte FOREIGN KEY pour la table inscription:

```
CONSTRAINT fk1_inscription  
FOREIGN KEY (numCINETu)  
REFERENCES etudiant(numCINETu)
```

2 - FOREIGN KEY (suite)

Voici la syntaxe générale qu'on utilise pour la définition d'une contrainte de clé étrangère

```
CONSTRAINT nom_contrainte
FOREIGN KEY (nom_colonne)
REFERENCES table_parent(nom_colonne_p)
[ON DELETE reference_option]
[ON UPDATE reference_option]
```

Avec :

- **nom_contrainte** : le nom qu'on va donner à la contrainte FOREIGN KEY
- **nom_colonne** : la colonne dans la table de référencement
- **table_parent** : la table référencée
- **nom_colonne_p** : clé primaire de la table référencée
- **reference_option** : détermine l'action que MySQL effectuera lorsque les valeurs des colonnes de clé parent seront supprimées (**ON DELETE**) ou mises à jour (**ON UPDATE**).



3 - NOT NULL

- La contrainte NOT NULL est une contrainte de colonne qui garantit que les valeurs stockées dans une colonne ne sont pas NULL.
- La syntaxe d'une contrainte NOT NULL est la suivante :

```
nom_colonne type_donnee NOT NULL;
```

4 - DEFAULT

- La contrainte MySQL DEFAULT permet de spécifier une valeur par défaut pour une colonne :

```
nom_colonne type_donnee DEFAULT valeur_par_defaut;
```

- Dans cette syntaxe, le mot clé DEFAULT est suivi par la valeur par défaut de la colonne. Le type de la valeur par défaut doit correspondre au type de données de la colonne.

Les contraintes d'intégrité MySQL : UNIQUE

- UNIQUE est une contrainte d'intégrité qui garantit que les valeurs d'une colonne ou d'un groupe de colonnes sont uniques. Ainsi elle peut être une contrainte de colonne ou une contrainte de table.
- Afin de définir la contrainte UNIQUE pour une colonne lors de la création de la table, on utilise cette syntaxe :

```
nom_colonne type_donnee UNIQUE;
```

Les contraintes d'intégrité MySQL : UNIQUE

- Pour définir une contrainte UNIQUE pour plusieurs colonnes (contrainte de table), on utilise la syntaxe suivante :

```
CREATE TABLE nom_table(  
    nom_colonne1 definition_colonne,  
    nom_colonne2 definition_colonne,  
    ...  
    UNIQUE (nom_colonne1, nom_colonne2,...)  
) ;
```

- Si on définit la contrainte UNIQUE sans la nommer, MySQL génère automatiquement un nom pour celle-ci. Pour définir une contrainte UNIQUE avec un nom, on utilise cette syntaxe :

```
[CONSTRAINT nom_contrainte]  
UNIQUE (liste_colonne)
```

5 - CHEC

- La contrainte **CHECK** assure que les valeurs stockées dans une colonne ou un groupe de colonnes satisfont à une expression booléenne. Elle est ainsi considérée comme une contrainte de colonne et une contrainte de table.
- A partir de la version MySQL 8.0.16, la contrainte CHECK de table et de colonne est prise en charge lors de la création de la table, et ce pour tous les moteurs de stockage.
- Voici la syntaxe à utiliser :

```
[CONSTRAINT [nom_contrainte]] CHECK(expression) [[NOT] ENFORCED]
```

- Où il faudra:
 - Spécifier le nom de la contrainte à créer. Si le nom de la contrainte est omis, MySQL génère automatiquement un nom avec la convention suivante : **nom_table_chk_n** (n étant un entier).
 - Spécifiez une expression booléenne qui doit être évaluée à **TRUE** ou **FALSE** pour chaque ligne de la table. Si l'expression est évaluée à **FALSE**, les valeurs entrées violent la contrainte.
 - En option, spécifier la clause d'application pour indiquer si la contrainte de vérification est appliquée :
 - Utilisez **ENFORCED** ou omettez simplement la clause **ENFORCED** pour créer et appliquer la contrainte.
 - Utilisez **NOT ENFORCED** pour créer la contrainte mais ne pas l'appliquer.

5 - CHECK (suite) - Exemple 1 : Contrainte de colonne

- Contrainte MySQL CHECK - exemple de contrainte de colonne

```
CREATE TABLE Produits (
    Num_Produit VARCHAR(18) PRIMARY KEY,
    description VARCHAR(40),
    cout DECIMAL(10,2) NOT NULL CHECK (cout >= 0),
    prix DECIMAL(10,2) NOT NULL CHECK (prix >= 0)
);
```

- La contrainte check assure que lors de l'insertion, la valeur de chacune des colonnes cout et prix sera ≥ 0 .

5 - CHECK (suite) - Exemple 2 : Contrainte de table

- Contrainte MySQL CHECK - exemple de contrainte de table

```
CREATE TABLE Produits (
    Num_Produit VARCHAR(18) PRIMARY KEY,
    description VARCHAR(40),
    cout DECIMAL(10,2) NOT NULL CHECK (cout >= 0),
    prix DECIMAL(10,2) NOT NULL CHECK (prix >= 0),
    CONSTRAINT produits_chk_prix_et_cout
        CHECK(price >= cost)
);
```

- Dans cette requête, nous avons ajouté une contrainte CHECK nommée **produits_chk_prix_et_cout** comme contrainte de table, et qui assure que la valeur du prix doit toujours être supérieure à la valeur du coût pour chaque élément de la table produit.



CHAPITRE 1

Créer une Base de Données

1. Création des Bases de Données
2. Choix de moteur
3. Création des tables
4. Définition des colonnes
5. Typage des colonnes
6. Contraintes d'intégrité
7. **Manipulation d'objet table**

Commande DROP TABLE

- La commande DROP TABLE supprime définitivement une table et ses données de la base de données. Dans MySQL, nous pouvons également supprimer plusieurs tables à la fois en suivant la syntaxe de base :

```
DROP [TEMPORARY] TABLE [IF EXISTS] nom_table1, nom_table2  
...
```

- L'option **TEMPORARY** permet de supprimer uniquement les tables temporaires. Ceci prévient que l'utilisateur supprime accidentellement des tables non temporaires.

Commande DROP TABLE

- L'option **IF EXISTS** fait que MySQL supprime une table uniquement si elle existe. Si vous supprimez une table inexistante avec l'option **IF EXISTS**, MySQL génère une **NOTE**, qui peut être récupérée à l'aide de l'instruction **SHOW WARNINGS**.
- Notez que l'instruction **DROP TABLE** supprime uniquement les tables. Il ne supprime pas les privilèges utilisateur spécifiques associés aux tables. Par conséquent, si vous créez une table avec le même nom que celle supprimée, MySQL appliquera les privilèges existants à la nouvelle table, ce qui peut poser un risque de sécurité.
- Enfin, pour exécuter **DROP TABLE**, l'utilisateur doit disposer des privilèges **DROP** pour la table qu'il va supprimer.

Commande ALTER TABLE

- ALTER TABLE est la commande utilisée pour changer la structure d'une table :
 - Ajouter, modifier, renommer et supprimer une colonne
 - Renommer une table
 - Ajouter et supprimer une contrainte d'intégrité.

Ajouter une ou plusieurs colonnes à une table :

```
ALTER TABLE nom_table
    ADD nouvelle_colonne1 [definition1],
    ADD nouvelle_colonne2 [definition2],
    ...;
```

Commande ALTER TABLE : Ajouter une ou plusieurs colonnes à une table (suite)

- Exemples :
 - Ajouter une colonne « num_produit » à la table Produits :

```
ALTER TABLE Produits (
    ADD num_Produit VARCHAR(18)
);
```

- Ajouter plusieurs colonnes à la table Produits :

```
ALTER TABLE Produits (
    ADD Num_Produit VARCHAR(18) ,
    cout DECIMAL(10,2) NOT NULL CHECK (cout >= 0)
);
```

Commande ALTER TABLE : Modifier une ou plusieurs colonnes d'une table

- Cette modification peut porter sur le type ainsi que les contraintes associées à cette colonne en utilisant la syntaxe suivante :

```
ALTER TABLE nom_table
    MODIFY nouvelle_colonne1 [definition1],
    MODIFY nouvelle_colonne2 [definition2],
    ...;
```

Commande ALTER TABLE : Modifier une ou plusieurs colonnes d'une table (suite)

Exemples 1 :

- Modifier une colonne de la table « Produits » :
 - La colonne « num_produit » est déjà définie comme suit : **num_Produit VARCHAR(18)**.

```
ALTER TABLE Produits (
    MODIFY num_Produit VARCHAR(20) PRIMARY KEY );
```

- Permet de modifier cette colonne de telle façon à :
 - Augmenter la taille du varchar(18) à varchar (20).
 - Définir cette colonne comme clé primaire de la table « Produits » via l'ajout de la contrainte **PRIMARY KEY**.

Commande ALTER TABLE : Modifier une ou plusieurs colonnes d'une table (Suite)

Exemple 2 :

- Afin de modifier deux colonnes de la table « Produits » :

```
ALTER TABLE Produits (
    MODIFY num_Produit VARCHAR(20) PRIMARY KEY,
    MODIFY cout DECIMAL(10,2) CHECK (cout >= 0)
);
```

- Cette commande a permis de :
 - Changer le type de la colonne num_Produit et la définir comme clé primaire.
 - Ajouter la contrainte CHECK sur la colonne Cout.

Commande ALTER TABLE : Renommer une ou plusieurs colonnes d'une table

- Afin de renommer une colonne d'une table, utilise la syntaxe suivante :

```
ALTER TABLE nom_table  
  
    CHANGE COLUMN nom_original nouveau_nom [definition] ;
```

Exemple :

- Pour renommer la colonne « cout » en « cout_produit » on exécute la commande suivante:

```
ALTER TABLE Produits (  
  
    CHANGE COLUMN cout cout_produit DECIMAL(10,2) CHECK (cout >= 0)  
  
);
```

Commande ALTER TABLE : Supprimer une colonne d'une table

- Afin de supprimer une colonne d'une table, on utilise la syntaxe suivante :

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

Exemple :

- Supprimer la colonne « description » de la table « Produits»:

```
ALTER TABLE Produits
    DROP COLUMN description ;
```

Commande ALTER TABLE : Renommer une table

- Pour renommer une table, on utilise la syntaxe suivante :

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

Exemple :

- Renommer la table « Produits » en « Articles » :

```
ALTER TABLE Produits
RENAME TO Articles ;
```

Commande ALTER TABLE : Ajouter et supprimer une contrainte de table

PRIMARY KEY :

- Ajouter une clé primaire:

```
ALTER TABLE nom_table
ADD PRIMARY KEY(column_list);
```

- Supprimer une clé primaire:

```
ALTER TABLE nom_table
DROP PRIMARY KEY;
```

Commande ALTER TABLE : Ajouter et supprimer une contrainte de table

FOREIGN KEY :

- Ajouter une clé étrangère :

```
ALTER TABLE nom_table
ADD CONSTRAINT constraint_name
FOREIGN KEY (column_name, ...)
REFERENCES parent_table (column_name,...);
```

- Supprimer une clé étrangère :

```
ALTER TABLE nom_table
DROP FOREIGN KEY nom_contrainte;
```

Commande ALTER TABLE : Ajouter et supprimer une contrainte de table

UNIQUE :

- Ajouter la contrainte UNIQUE :

```
ALTER TABLE nom_table
ADD CONSTRAINT nom_contrainte
UNIQUE (liste_colonnes);
```

- Supprimer la contrainte UNIQUE : Il faut supprimer l'index que MYSQL crée pour cette contrainte et qui porte le même nom.

```
ALTER TABLE nom_table
DROP INDEX nom_contrainte;
```



CHAPITRE 2

Réaliser des requêtes SQL

Ce que vous allez apprendre dans ce chapitre :

- Maitriser les principales requêtes SQL
- Gérer les fonctions du SGBD





CHAPITRE 2

Réaliser des requêtes SQL

1. Requêtes LMD
2. Requêtes de sélection
3. Expression du SGBD
4. Fonctions d'agrégation du SGBD
5. Sous requêtes
6. Requêtes de l'union
7. Jointures

Dans ce qui suit, nous allons apprendre à utiliser les requêtes SQL –LMD : Langage de manipulation de données(LMD).

INSERT :

- La commande INSERT permet d'insérer une ou plusieurs lignes de données dans une table selon la syntaxe suivante :

```
INSERT INTO nom_table(colonne1,colonne2,...)
VALUES (valeur1,valeur2,...);
```

On doit spécifier :

- Le nom de la table ainsi qu'une liste de ses colonnes séparées par des virgules entre parenthèses.
- La liste de valeurs à insérer dans ces colonnes, séparées par des virgules.

Il faut noter que :

- Le nombre de colonnes et de valeurs doit être le même. De plus, les positions des colonnes doivent correspondre aux positions de leurs valeurs.
- Pour les colonnes qui ne figurent pas dans la liste spécifiée MySQL insert les valeurs par default ou alors NULL si elles ne sont pas spécifiées.
- Si une colonne est définie comme une colonne AUTO_INCREMENT, MySQL génère un entier séquentiel chaque fois qu'une ligne est insérée dans la table.

INSERT : Insérer plusieurs lignes

- Pour insérer plusieurs lignes dans une table à l'aide d'une seule instruction **INSERT**, on utilise la syntaxe suivante :

```
INSERT INTO nom_table(c1,c2,...)
VALUES (v01,v02,...),
(v11,v22,...),
..
(vn1,vn2,...)
;
```

- Les lignes à ajouter dans la table sont séparées par des virgules dans la clause VALUES.

INSERT : Exemples

- Soit la table « Produit » créée à partir de la requête suivante :

```
CREATE TABLE Produits (
    Num_Produit VARCHAR(18) PRIMARY KEY,
    description VARCHAR(40) DEFAULT 'Non specifie',
    cout DECIMAL(10,2) NOT NULL CHECK (cout >= 0),
    prix DECIMAL(10,2) NOT NULL CHECK (prix >= cout),
    Date_ajout DATE
);
```

- 1 - On veut ajouter le produit suivant : Numéro du Produit est P12, Son prix est 14 et le cout 12.
- Voici la commande à exécuter :

```
INSERT INTO Produits(num_produit,cout,prix)
VALUES ('P12',12,14);
```

02 - Réaliser des requêtes SQL

Requêtes LMD



INSERT : Exemples

- Ajout de plusieurs lignes : On veut ajouter les produits suivant :
 - Numéro du Produit est P13, Le cout: 120, le prix 140, ajouté le 01/01/2022.
 - Numéro du Produit est P100, Description: Laptop, le cout: 120, le prix 140, ajouté aujourd'hui.
- Voici la commande à exécuter :

```
INSERT INTO Produits(num_produit,description,cout,prix,date_ajout)
VALUES ('P13','','120,140,'2022-01-01'),
       ('P100','Laptop',5000,6000,CURRENT_DATE)
;
```

02 - Réaliser des requêtes SQL

Requêtes LMD



WEBFORCE
BE THE CHANGE

INSERT : Exemples

- Voici le contenu de la table après l'exécution de ces requêtes :

```
mysql> select * from Produits;
+-----+-----+-----+-----+-----+
| Num_Produit | description | cout      | prix     | Date_ajout |
+-----+-----+-----+-----+-----+
| P100        | Laptop       | 5000.00   | 6000.00  | 2022-01-14 |
| P12         | Non specifie | 12.00     | 14.00    | NULL        |
| P13         |               | 120.00    | 140.00   | 2021-01-01 |
+-----+-----+-----+-----+-----+
3 rows in set (0.00 sec)
```

- On remarque que :

- La description non spécifiée dans le premier insert a été remplacée par la valeur par défaut : 'Non spécifié'
- Pour insérer une valeur de date dans une colonne, il faut utiliser le format 'YYYY-MM-DD' ou :
 - YYYY représente une année à quatre chiffres.
 - MM représente un mois à deux chiffres.
 - DD représente un jour à deux chiffres.
- Nous avons utilisé la fonction CURRENT_DATE() qui retourne la date système.

UPDATE :

- L'instruction **UPDATE** permet de mettre à jour les données d'une table. Elle sert à modifier les valeurs dans une ou plusieurs colonnes d'une seule ligne ou de plusieurs lignes, selon la syntaxe suivante :

```
UPDATE nom_table  
SET  
    nom_colonne1 = expr1,  
    nom_colonne2 = expr2,  
    ...  
[WHERE  
    condition];
```

- **On doit spécifier :**
 - Le nom de la table dont on souhaite mettre à jour les données.
 - La colonne qui va être mise à jour et la nouvelle valeur dans la clause SET. Pour mettre à jour les valeurs dans plusieurs colonnes, on utilise une liste d'affectations séparées par des virgules.
 - En option, Définir une condition dans la clause WHERE. Si cette étape est omise, l'instruction UPDATE modifiera **toutes les lignes de la table**.

02 - Réaliser des requêtes SQL

Requêtes LMD



UPDATE : Exemples

- Rappelons la table Produits :

```
mysql> select * from Produits;
+-----+-----+-----+-----+-----+
| Num_Produit | description | cout      | prix     | Date_ajout |
+-----+-----+-----+-----+-----+
| P100        | Laptop       | 5000.00   | 6000.00  | 2022-01-14 |
| P12         | Non specifie | 12.00     | 14.00    | NULL        |
| P13         |               | 120.00    | 140.00   | 2021-01-01 |
+-----+-----+-----+-----+-----+
3 rows in set (0.00 sec)
```

- Modifier la date d'ajout du produit P12 en 31/12/2021 :

```
UPDATE Produits
SET
  Date_ajout = '2021-12-31'
WHERE Num_Produit='P12';
```

UPDATE : Exemples

- Modifier les données de telle façon que Description ='Non specifie' et Prix = 1.5*cout :

```
UPDATE Produits
SET
    Description = 'Non specifie',
    Prix = 1.5*Cout ;
WHERE Num_Produit='P12' ;
```

- Résultat suivant les deux modifications :

```
mysql> select * from Produits;
+-----+-----+-----+-----+-----+
| Num_Produit | description | cout     | prix      | Date_ajout |
+-----+-----+-----+-----+-----+
| P100        | Non specifie | 5000.00 | 7500.00  | 2022-01-14 |
| P12         | Non specifie | 12.00   | 18.00    | 2021-12-31 |
| P13         | Non specifie | 120.00  | 180.00   | 2021-01-01 |
+-----+-----+-----+-----+-----+
```

DELETE :

- L'instruction **DELETE** permet de supprimer une ou plusieurs lignes d'une table en utilisant la syntaxe suivante :

```
DELETE FROM nom_table  
WHERE Conditions;
```

- Cette instruction permet d'utiliser, en option, une condition pour spécifier les lignes à supprimer dans la clause WHERE, si cette dernière est omise, l'instruction DELETE supprimera toutes les lignes de la table.
- Pour une table qui a une contrainte de clé étrangère, lorsque vous supprimez des lignes de la table parent, les lignes de la table enfant peuvent aussi être supprimées automatiquement à l'aide de l'option ON DELETE CASCADE.

DELETE : Exemples

- De la table Produit :

```
mysql> select * from Produits;
+-----+-----+-----+-----+
| Num_Produit | description | cout      | prix      | Date_ajout |
+-----+-----+-----+-----+
| P100        | Non specifie | 5000.00   | 7500.00   | 2022-01-14 |
| P12         | Non specifie | 12.00     | 18.00     | 2021-12-31 |
| P13         | Non specifie | 120.00    | 180.00    | 2021-01-01 |
+-----+-----+-----+-----+
```

- On veut supprimer les Produits dont le cout <=12 .

```
DELETE FROM Produits
WHERE cout<=12;
```

- Voici la table après l'exécution de cette requête

```
mysql> DELETE FROM Produits
      -> WHERE cout<=12;
Query OK, 1 row affected (0.01 sec)

mysql> select * from Produits;
+-----+-----+-----+-----+
| Num_Produit | description | cout      | prix      | Date_ajout |
+-----+-----+-----+-----+
| P100        | Non specifie | 5000.00   | 7500.00   | 2022-01-14 |
| P13         | Non specifie | 120.00    | 180.00    | 2021-01-01 |
+-----+-----+-----+-----+
2 rows in set (0.00 sec)
```



CHAPITRE 2

Réaliser des requêtes SQL

1. Requêtes LMD
2. **Requêtes de sélection**
3. Expression du SGBD
4. Fonctions d'agrégation du SGBD
5. Sous requêtes
6. Requêtes de l'union
7. Jointures

02 - Réaliser des requêtes SQL

Requêtes de sélection



SELECT :

- L'instruction **SELECT** permet de consulter les données et de les présenter triées et/ou regroupées suivant certains critères.
- L'instruction **SELECT** basique est comme suit :

```
SELECT [Liste_select]  
FROM nom_table;
```

- Dans cette syntaxe, il faut :
 - Spécifier une ou plusieurs colonnes à partir desquelles on veut sélectionner des données après le mot-clé SELECT : Pour sélectionner toute les colonnes de la table, on utilise « `SELECT *` » , Sinon on spécifie les noms des colonnes séparés par une virgule (,).
 - Spécifier le nom de la table à partir de laquelle on veut sélectionner des données après le mot-clé FROM.
- **N.B** : Lors de l'exécution de l'instruction SELECT, MySQL évalue la clause FROM avant la clause SELECT :



02 - Réaliser des requêtes SQL

Requêtes de sélection



SELECT :

- Exemples de requêtes SELECT simples :

```
mysql> select * from Produits;
+-----+-----+-----+-----+-----+
| Num_Produit | description | cout | prix | Date_ajout |
+-----+-----+-----+-----+-----+
| P100 | Laptop | 5000.00 | 6000.00 | 2022-01-14 |
| P12 | Non specifie | 12.00 | 14.00 | NULL |
| P13 | | 120.00 | 140.00 | 2022-01-01 |
+-----+-----+-----+-----+
3 rows in set (0.00 sec)

mysql> commit;
Query OK, 0 rows affected (0.00 sec)

mysql> Select Num_produit, Date_ajout from Produits;
+-----+-----+
| Num_produit | Date_ajout |
+-----+-----+
| P100 | 2022-01-14 |
| P12 | NULL |
| P13 | 2022-01-01 |
+-----+-----+
3 rows in set (0.00 sec)
```

02 - Réaliser des requêtes SQL

Requêtes de sélection

SELECT :

- Options de la Requête SELECT :
 - La requête SQL plus avancées prend la forme suivante :

```
SELECT [DISTINCT] Liste_Select  
FROM   Liste_Tables  
WHERE  Liste_Conditions_Recherche  
GROUP BY Liste_regroupement  
HAVING Liste_Conditions_regroupement  
ORDER BY liste_Tri
```

- MySQL exécute cette requête dans cet ordre :



02 - Réaliser des requêtes SQL

Requêtes de sélection

DISTINCT :

- DISTINCT est une option qui permet de supprimer les lignes en double.

```
1 •  SELECT description FROM dbtest.produits;
2
```

Result Grid | Filter Rows: Export: | Wrap Cell

description
Laptop
Non specifie
Laptop

```
1 •  SELECT distinct description FROM dbtest.produits;
2
```

Result Grid | Filter Rows: Export: | Wrap Cell

description
Laptop
Non specifie

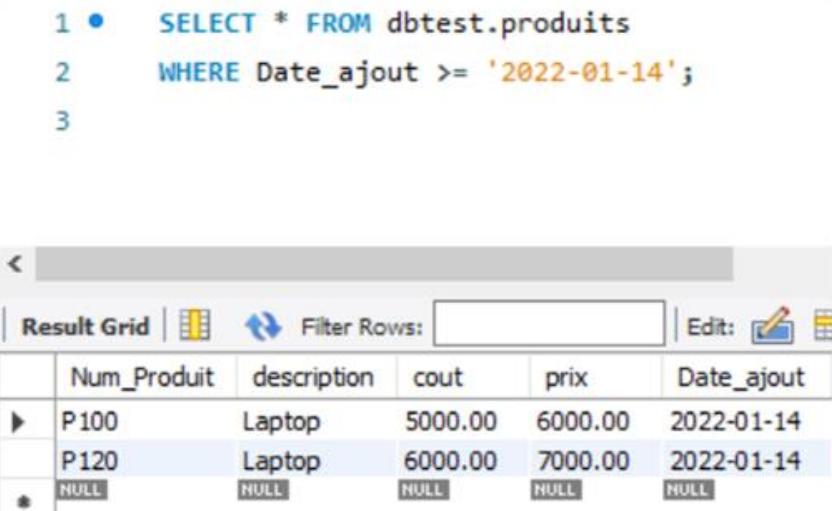
02 - Réaliser des requêtes SQL

Requêtes de sélection

WHERE :

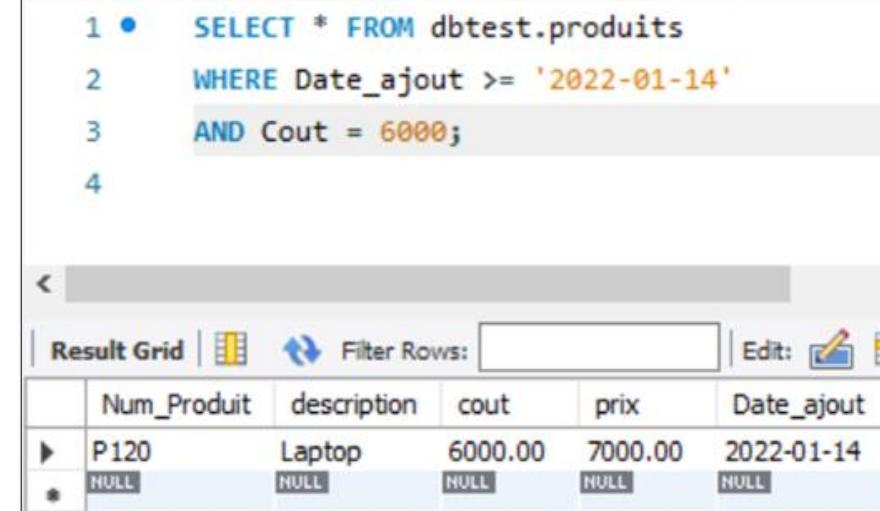
- WHERE définit la liste de conditions que les données recherchées doivent vérifier. La condition de recherche est une combinaison d'une ou plusieurs expressions utilisant l'opérateur logique **AND**, **OR** et **NOT**. L'instruction **SELECT** inclura toute ligne qui satisfait la condition de recherche dans le jeu de résultats.
- WHERE est aussi utilisé dans **UPDATE** ou **DELETE** pour spécifier les lignes à mettre à jour ou à supprimer.

```
1 •  SELECT * FROM dbtest.produits
2      WHERE Date_ajout >= '2022-01-14';
3
```



	Num_Produit	description	cout	prix	Date_ajout
▶	P100	Laptop	5000.00	6000.00	2022-01-14
▶	P120	Laptop	6000.00	7000.00	2022-01-14
*	NULL	NULL	NULL	NULL	NULL

```
1 •  SELECT * FROM dbtest.produits
2      WHERE Date_ajout >= '2022-01-14'
3          AND Cout = 6000;
4
```



	Num_Produit	description	cout	prix	Date_ajout
▶	P120	Laptop	6000.00	7000.00	2022-01-14
*	NULL	NULL	NULL	NULL	NULL

02 - Réaliser des requêtes SQL

Requêtes de sélection

GROUP BY :

- La clause **GROUP BY** regroupe un ensemble de lignes dans un ensemble de lignes récapitulatives par valeurs de colonnes ou d'expressions. La clause **GROUP BY** renvoie une ligne pour chaque groupe, ceci réduit le nombre de lignes dans le jeu de résultats.
- En pratique, on utilise souvent la clause **GROUP BY** avec des fonctions d'agrégation telles que **SUM**, **AVG**, **MAX**, **MIN** et **COUNT**. La fonction d'agrégation qui apparaît dans la clause **SELECT** fournit les informations de chaque groupe.
- Exemple :**

```
1
2 •  SELECT description, Prix-Cout from produits;
3
4
```

Result Grid | Filter Rows: [] Export: [] Wrap []

description	Prix-Cout
Laptop	1000.00
Non specifie	2.00
Laptop	1000.00
	20.00
	20.00

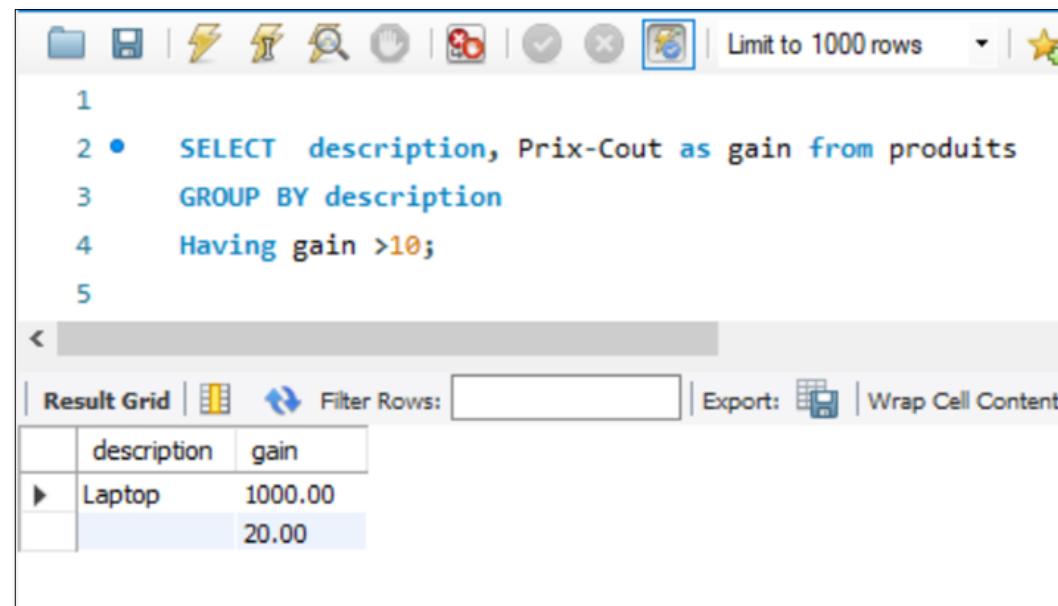
```
1
2 •  SELECT description, Prix-Cout from produits
3     GROUP BY description;
4
5
```

Result Grid | Filter Rows: [] Export: [] Wrap []

description	Prix-Cout
Laptop	1000.00
Non specifie	2.00
	20.00

HAVING :

La clause **HAVING** est utilisée dans l'instruction **SELECT** pour spécifier des conditions de filtre pour un groupe de lignes ou d'agrégats. Elle est souvent utilisée avec **GROUP BY** pour filtrer les groupes en fonction d'une condition spécifiée. Si la clause **GROUP BY** est omise, **HAVING** se comporte comme la clause **WHERE**.



The screenshot shows a MySQL Workbench interface. The query editor contains the following code:

```
1
2 •  SELECT description, Prix-Cout as gain from produits
3   GROUP BY description
4   Having gain >10;
5
```

The result grid displays the following data:

	description	gain
▶	Laptop	1000.00
		20.00

ORDER BY :

- La clause **ORDER BY** est utilisée pour trier les lignes du jeu de résultats. Elle peut porter sur plusieurs colonnes, chacune suivie, en option, de l'ordre de tri utilisé croissant **ASC** ou décroissant **DESC**. L'ordre de tri par défaut étant **ASC**.
- Exemple :

```
1
2 •   SELECT num_produit, Prix-Cout as gain from produits
3     order by gain ASC;
4
5
```

< [REDACTED]

Result Grid | Filter Rows: [REDACTED] Export: [REDACTED] Wrap Cell Content:

	num_produit	gain
▶	P12	2.00
	P13	20.00
	P14	20.00
	P100	1000.00
	P120	1000.00



CHAPITRE 2

Réaliser des requêtes SQL

1. Requêtes LMD
2. Requêtes de sélection
- 3. Expression du SGBD**
4. Fonctions d'agrégation du SGBD
5. Sous requêtes
6. Requêtes de l'union
7. Jointures

02 - Réaliser des requêtes SQL

Expression du SGBD



- Une expression se compose d'ensemble de colonnes, constantes et fonctions combinées au moyen d'opérateurs. On trouve les expressions dans les différentes parties du SELECT : en tant que colonne résultat, dans les clauses **WHERE**, **HAVING**, et **ORDER BY**.
- Il existe trois types d'expressions selon le type de données de SQL :
 - **Expressions arithmétiques**
 - **Expressions de chaînes de caractères**
 - **Expressions de dates.**
- A chaque type correspondent des opérateurs et des fonctions spécifiques.
- Afin d'utiliser des données de types différents dans la même expression, on peut utiliser les fonctions de conversion dont dispose le langage **SQL**, celle-ci permettent de convertir des chaines de caractères en date ou en nombre selon le besoin.

Les opérateurs MYSQL :

- Un opérateur est un symbole spécifiant une action exécutée sur une ou plusieurs expressions. Nous trouvons en SQL, différentes catégories d'opérateurs.
- Voici les principaux utilisables dans les requêtes de sélection.
 - **Opérateurs arithmétiques**
 - **Opérateurs de comparaison**
 - **Opérateurs logiques**

Les opérateurs MYSQL : Opérateurs arithmétiques

- Les opérateurs arithmétiques présents dans SQL sont les suivants :
 - + addition
 - - soustraction
 - * multiplication
 - / division
- Remarque : la division par 0 provoque une fin avec code d'erreur.

Les opérateurs MYSQL : Opérateurs arithmétiques

- Priorité des opérateurs :
 - Une expression arithmétique peut comporter plusieurs opérateurs. Dans ce cas, le résultat de l'expression peut varier selon l'ordre dans lequel sont effectuées les opérations. Les opérateurs de multiplication et de division sont prioritaires par rapport aux opérateurs d'addition et de soustraction. Des parenthèses peuvent être utilisées pour forcer l'évaluation de l'expression dans un ordre différent de celui découlant de la priorité des opérateurs.
 - Au moyen des opérateurs arithmétiques + et - il est possible de construire les expressions suivantes :
 - date +/- nombre : le résultat est une date obtenue en ajoutant le nombre de jours nombre à la date.
 - date2 - date1 : le résultat est le nombre de jours entre les deux dates.

Les opérateurs MYSQL : Opérateurs arithmétiques

Exemple : Calcul du Gain = Prix-Cout pour chacun des produits :

```
1
2 •   SELECT description, Prix-Cout from produits;
3
4
```

< [RETRIEVE] Filter Rows: Export: [grid] [Wrap]

	description	Prix-Cout
▶	Laptop	1000.00
	Non specifie	2.00
▶	Laptop	1000.00
		20.00
		20.00

Les opérateurs MYSQL : Opérateurs de comparaison

- Les opérateurs de comparaison testent si deux expressions sont identiques.
- Ils peuvent s'utiliser sur toutes les expressions composées de données structurées.
- Ces opérateurs sont: = (Égal à), > (Supérieur à), < (Inférieur à), >= (Supérieur ou égal à), <= (Inférieur ou égal à), <> (Différent de)

Les opérateurs MYSQL : Opérateurs logiques

- Les opérateurs logiques testent la valeur logique d'une condition. Les opérateurs logiques, comme les opérateurs de comparaison, retournent un type de données booléen de valeur TRUE ou FALSE.
- Un certain nombre d'entre eux (signalés par un * dans le tableau) sont utilisés pour comparer une valeur scalaire (unique) avec une sous requête.

Opérateur	Description
ALL (*)	TRUE si tous les éléments dun jeu de comparaisons sont TRUE.
AND	TRUE les deux expressions booléennes sont TRUE.
ANY (*)	TRUE si n'importe quel élément d'un jeu de comparaison est TRUE.
BETWEEN	TRUE si l'opérande est situé dans une certaine plage.
EXISTS (*)	TRUE si une sous-requête contient des lignes.
IN (*)	TRUE si l'opérande est égal à un élément dune liste d'expressions.
LIKE	TRUE si l'opérande correspond à un modèle.
NOT	Inverse la valeur de tout autre opérateur booléen.
OR	TRUE si l'une ou l'autre expression booléenne est TRUE.
SOME (*)	TRUE si certains éléments d'un jeu de comparaisons sont TRUE.
*	Utilisés avec des sous-requêtes

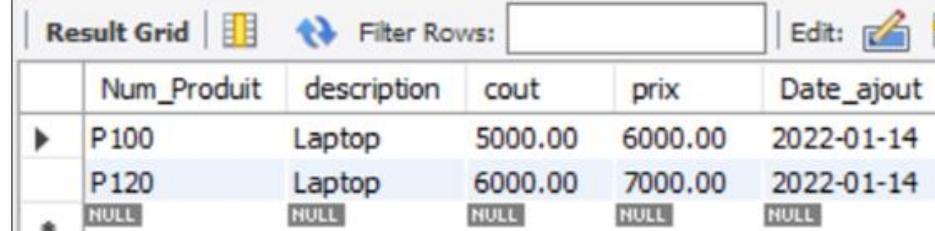
Les opérateurs MYSQL : Opérateurs logiques

Opérateur BETWEEN

- On utilise BETWEEN pour tester si une valeur est comprise entre une valeur minimale et une autre maximale. La syntaxe de l'opérateur BETWEEN : **valeur BETWEEN Minimum AND Maximum**

Exemple :

```
2 •  SELECT * from produits
3      WHERE cout BETWEEN 5000 and 6000;
4
5
```



The screenshot shows the MySQL Workbench interface with a query editor containing the following SQL code:

```
2 •  SELECT * from produits
3      WHERE cout BETWEEN 5000 and 6000;
4
5
```

Below the code, the "Result Grid" tab is selected, displaying the results of the query. The results are as follows:

	Num_Produit	description	cout	prix	Date_ajout
▶	P100	Laptop	5000.00	6000.00	2022-01-14
▶	P120	Laptop	6000.00	7000.00	2022-01-14
*	NULL	NULL	NULL	NULL	NULL

Les opérateurs MYSQL : Opérateurs logiques

Opérateur LIKE/NOT LIKE

- On Utilise l'opérateur LIKE pour tester si une valeur correspond à un modèle spécifique. l'opérateur NOT sert à annuler l'opérateur LIKE.
- Le modèle est défini en utilisant les caractères génériques suivants :

Caractère générique	Description
%	Toute chaîne de zéro caractère ou plus
-	Nimporte quel caractère à cet emplacement
[]	Tout caractère de l'intervalle ([a-f]) ou de l'ensemble spécifié ([abcdef])
[^]	Tout caractère en dehors de l'intervalle ([^a-f]) ou de l'ensemble spécifié (^abcdef)]. ^ représente le NOT

Exemples :

```
2 •  SELECT * from produits
3   WHERE description like 'L%';
4
5
< [REDACTED]
```

Result Grid | Filter Rows: Edit: Num_Produit description cout prix Date_ajout

Num_Produit	description	cout	prix	Date_ajout
P100	Laptop	5000.00	6000.00	2022-01-14
P120	Laptop	6000.00	7000.00	2022-01-14
*	NULL	NULL	NULL	NULL

La liste des produits dont la description commence par 'L'

```
2 •  SELECT * from produits
3   WHERE Num_Produit like '%2_';
4
5
< [REDACTED]
```

Result Grid | Filter Rows: Edit: Num_Produit description cout prix Date_ajout

Num_Produit	description	cout	prix	Date_ajout
P120	Laptop	6000.00	7000.00	2022-01-14
*	NULL	NULL	NULL	NULL

La liste des produits qui ont '2' dans la case avant dernière de leur Num_Produit.

Les fonctions intégrées MySQL :

- MySQL, comme les autres SGBD, propose de nombreuses fonctions intégrées qui permettent de manipuler les données utilisateurs ou les données du système.

Il existe plusieurs catégories de fonctions :

- Fonctions Mathématiques
 - Fonctions de traitement de chaînes
 - Fonctions de manipulation de dates
 - Fonctions de conversion
- La liste exhaustive des fonctions MySQL est disponible sur le lien : <https://dev.mysql.com/doc/refman/8.0/en/functions.html>

Les fonctions intégrées MYSQL : Fonctions Mathématiques

- Ce sont des fonctions ayant un ou plusieurs nombres comme arguments, et qui renvoient une valeur numérique.

Voici quelques exemples :

Fonction	Description
ABS()	Retourne la valeur absolue d'un nombre.
CEIL()	Renvoie la plus petite valeur entière supérieure ou égale au nombre d'entrée.
FLOOR()	Renvoie la plus grande valeur entière non supérieure au nombre d'entrée.
MOD()	Renvoie le reste d'un nombre divisé par un autre
ROUND()	Arrondit un nombre à un nombre spécifié de places décimales.
TRUNCATE()	Tronque un nombre à un nombre spécifié de places décimales.

Les fonctions intégrées MySQL : Fonctions de traitement de chaînes

Voici une liste contenant les fonctions de chaîne MySQL les plus utilisées qui permettent de manipuler efficacement les données de chaîne de caractères.

Name	Description
CONCAT	Concaténer deux ou plusieurs chaînes en une seule chaîne
INSTR	Renvoie la position de la première occurrence d'une sous-chaîne dans une chaîne
LENGTH	Obtenir la longueur d'une chaîne en octets et en caractères
LEFT	Obtient un nombre spécifié de caractères les plus à gauche d'une chaîne
LOWER	Convertir une chaîne en minuscule
LTRIM	Supprimer tous les espaces du début d'une chaîne
REPLACE	Recherche et remplace une sous-chaîne dans une chaîne
RIGHT	retourne un nombre spécifié de caractères les plus à droite d'une chaîne
RTRIM	Supprime tous les espaces de la fin d'une chaîne
SUBSTRING	Extraire une sous-chaîne à partir d'une position avec une longueur spécifique.
SUBSTRING_INDEX	Renvoie une sous-chaîne à partir d'une chaîne avant un nombre spécifié d'occurrences d'un délimiteur
TRIM	Supprime les caractères indésirables d'une chaîne
FIND_IN_SET	Rechercher une chaîne dans une liste de chaînes séparées par des virgules
FORMAT	Mettre en forme un nombre avec une locale spécifique, arrondi au nombre de décimales
UPPER	Convertir une chaîne en majuscule

Les fonctions intégrées MySQL : Fonctions de traitement de chaînes

Exemple :

- Pour chaque produit, Retourner les valeurs suivantes :
 - Num_Produit
 - Une colonne « résultat » qui contient : Num_Produit 'Ajoute le' Date_ajout
 - Une colonne « nouveau » qui contient : Remplacer 'P' dans Num_Produit par 'NP'.

```
7
8 • Select Num_Produit, CONCAT(Num_Produit,' ajoute le ', Date_ajout) as resultat,
9 REPLACE(Num_Produit,'P','NP') as Nouveau
10 from produits;
11
```

< [REDACTED]

Result Grid | Filter Rows: [] Export: [] Wrap Cell Content: []

	Num_Produit	resultat	Nouveau
▶	P100	P100 ajoute le 2022-01-14	NP100
	P12	NULL	NP12
	P120	P120 ajoute le 2022-01-14	NP120
	P13	P13 ajoute le 2022-01-01	NP13
	P14	P14 ajoute le 2022-01-01	NP14

Les fonctions intégrées MySQL : Fonctions de manipulation de dates

Voici une liste contenant les fonctions de manipulation de dates MySQL les plus utilisées qui permettent de manipuler efficacement les données date :

Name	Description
CURDATE	Renvoie la date actuelle.
DATEDIFF	Calcule le nombre de jours entre deux valeurs DATE.
DAY	Obtient le jour du mois d'une date spécifiée.
DATE_ADD	Ajoute une valeur de temps à la valeur de date.
DATE_SUB	Soustrait une valeur d'heure d'une valeur de date.
DATE_FORMAT	Formate une valeur de date en fonction d'un format de date spécifié.
DAYNAME	Obtient le nom d'un jour de la semaine pour une date spécifiée.
DAYOFWEEK	Renvoie l'index des jours de la semaine pour une date.
EXTRACT	Extrait une partie d'une date.
LAST_DAY	Renvoie le dernier jour du mois d'une date spécifiée
NOW	Renvoie la date et l'heure actuelles d'exécution de l'instruction.
MONTH	Renvoie un entier qui représente un mois d'une date spécifiée.
STR_TO_DATE	Convertit une chaîne en une valeur de date et d'heure basée sur un format spécifié.
SYSDATE	Renvoie la date actuelle.
TIMEDIFF	Calcule la différence entre deux valeurs TIME ou DATETIME.
TIMESTAMPDIFF	Calcule la différence entre deux valeurs DATE ou DATETIME.
WEEK	Renvoie le numéro de semaine d'une date
WEEKDAY	Renvoie un index des jours de la semaine pour une date.
YEAR	Renvoie l'année pour une date spécifiée

Les fonctions intégrées MySQL : Fonctions de manipulation de dates

Exemple :

- La liste des produits, leur date d'ajout, l'année d'ajout, Le jour de la semaine, et depuis combien de jours ils ont été ajoutés.

```
1
2 •  SELECT Num_Produit, Date_ajout, Year(Date_ajout) as Annee, DAYNAME(Date_ajout) as Jour,
3     DATEDIFF(SYSDATE(), Date_ajout) as Depuis_j
4   from produits
5 ;
```

Result Grid | Filter Rows: Export: Wrap Cell Content:

	Num_Produit	Date_ajout	Annee	Jour	Depuis_j
▶	P100	2022-01-14	2022	Friday	2
	P12	NULL	NULL	NULL	NULL
	P120	2022-01-14	2022	Friday	2
	P13	2022-01-01	2022	Saturday	15
	P14	2022-01-01	2022	Saturday	15

Les fonctions intégrées MYSQL : Fonctions de conversion

- **TO_CHAR(nombre,format)** - Renvoie la chaîne de caractères en obtenue en convertissant nombre en fonction de format.
- **TO_CHAR(date,format)** - Renvoie conversion d'une date en chaîne de caractères. Le format indique quelle partie de la date doit apparaître.
- **TO_DATE(chaîne,format)** - Permet de convertir une chaîne de caractères en donnée de type date. Le format est identique à celui de la fonction TO_CHAR.
- **TO_NUMBER(chaine)** - Convertit chaine en sa valeur numérique.



CHAPITRE 2

Réaliser des requêtes SQL

1. Requêtes LMD
2. Requêtes de sélection
3. Expression du SGBD
4. **Fonctions d'agrégation du SGBD**
5. Sous requêtes
6. Requêtes de l'union
7. Jointures

02 - Réaliser des requêtes SQL

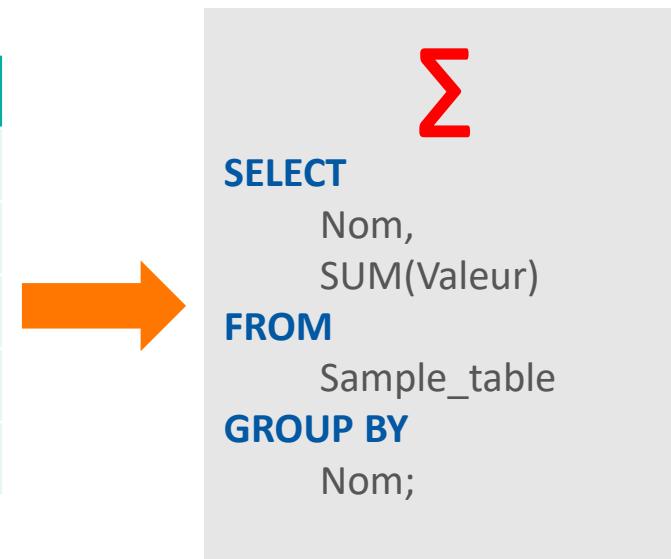
Fonctions d'agrégation du SGBD

- Une fonction d'agrégation effectue un calcul sur plusieurs valeurs et renvoie une seule valeur.
- Les fonctions d'agrégation les plus utilisées : SUM, AVG, COUNT, MAX et MIN.
- Les fonctions d'agrégation sont souvent utilisées avec la clause GROUP BY pour calculer une valeur agrégée pour chaque groupe, par exemple la valeur moyenne par groupe ou la somme des valeurs dans chaque groupe.

La fonction SUM() :

- La fonction SUM() retourne la somme des valeurs d'une colonne.
- L'image suivante illustre l'utilisation de la fonction d'agrégation SUM() avec une clause GROUP BY :

Nom	Valeur
A	10
A	20
B	40
C	20
C	50



The diagram illustrates the process of using the SUM() function with GROUP BY. It starts with a sample table on the left, followed by a large orange arrow pointing to a central SQL query box. This query box contains:
Σ
SELECT
Nom,
SUM(Valeur)
FROM
Sample_table
GROUP BY
Nom;

Nom	SUM (Valeur)
A	30
B	40
C	70

02 - Réaliser des requêtes SQL

Fonctions d'agrégation du SGBD

La fonction SUM() :

Exemple :

```
12 •   select description, Sum(cout) from Produits
13     group by description;
14
15
```

< [REDACTED]

Result Grid | Filter Rows: [] Export: [] Wrap

	description	Sum(cout)
▶	Laptop	11000.00
	Non specifie	12.00
		140.00

La fonction AVG() :

- La fonction AVG() retourne la moyenne des valeurs d'une colonne.

Exemple :

```
18 •      select description, AVG(cout) from Produits
19          group by description;
20
```

Result Grid | Filter Rows: Export: Wrap Cell

	description	AVG(cout)
▶	Laptop	5500.000000
▶	Non specifie	12.000000
▶	Accessoires	70.000000

02 - Réaliser des requêtes SQL

Fonctions d'agrégation du SGBD

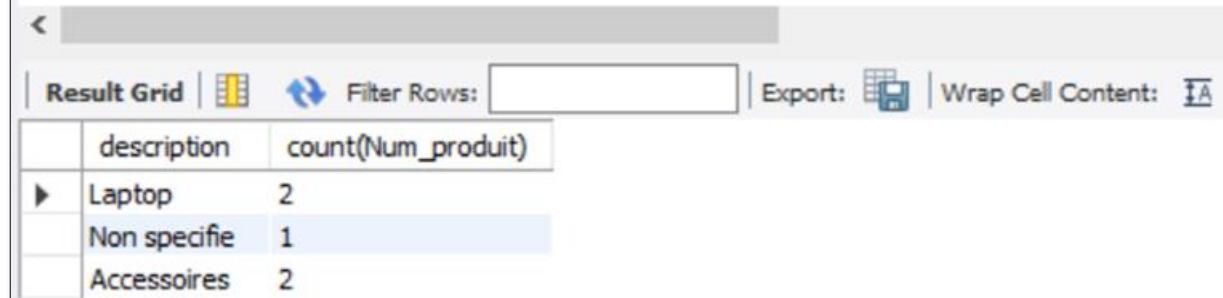
La fonction COUNT() :

- La fonction COUNT() retourne le nombre d'enregistrements sélectionnés.

Exemple :

```
18 •    select description, count(Num_produit) from Produits
19      group by description;
```

```
20
21
```



The screenshot shows a MySQL Workbench interface. At the top, there is a code editor with the following SQL query:

```
18 •    select description, count(Num_produit) from Produits
19      group by description;
```

Below the code editor is a result grid. The grid has two columns: "description" and "count(Num_produit)". The data is as follows:

	description	count(Num_produit)
▶	Laptop	2
	Non specifie	1
	Accessoires	2

02 - Réaliser des requêtes SQL

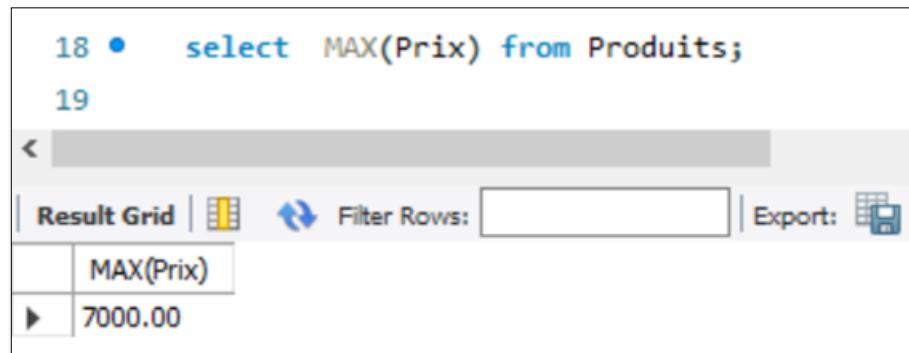
Fonctions d'agrégation du SGBD

La fonction MAX/MIN :

- Les fonctions **MAX()** et **MIN()** retournent respectivement le maximum et le minimum des valeurs des enregistrements sélectionnés.

Exemple :

```
18 •   select MAX(Prix) from Produits;
19
```

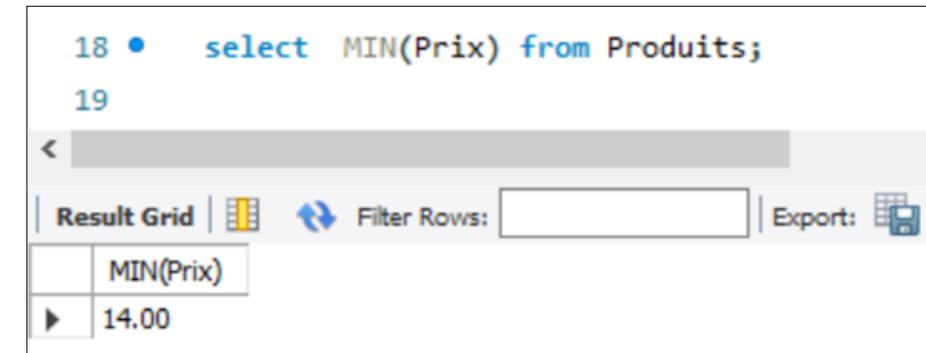


The screenshot shows a database query results grid. The query is: "select MAX(Prix) from Produits;". The result is a single row with one column labeled "MAX(Prix)" containing the value "7000.00". Below the grid are standard navigation buttons: back, forward, and search/filter.

MAX(Prix)
7000.00

MAX (Prix)

```
18 •   select MIN(Prix) from Produits;
19
```



The screenshot shows a database query results grid. The query is: "select MIN(Prix) from Produits;". The result is a single row with one column labeled "MIN(Prix)" containing the value "14.00". Below the grid are standard navigation buttons: back, forward, and search/filter.

MIN(Prix)
14.00

MIN (Prix)



CHAPITRE 2

Réaliser des requêtes SQL

1. Requêtes LMD
2. Requêtes de sélection
3. Expression du SGBD
4. Fonctions d'agrégation du SGBD
5. **Sous requêtes**
6. Requêtes de l'union
7. Jointures

02 - Réaliser des requêtes SQL

Sous requêtes



- Une sous-requête est une requête imbriquée dans une autre requête telle que SELECT, INSERT, UPDATE ou DELETE.
- Une sous-requête est appelée « requête interne » tandis que la requête qui contient la sous-requête est appelée une requête externe. La requête dite interne est évaluée pour chaque ligne de la requête externe.
- Une sous-requête peut être utilisée partout où une expression est utilisée et doit être fermée entre parenthèses.

Exemples :

- La table Produits :

	Num_Produit	description	cout	prix	Date_ajout	Max(Prix)
▶	P100	Laptop	5000.00	6000.00	2022-01-14	6000.00
	P12	Non specifie	12.00	14.00	NULL	14.00
	P120	Laptop	6000.00	7000.00	2022-01-14	7000.00
	P13	Accessoires	120.00	140.00	2022-01-01	140.00
	P14	Accessoires	20.00	40.00	2022-01-01	40.00

02 - Réaliser des requêtes SQL

Sous requêtes



WEBFORCE
BE THE CHANGE

Sous-Requête Mono-ligne :

- On utilise des opérateurs de comparaison, par exemple =, >, < pour comparer une seule valeur renvoyée par la sous-requête avec l'expression dans la clause WHERE.
- Afficher le Produit ayant le Prix minimale :
 - Select Num_produit, Description, Prix from Produits
 - where Prix = (Select Min(prix) from Produits)

```
18 • select Num_produit, Description, Prix from Produits
19      where Prix = (Select Min(prix) from Produits);
20
```

< [REDACTED]

Result Grid | Filter Rows: [] Export: [] Wrap Cell Content:

	Num_produit	Description	Prix
▶	P12	Non specifie	14.00

02 - Réaliser des requêtes SQL

Sous requêtes



Sous-Requête avec IN/NOT IN:

- Si une sous-requête renvoie plusieurs valeurs, on peut utiliser d'autres opérateurs tels que l'opérateur IN ou NOT IN dans la clause WHERE.
- Afficher la liste des produits de la table Produits qui existent sur la table Sales :
 - **Select * from Produits**
 - **where Num_produit IN (Select Num_produits from Sales)**

```
21 •   select * from Produits
22     where Num_produit IN (Select Num_produits from Sales);
23
```

The screenshot shows a MySQL Workbench interface. At the top, there is a code editor window containing the following SQL query:

```
21 •   select * from Produits
22     where Num_produit IN (Select Num_produits from Sales);
23
```

Below the code editor is a result grid titled "Result Grid". The grid has the following columns: Num_Produit, description, cout, prix, and Date_ajout. The data is as follows:

	Num_Produit	description	cout	prix	Date_ajout
▶	P100	Laptop	5000.00	6000.00	2022-01-14
	P12	Non specifie	12.00	14.00	NULL
	P120	Laptop	6000.00	7000.00	2022-01-14

Sous-Requête avec ALL/ANY

- On peut utiliser les opérateurs **ANY**, ou **ALL** pour comparer la valeur d'une expression avec les valeurs d'un attribut d'une requête interne.
- Le mot-clé **ALL** spécifie tous les éléments retournés tandis que le mot-clé **ANY** spécifie l'un d'entre eux.
- Afficher les produits dont le cout est Inferieur à tous les prix des produits.
 - **Select Num_produit, Description from Produits where cout < ALL (Select Prix from Produits)**

```
44 •  Select Num_produit, Description from Produits
45      where cout < ALL (Select Prix from Produits);
46
```

Result Grid | Filter Rows: [] Export: [] Wrap Cell Content:

	Num_produit	Description
▶	P12	Non specifie

Sous-Requête avec ALL/ANY

- On peut utiliser les opérateurs **ANY**, ou **ALL** pour comparer la valeur d'une expression avec les valeurs d'un attribut d'une requête interne.
- Le mot-clé **ALL** spécifie tous les éléments retournés tandis que le mot-clé **ANY** spécifie l'un d'entre eux.
- Afficher les Produits dont le prix est inférieur à un des couts de produits :
 - **Select Num_produit, Description from Produits**
 - **where prix < ANY (Select cout from Produits)**

```
44 • Select Num_produit, Description from Produits
45      where prix < ANY (Select cout from Produits);
46
```

Result Grid | Filter Rows: [] | Export: | Wrap Cell Content: []

	Num_produit	Description
▶	P12	Non specifie
	P13	Accessoires
	P14	Accessoires

02 - Réaliser des requêtes SQL

Sous requêtes



WEBFORCE
BE THE CHANGE

Sous-Requête avec EXISTS/NOT EXISTS

- Lorsqu'une sous-requête est utilisée avec l'opérateur EXISTS ou NOT EXISTS, une sous-requête renvoie une valeur booléenne TRUE ou FALSE.
- Afficher les produits avec une colonne supplémentaire indiquant si le produit est présent la table Sales :
 - Select P.Num_produit, P.Description,
 - EXISTS(Select * from Sales S Where P.Num_produit = S.Num_Produit) as existe
 - From Produits P

```
25 •   Select P.Num_produit, P.Description,  
26   EXISTS(Select * from Sales S  
27     Where P.Num_produit = S.Num_Produit) as existe  
28   From Produits P;
```

	Num_produit	Description	existe
▶	P100	Laptop	1
	P12	Non specifie	1
	P120	Laptop	1
	P13	Accessoires	1
	P14	Accessoires	1



CHAPITRE 2

Réaliser des requêtes SQL

1. Requêtes LMD
2. Requêtes de sélection
3. Expression du SGBD
4. Fonctions d'agrégation du SGBD
5. Sous requêtes
6. **Requêtes de l'union**
7. Jointures

- La combinaison des résultats des requêtes consiste à transformer deux ou plusieurs jeux de résultat en un seul.
- Les jeux de résultats combinés doivent tous avoir la même structure : Même nombre de colonnes et même types de données.
- Il existe trois types de combinaison : UNION (UNION ALL), INTERSECT, MINUS.

Opérateur UNION

- L'opérateur UNION permet de combiner deux ou plusieurs ensembles de résultats de requêtes en un seul ensemble de résultats. Voici la syntaxe d'utilisation de l'opérateur UNION :

```
SELECT column_list1  
UNION [DISTINCT | ALL]  
SELECT column_list2  
UNION [DISTINCT | ALL]  
SELECT column_list3  
...
```

02 - Réaliser des requêtes SQL

Requêtes de l'union

Opérateur UNION

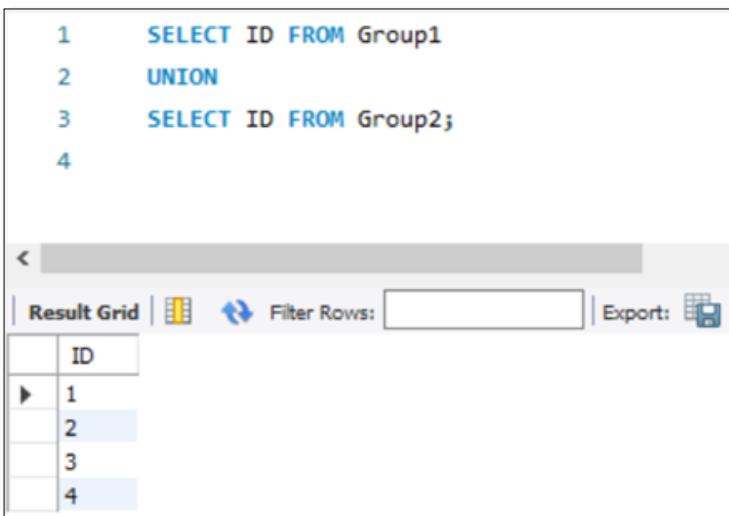
- Par défaut, l'opérateur **UNION** supprime les lignes en double même si on ne spécifie pas l'opérateur DISTINCT.
- Soient les deux tables :

GROUP1
ID
1
2
3

GROUP2
ID
2
3
4

- La requête UNION :

```
1  SELECT ID FROM Group1
2  UNION
3  SELECT ID FROM Group2;
4
```



The screenshot shows the MySQL Workbench interface with a query editor containing the following SQL code:

```
1  SELECT ID FROM Group1
2  UNION
3  SELECT ID FROM Group2;
4
```

Below the code, the "Result Grid" pane displays the following data:

ID
1
2
3
4

The results show that the UNION operation has combined the data from both tables, resulting in four unique rows (1, 2, 3, 4) without any duplicates.

Opérateur UNION

- La requête UNION ALL :

```
1  SELECT ID FROM Group1
2  UNION ALL
3  SELECT ID FROM Group2;
```

Result Grid | Filter Rows: [] | Export: [] | Wrap Cell Content: []

ID
1
2
3
2
3
4

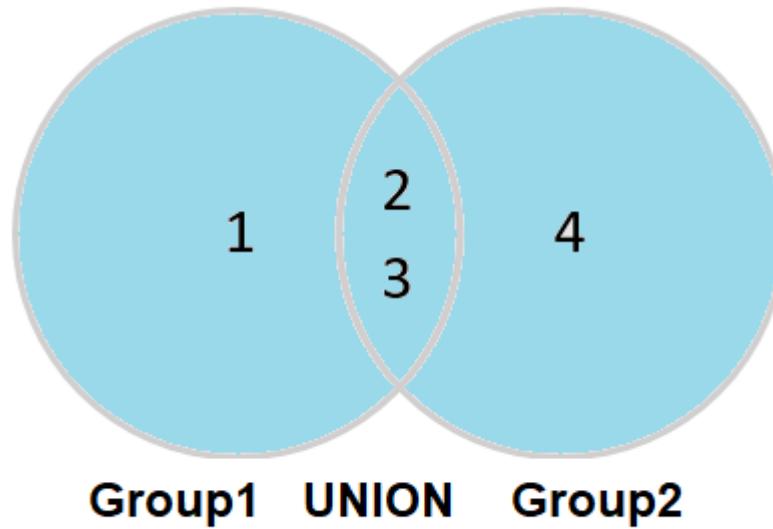
- Si on utilise UNION ALL explicitement, les lignes dupliquées, sont affichées dans le résultat. Du fait que UNION ALL ne gère pas les doublons, il s'exécute plus rapidement que UNION DISTINCT.

02 - Réaliser des requêtes SQL

Requêtes de l'union

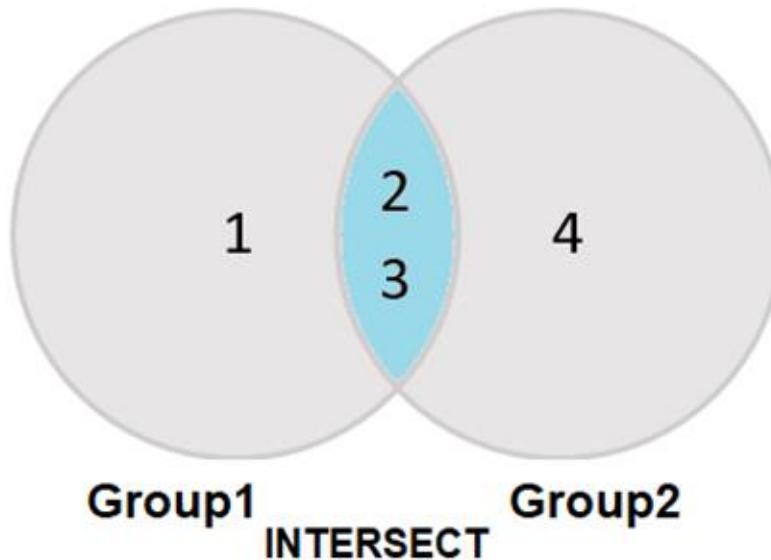
Opérateur UNION

- Le diagramme de Venn suivant illustre l'union de deux ensembles de résultats provenant des tables Group1 et Group2 :



Opérateur INTERSECT

- L'opérateur INTERSECT compare les ensembles de résultats de deux requêtes ou plus et renvoie les lignes distinctes générées par les deux requêtes.
- MySQL ne prend pas en charge l'opérateur INTERSECT. Cependant, On peut l'émuler en utilisant les jointures.
- Le schéma suivant illustre l'opérateur INTERSECT :



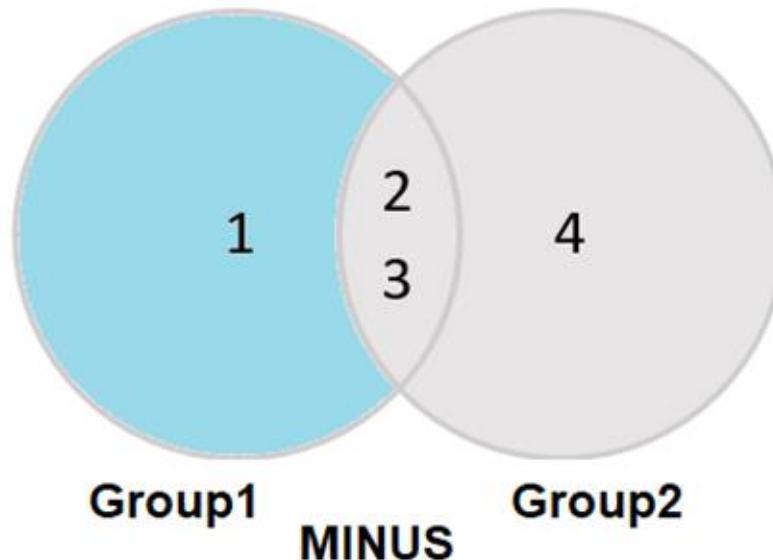
Opérateur INTERSECT

- Contrairement à l'opérateur UNION, l'opérateur INTERSECT renvoie l'intersection entre deux cercles.
- Donc la requête :
 - (SELECT ID FROM Group1)
 - INTERSECT
 - (SELECT ID FROM Group2)
- Retourne :

ID
2
3

Opérateur MINUS (EXCEPT)

- L'opérateur MINUS compare les résultats de deux requêtes et renvoie des lignes distinctes du jeu de résultats de la première requête qui n'apparaissent pas dans le jeu de résultats de la deuxième requête. MySQL ne prend pas en charge l'opérateur MINUS. Cependant, On peut l'émuler en utilisant les jointures.
- Le schéma suivant illustre l'opérateur MINUS :



- Donc la requête :
 - (SELECT ID FROM Group1)
 - MINUS
 - (SELECT ID FROM Group2)

- Retourne :

ID
1



CHAPITRE 2

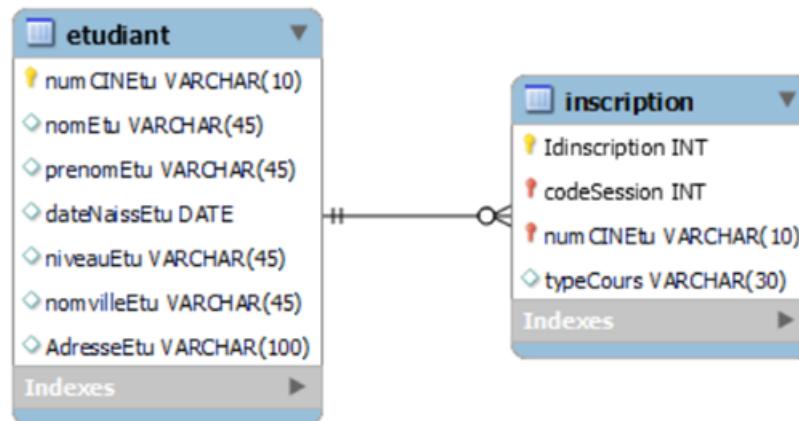
Réaliser des requêtes SQL

1. Requêtes LMD
2. Requêtes de sélection
3. Expression du SGBD
4. Fonctions d'agrégation du SGBD
5. Sous requêtes
6. Requêtes de l'union
7. **Jointures**

02 - Réaliser des requêtes SQL

Jointures

- Une base de données relationnelle se compose de plusieurs tables liées entre elles à l'aide de colonnes communes, appelées clés étrangères.
- Dans l'exemple Centre de Formation déjà présenté dans ce cours nous trouvons que les deux tables « Etudiant » et « Inscription » sont liées a l'aide de la colonne : numCINEtu



- Afin d'avoir plus d'information sur les étudiants ou les inscriptions, on a besoin de chercher dans les deux tables à la fois. D'où la nécessité des jointures.
- La **jointure** consiste à rechercher entre deux tables ayant un attribut commun (même type et même domaine de définition) tous les tuples (toutes les lignes) pour lesquels ces attributs ont la même valeur.
 - MySQL prend en charge les types de jointures suivants :
 - **INNER JOIN** (Jointure interne)
 - **LEFT JOIN** (Joint gauche)
 - **RIGHT JOIN** (Joindre à droite)
 - **CROSS JOIN** (Jointure croisée)
 - Pour joindre des tables, On utilise la clause de jointure dans l'instruction SELECT après la clause FROM.
 - Notez que MySQL ne prend pas en charge la jointure FULL OUTER JOIN.

INNER JOIN

- INNER JOIN joint deux tables en fonction d'une condition connue sous le nom de prédictat de jointure.
- Elle spécifie ainsi toutes les paires correspondantes de lignes renvoyées et ignore les lignes n'ayant pas de correspondance entre les deux tables. La clause INNER JOIN ne retient que les lignes des deux tables pour lesquelles l'expression exprimée au niveau de ON se vérifie.

Exemple :

	Num_Produit	description	cout	prix	Date_ajout
▶	P100	Laptop	5000.00	6000.00	2022-01-14
	P12	Non specifie	12.00	14.00	NULL
	P120	Laptop	6000.00	7000.00	2022-01-14
	P13	Accessoires	120.00	140.00	2022-01-01
	P14	Accessoires	20.00	40.00	2022-01-01

Produits

	IdSales	Num_Produit	Quantite	Client	Date_Vente
▶	1195	P120	2	Client 8	2021-10-20
	1196	P13	50	Client 8	2021-10-20
	1198	P100	6	Client 1	2021-11-30
	1201	P12	12	Client 1	2022-01-15

Sales

INNER JOIN

- Afin d'avoir pour chaque produit vendu, son prix, sa description et la quantité qui a été vendue, nous avons besoin de joindre les deux tables comme suit :

```
SELECT
    P.Num_Produit,
    P.Description,
    P.prix,
    S.quantite

FROM
    Produits P
INNER JOIN Sales S ON P.Num_Produit=S.Num_Produit;
```

INNER JOIN

- Résultat de la requête :

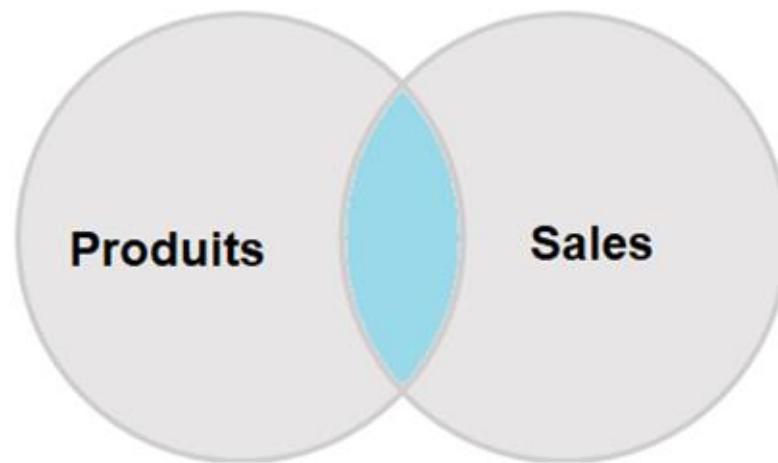
```
1  SELECT
2      P.Num_Produit,
3      P.Description,
4      P.prix,
5      S.Quantite
6  FROM
7      Produits P
8  INNER JOIN Sales S ON P.Num_Produit=S.Num_Produit;
9
```

Result Grid | Filter Rows: Export: Wrap Cell Content:

	Num_Produit	Description	prix	Quantite
▶	P120	Laptop	7000.00	2
	P13	Accessoires	140.00	50
	P100	Laptop	6000.00	6
	P12	Non specifie	14.00	12

INNER JOIN

- Dans cet exemple, INNER JOIN utilise les valeurs des colonnes de « Num_Produit » dans les tables « Produits » et « Sales » pour faire la correspondance.
- Le diagramme de Venn suivant illustre la jointure interne :



INNER JOIN

- Si la colonne de jointure a le même nom dans les deux tables objets de la jointure, on peut utiliser la syntaxe suivante :

```
SELECT
    P.Num_Produit,
    P.Description,
    P.prix,
    S.quantite

FROM
    Produits P
INNER JOIN Sales S USING (Num_Produit);
```

LEFT JOIN

- Lors de la jointure de deux tables à l'aide d'une jointure gauche, les concepts de tables gauche et droite sont introduits.
- La jointure gauche sélectionne les données à partir de la table de gauche. Pour chaque ligne de la table de gauche, la jointure de gauche est comparée à chaque ligne de la table de droite.
- Si les valeurs des deux lignes satisfont la condition de jointure, la clause de jointure gauche crée une nouvelle ligne dont les colonnes contiennent toutes les colonnes des lignes des deux tables et inclut cette ligne dans le jeu de résultats.
- Si les valeurs des deux lignes ne correspondent pas, la clause de jointure gauche crée toujours une nouvelle ligne dont les colonnes contiennent les colonnes de la ligne de la table de gauche et NULL pour les colonnes de la ligne de la table de droite.
- En d'autres termes, la jointure gauche sélectionne toutes les données de la table de gauche, qu'il existe ou non des lignes correspondantes dans la table de droite.

LEFT JOIN

Exemple :

```
SELECT
    P.Num_Produit,
    P.Description,
    P.prix,
    S.quantite

FROM
    Produits P
LEFT JOIN Sales S ON P.Num_Produit=S.Num_Produit;
```

- Ou Alors : LEFT JOIN Sales S USING(Num_Produit).

LEFT JOIN

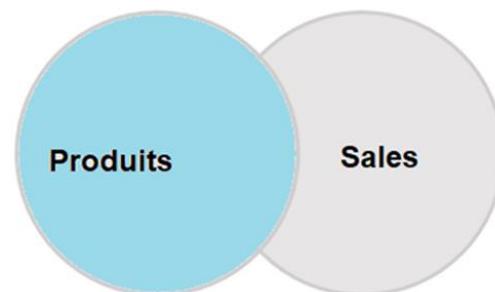
- Le résultat est le suivant :

```
1  SELECT
2      P.Num_Produit,
3      P.Description,
4      P.prix,
5      S.Quantite
6  FROM
7      Produits P
8  Left JOIN Sales S ON P.Num_Produit=S.Num_Produit;
9
```

Result Grid | Filter Rows: [] | Export: [] | Wrap Cell Content:

	Num_Produit	Description	prix	Quantite
▶	P120	Laptop	7000.00	2
▶	P13	Accessoires	140.00	50
▶	P100	Laptop	6000.00	6
▶	P12	Non specifie	14.00	12
▶	P14	Accessoires	40.00	NULL

- Le diagramme de Venn suivant illustre la jointure gauche :



RIGHT JOIN

- La clause de jointure droite est similaire à la clause de jointure gauche sauf que le traitement des tables gauche et droite est inversé. La jointure droite commence à sélectionner les données de la table de droite au lieu de la table de gauche : RIGHT JOIN sélectionne toutes les lignes de la table de droite et fait correspondre les lignes de la table de gauche. Si une ligne de la table de droite n'a pas correspondances dans la table de gauche, la colonne de la table de gauche aura NULL dans le jeu de résultats final.

02 - Réaliser des requêtes SQL

Jointures



RIGHT JOIN

Exemple :

```
SELECT
    S.Num_Produit,
    S.Quantite,
    P.prix
FROM
    Sales S
Right JOIN Produits P ON P.Num_Produit=S.Num_Produit;
```

- Ou Alors : RIGHT JOIN Produits P USING(Num_Produit).

RIGHT JOIN

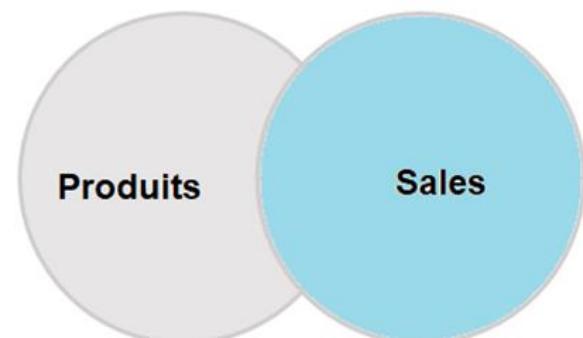
- Le résultat est le suivant :

```
1  SELECT
2      S.Num_Produit,
3      S.Quantite,
4      P.prix
5  FROM
6      Sales S
7  Right JOIN Produits P ON P.Num_Produit=S.Num_Produit;
8
```

Result Grid | Filter Rows: [] | Export: [] | Wrap Cell Content:

Num_Produit	Quantite	prix
P120	2	7000.00
P13	50	140.00
P100	6	6000.00
P12	12	14.00
NULL	NULL	40.00

- Le diagramme de Venn suivant illustre la jointure droite :



02 - Réaliser des requêtes SQL

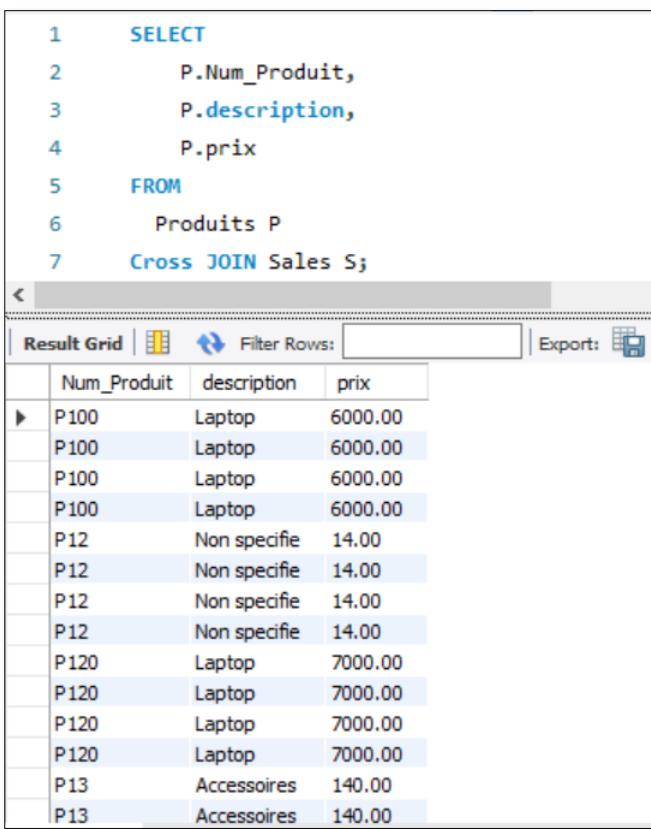
Jointures

CROSS JOIN

- Contrairement à autres types de jointures, la jointure croisée n'a pas de condition de jointure. Elle crée le produit cartésien des lignes des tables jointes. La jointure croisée combine chaque ligne de la première table avec chaque ligne de la table de droite pour créer le jeu de résultats.

Exemple :

```
1  SELECT
2      P.Num_Produit,
3      P.description,
4      P.prix
5  FROM
6      Produits P
7  Cross JOIN Sales S;
```



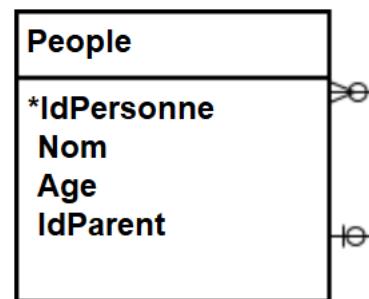
	Num_Produit	description	prix
▶	P100	Laptop	6000.00
	P12	Non specifie	14.00
	P12	Non specifie	14.00
	P12	Non specifie	14.00
	P12	Non specifie	14.00
	P120	Laptop	7000.00
	P13	Accessoires	140.00
	P13	Accessoires	140.00

SELF JOIN

- L'auto-jointure est souvent utilisée pour interroger des données hiérarchiques ou pour comparer une ligne avec d'autres lignes dans la même table.
- Pour effectuer une auto-jointure, on utilise des alias de la table pour ne pas répéter deux fois le même nom de table dans la même requête.
- **N.B :** Référencer une table deux fois ou plus dans une requête sans utiliser d'alias de table provoquera une erreur.

Exemple :

- La table « People » est définie ainsi :



	idPersonne	Nom	Age	idParent
▶	12	Mohammad	35	2
	2	Abdullah	54	1
	14	Ibrahim	1	12
	5	Ali	12	3
	3	Talib	56	1
	1	Mottalib	70	1

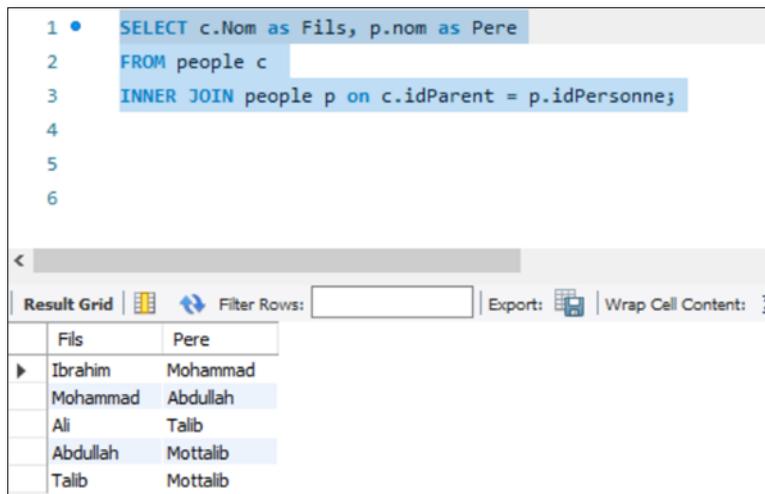
- La colonne IdParent définit le père de chaque personne, qui est aussi un élément de la table » People »

SELF JOIN

- Afin d'avoir la liste des personnes et leurs parents, il faut utiliser une auto-jointure.
- Au moyen de **INNER JOIN** :

```
SELECT c.Nom as Fils, p.nom as Pere
FROM people c
INNER JOIN people p on c.idParent = p.idPersonne;
```

- Le résultat nous donne uniquement les Personnes ayant un parent défini :



The screenshot shows a database interface with the following details:

- SQL Query:**

```
1 • SELECT c.Nom as Fils, p.nom as Pere
2   FROM people c
3 INNER JOIN people p on c.idParent = p.idPersonne;
4
5
6
```
- Result Grid:**

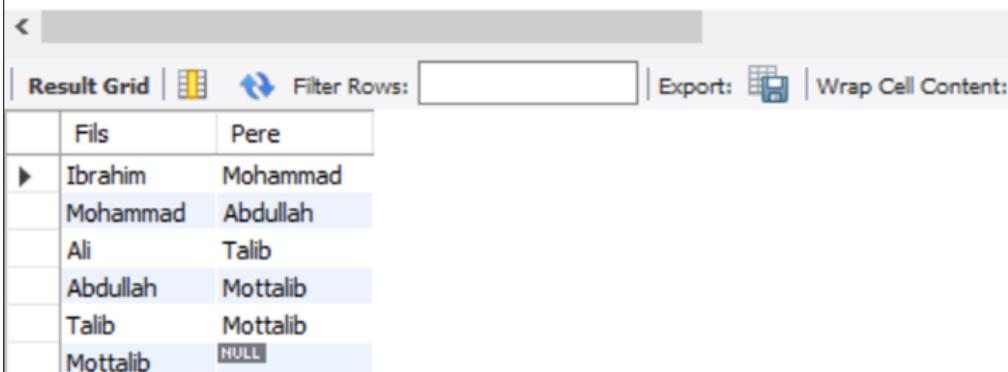
	Fils	Pere
▶	Ibrahim	Mohammad
	Mohammad	Abdullah
	Ali	Talib
	Abdullah	Mottalib
	Talib	Mottalib

SELF JOIN

- Afin d'avoir la liste des personnes et leurs parents, il faut utiliser une auto-jointure.
- Au moyen de **LEFT JOIN** :
- Le résultat comprend même les personnes qui n'ont pas un père défini.

Exemple :

```
1 •   SELECT c.Nom as Fils, p.nom as Pere
2     FROM people c
3     Left JOIN people p on c.idParent = p.idPersonne;
4
5
6
```



The screenshot shows a database query results grid. At the top, there is a code editor window containing the SQL query. Below it is a results grid with two columns: "Fils" and "Pere". The grid displays the following data:

Fils	Pere
Ibrahim	Mohammad
Mohammad	Abdullah
Ali	Talib
Abdullah	Mottalib
Talib	Mottalib
Mottalib	NULL



CHAPITRE 3

Administre une base de données

Ce que vous allez apprendre dans ce chapitre :

- Maîtriser les différentes fonctions liées à l'administration des BDD
- Vous initier avec les commandes de gestion des comptes et de priviléges de base





CHAPITRE 3

Administrer une base de données

1. **Backup/Restore**
2. Importation
3. Exportation
4. Commandes de création des comptes utilisateurs
5. Commandes de gestion des priviléges de base

03 - Administrer une base de données

Backup/Restore



Backup

L'outil `mysqldump` permet de sauvegarder une ou plusieurs bases de données. Il génère un fichier texte contenant les instructions SQL qui peuvent recréer les bases de données. `mysqldump` est situé dans le répertoire `root/bin` du répertoire d'installation de MySQL.

This PC > Windows (C:) > Program Files > MySQL > MySQL Server 8.0 > bin			
Name	Date modified	Type	Size
mysql_ssl_rsa_setup.exe	12/9/2019 1:40 PM	Application	6,147 KB
mysql_tzinfo_to_sql.exe	12/9/2019 1:40 PM	Application	6,063 KB
mysql_upgrade.exe	12/9/2019 1:40 PM	Application	6,753 KB
mysqladmin.exe	12/9/2019 1:40 PM	Application	6,674 KB
mysqlbinlog.exe	12/9/2019 1:40 PM	Application	6,946 KB
mysqlcheck.exe	12/9/2019 1:40 PM	Application	6,681 KB
mysqld.exe	12/9/2019 1:40 PM	Application	46,488 KB
mysqld_multi.pl	12/9/2019 9:23 PM	PL File	29 KB
mysqldump.exe	12/9/2019 1:40 PM	Application	6,740 KB
mysqldumpslow.pl	12/9/2019 9:23 PM	PL File	8 KB

03 - Administrer une base de données

Backup/Restore



Backup d'une ou plusieurs bases de données :

- La commande pour faire un backup avec mysqldump :

```
mysqldump --user=<username>
--password=<password>
--result-file=<Lien_Fichier_Backup>
--databases <Liste_des_databases>
```

- Dans cette syntaxe on doit définir :
 - Username et password: le nom et le mot de passe de l'utilisateur qui est connecté sur MySQL
 - Lien du fichier du backup
 - Le ou les noms des bases de données qu'on veut sauvegarder.
- Pour faire un backup de plusieurs bases de données à la fois on suit la syntaxe suivante : **--databases Nom_Base1, Nom_Base2, ...**
- Si on veut faire un backup de toutes les bases de données d'une instance MySql, on remplace l'option : **--databases <Liste_des_databases>**
- Par : **--all-databases**

03 - Administrer une base de données

Backup/Restore



Backup d'une ou plusieurs bases de données

Exemples : Backup de la base de données « dbtest »

```
Command Prompt
Microsoft Windows [Version 10.0.19042.1237]
(c) Microsoft Corporation. All rights reserved.

C:\Users\Khaoula>cd C:\Program Files\MySQL\MySQL Server 8.0\bin

C:\Program Files\MySQL\MySQL Server 8.0\bin>mysqldump --user=root --password=Mypassword4@ --result-file=c:\backup\dbtest
_backup.sql --databases dbtest
mysqldump: [Warning] Using a password on the command line interface can be insecure.
```

Backup d'une ou plusieurs bases de données

Exemples : Backup de la base de données « dbtest »

Le contenu du fichier dbtest_backup.sql après exécution :

```
dbtest_backup.sql x
18  --
19  -- Current Database: `dbtest`
20  --
21
22  CREATE DATABASE /*!32312 IF NOT EXISTS*/ `dbtest` /*!40100 DEFAULT CHARACTER SET utf8mb4
23
24  USE `dbtest`;
25
26  --
27  -- Table structure for table `group1`
28  --
29
30  DROP TABLE IF EXISTS `group1`;
31  /*!40101 SET @saved_cs_client      = @@character_set_client */;
32  /*!50503 SET character_set_client = utf8mb4 */;
33  CREATE TABLE `group1` (
34      `id` int NOT NULL,
35      PRIMARY KEY (`id`)
36  ) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4_0900_ai_ci;
37  /*!40101 SET character_set_client = @saved_cs_client */;
38
39  --
40  -- Dumping data for table `group1`
41  --
42
43  LOCK TABLES `group1` WRITE;
44  /*!40000 ALTER TABLE `group1` DISABLE KEYS */;
45  INSERT INTO `group1` VALUES (1),(2),(3);
46  /*!40000 ALTER TABLE `group1` ENABLE KEYS */;
47  UNLOCK TABLES;
```

Backup d'une ou plusieurs tables d'une base de données

- Afin de faire un backup de tables spécifiques d'une base de données, on exécute :
 - `mysqldump --user=<username> --password=<password> --result-file=<path_to_backup_file> <Nom_Base> <table1> <table2> <table3>..`

Exemple :

- Réaliser le backup des tables « Produits » et « Sales » de la base de données « dbtest »

```
mysqldump --user=root --password=Mypassword4@  
--result-file=c:\backup\backup_tables.sql dbtest produits sales
```

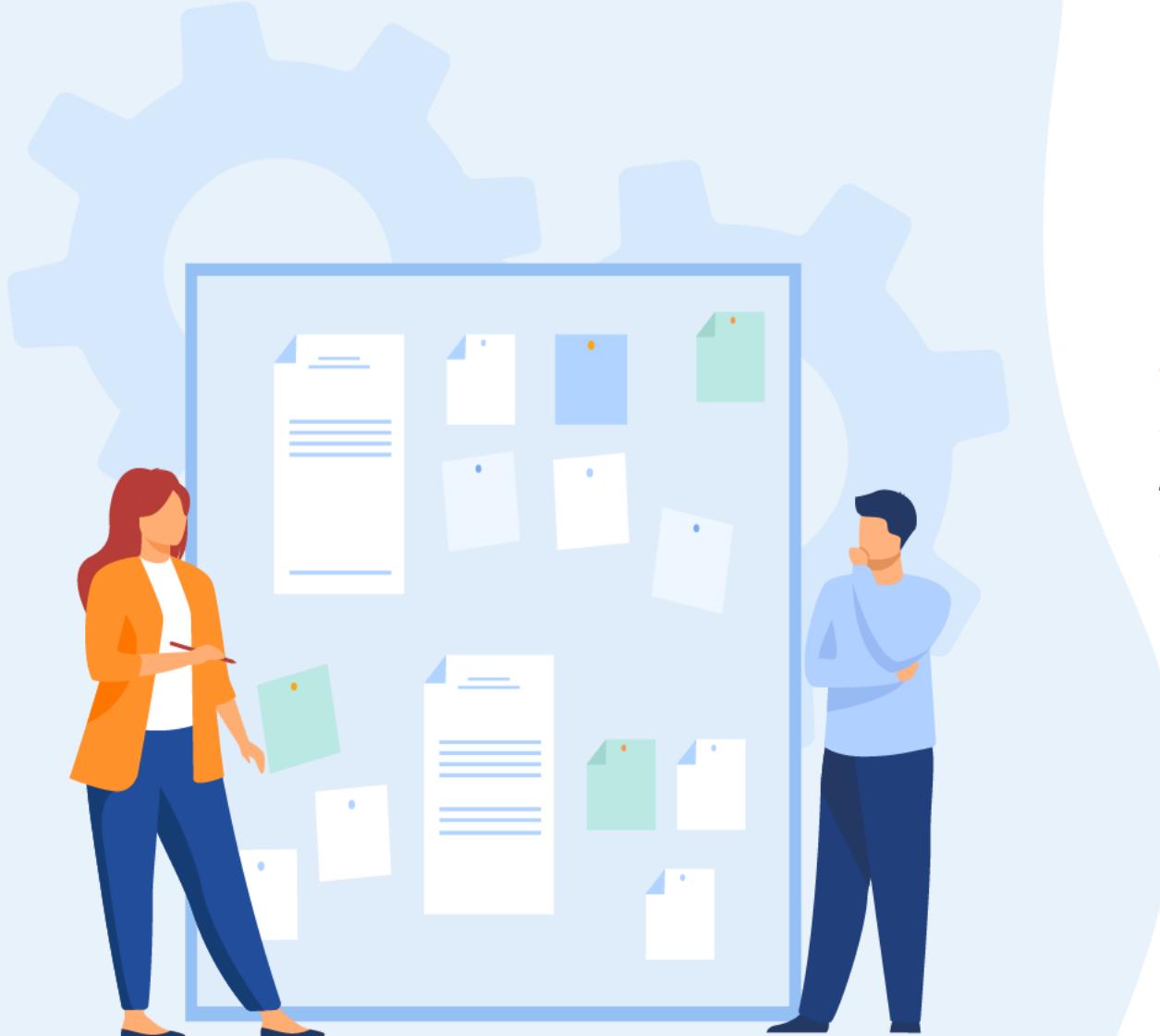
Backup de la structure ou les données d'une bases de données

- L'outil mysqldump permet aussi de sauvegarder juste la structure ou juste les données d'une base de données en utilisant respectivement les options : **--no-data** et **--no-create-info**
- Structure seulement :

```
mysqldump --user=<username>
           --password=<password>
           --result-file=<Lien_Fichier_Backup>
           --no-data
           --databases <Liste_des_databases>
```

- Données seulement :

```
mysqldump --user=<username>
           --password=<password>
           --result-file=<Lien_Fichier_Backup>
           --no-create-info
           --databases <Liste_des_databases>
```



CHAPITRE 3

Administre une base de données

1. Backup/Restore
2. **Importation**
3. Exportation
4. Commandes de création des comptes utilisateurs
5. Commandes de gestion des priviléges de base

03 - Administre une base de données

Importation



- Afin d'importer des données sous forme de fichier sql sur MySQL à partir de la ligne de commande, on peut utiliser la commande SOURCE comme dans le cas du Restore.

Exemple :

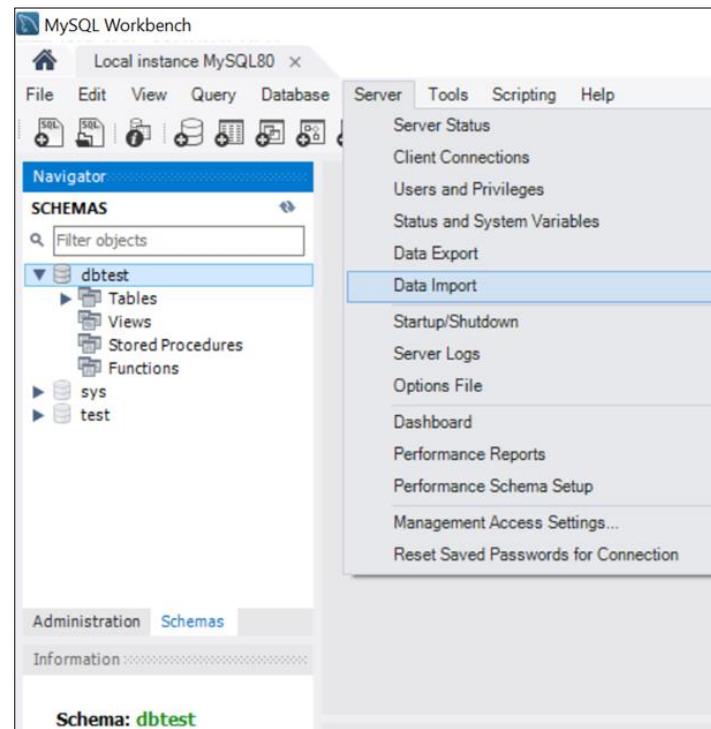
```
mysql>source c:\backup\fichier_backup.sql
```

- Il est recommandé d'utiliser la commande SOURCE pour restaurer une base de données car elle renvoie des informations très détaillées sur le processus, notamment des avertissements et des erreurs.

03 - Administrer une base de données

Importation

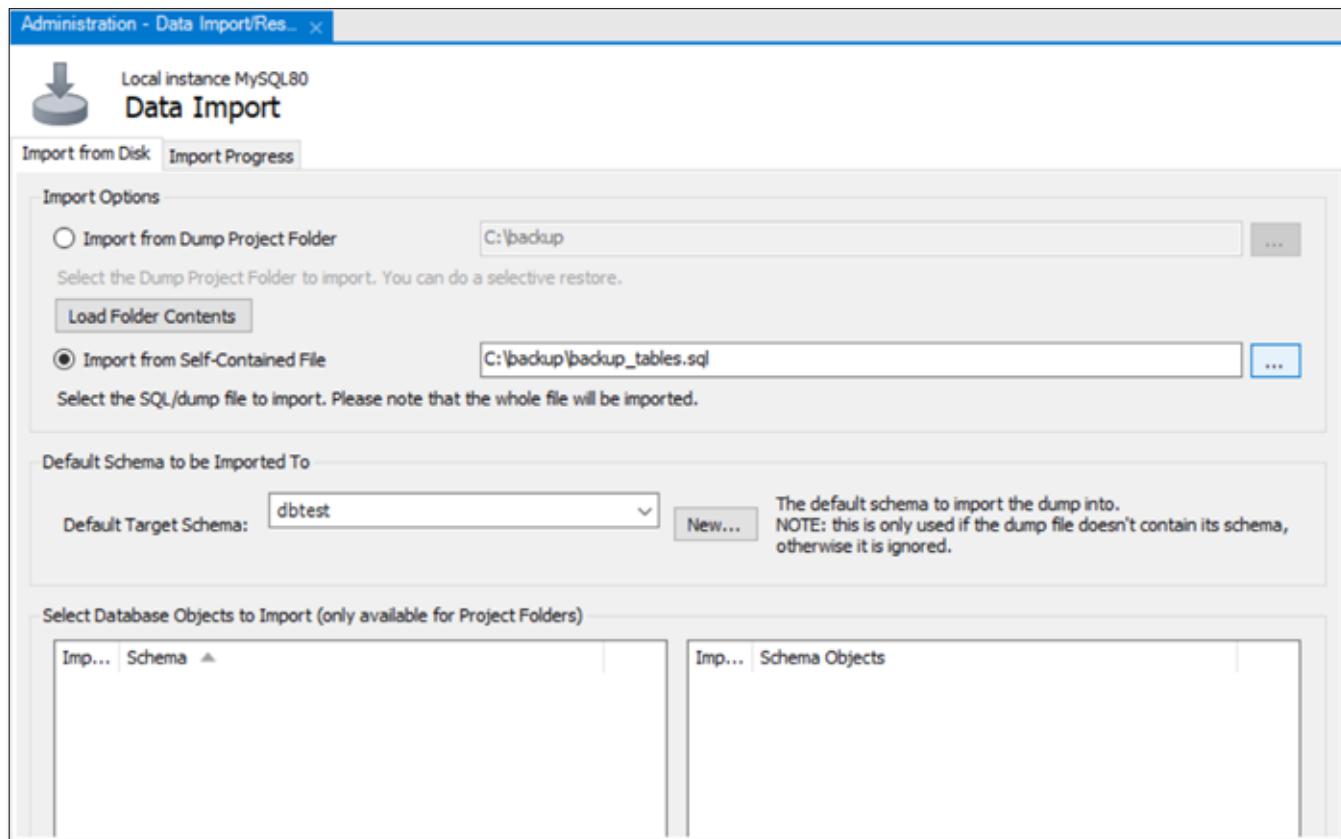
- L'utilitaire Import data dans Workbench permet aussi de réaliser cette tâche en suivant ces étapes :
 1. Ouvrez MySQL Workbench
 2. Dans la liste des MySQL Connexions, choisissez votre base de données
 3. Cliquer sur **Data Import** à partir de l'élément **Server** dans le menu de navigation



03 - Administrer une base de données

Importation

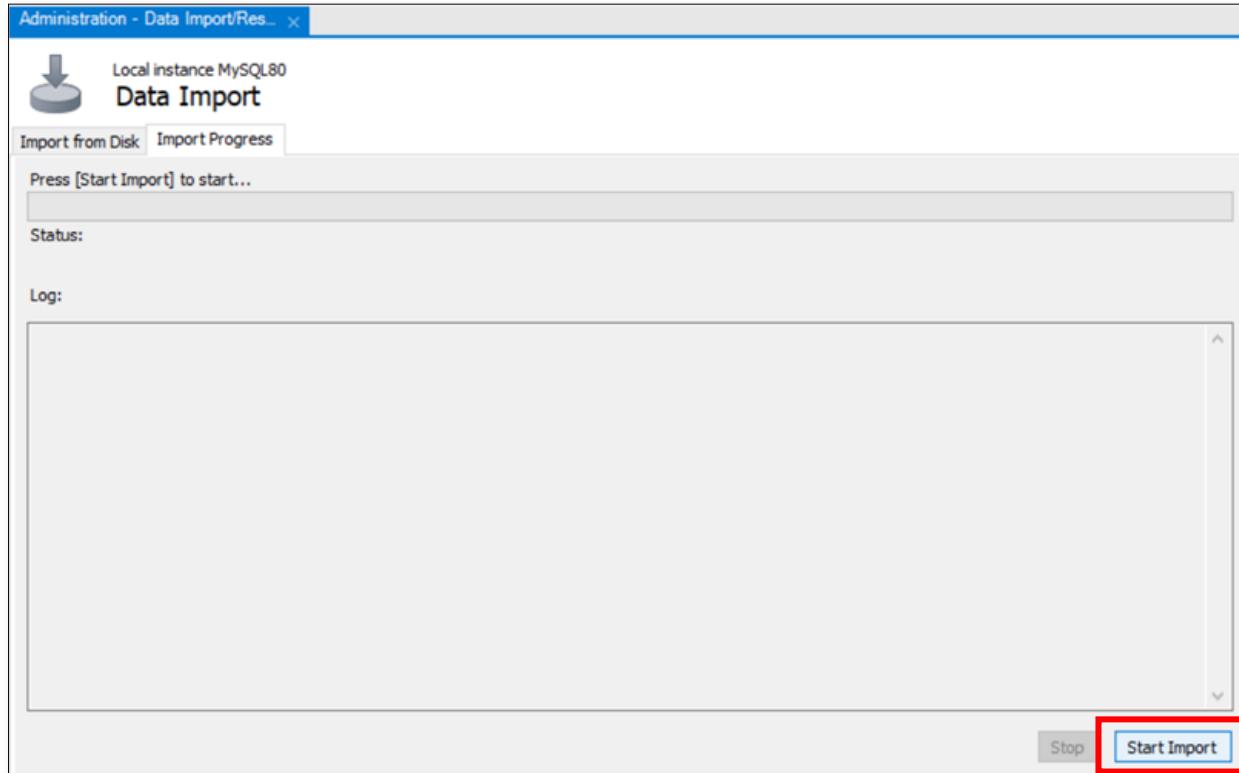
4. Dans le volet « Data Import from Disk », Section « Import Options », choisissez « Import from Self-Contained File », et sélectionner le fichier SQL qui contient les données à importer.



03 - Administrer une base de données

Importation

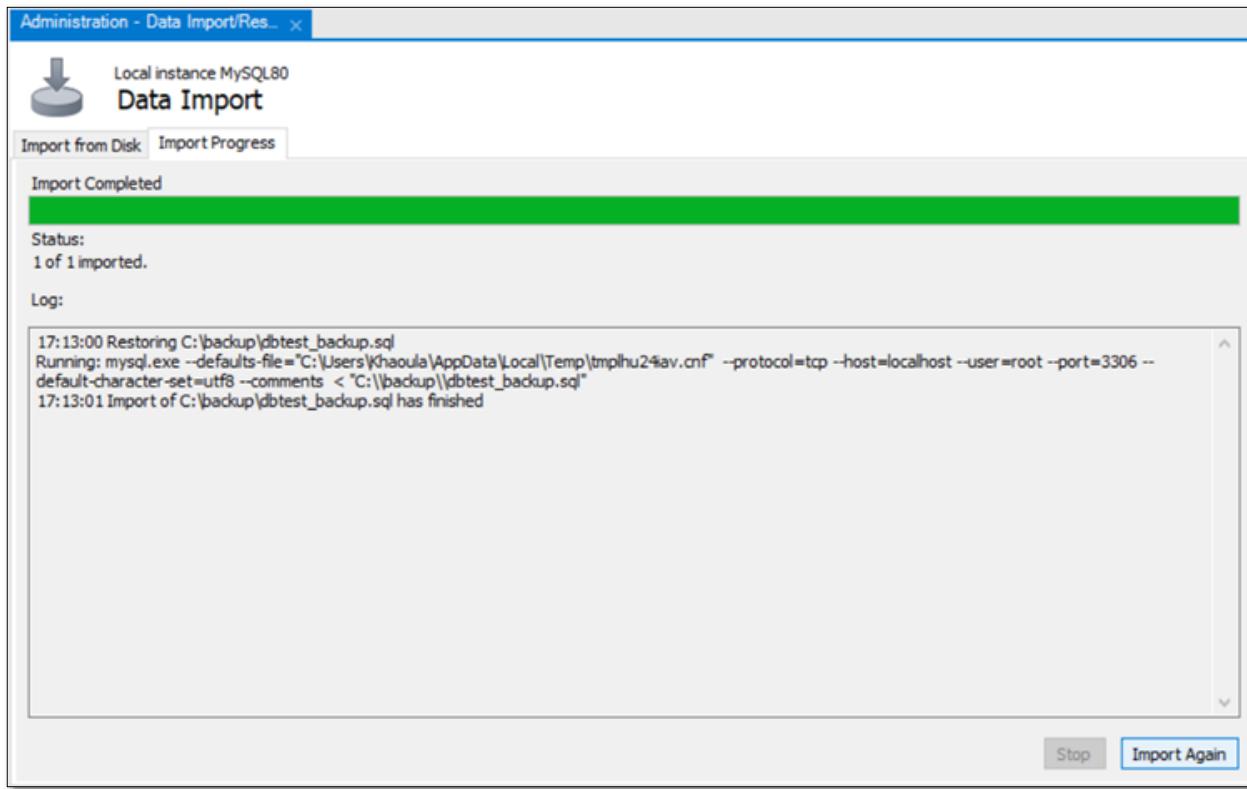
5. Choisissez le schéma cible par défaut (Default Target Schema) ou les données seront importées. Vous pouvez également créer une nouvelle base de données en choisissant New (Nouveau)
6. Passez sur le volet : Import Progress. Choisissez Start Import (Démarrer l'importation) pour lancer l'import



03 - Administrer une base de données

Importation

7. Votre importation peut prendre quelques minutes ou plus en fonction de la taille du fichier .SQL. Une fois l'importation terminée, vous devez voir un message semblable à ce qui suit :

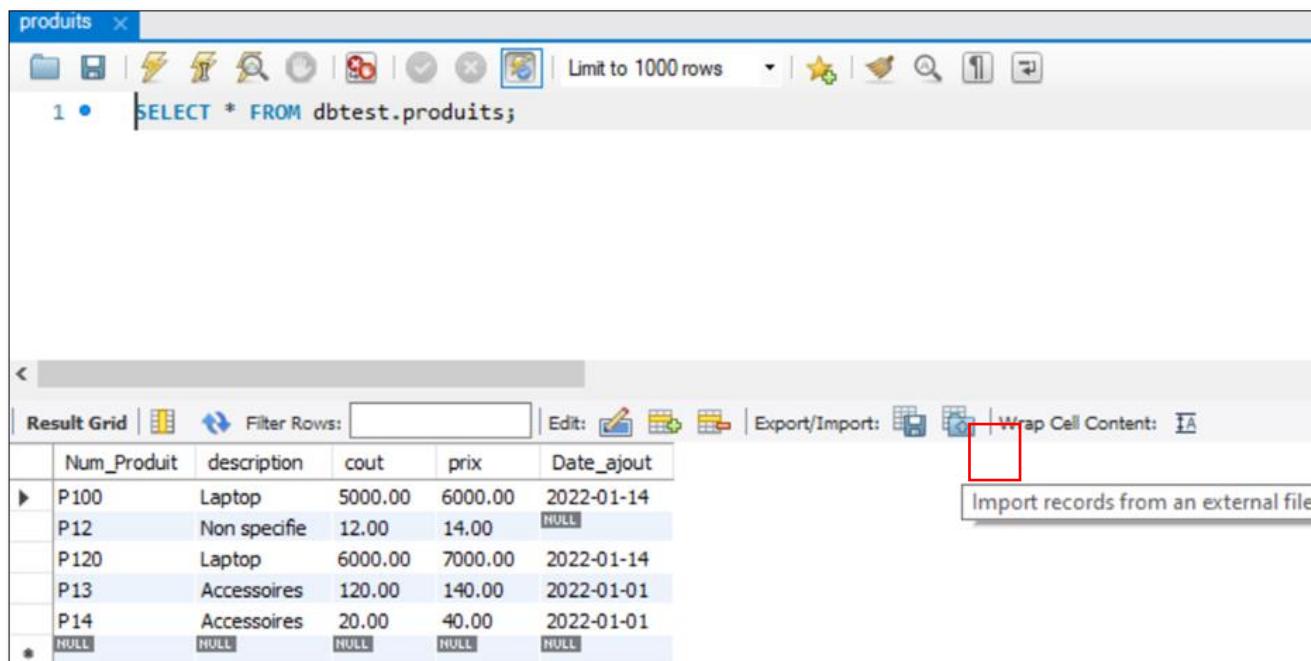


03 - Administrer une base de données

Importation

Importer des données vers une table :

- MySQL Workbench fournit un outil pour importer des données dans une table. Il permet de modifier les données avant de les charger.
- Voici les étapes à suivre pour importer des données dans une table :
 - Ouvrez la table dans laquelle vous voulez importer des données
 - Cliquez sur l'icône « Import records from an external file »



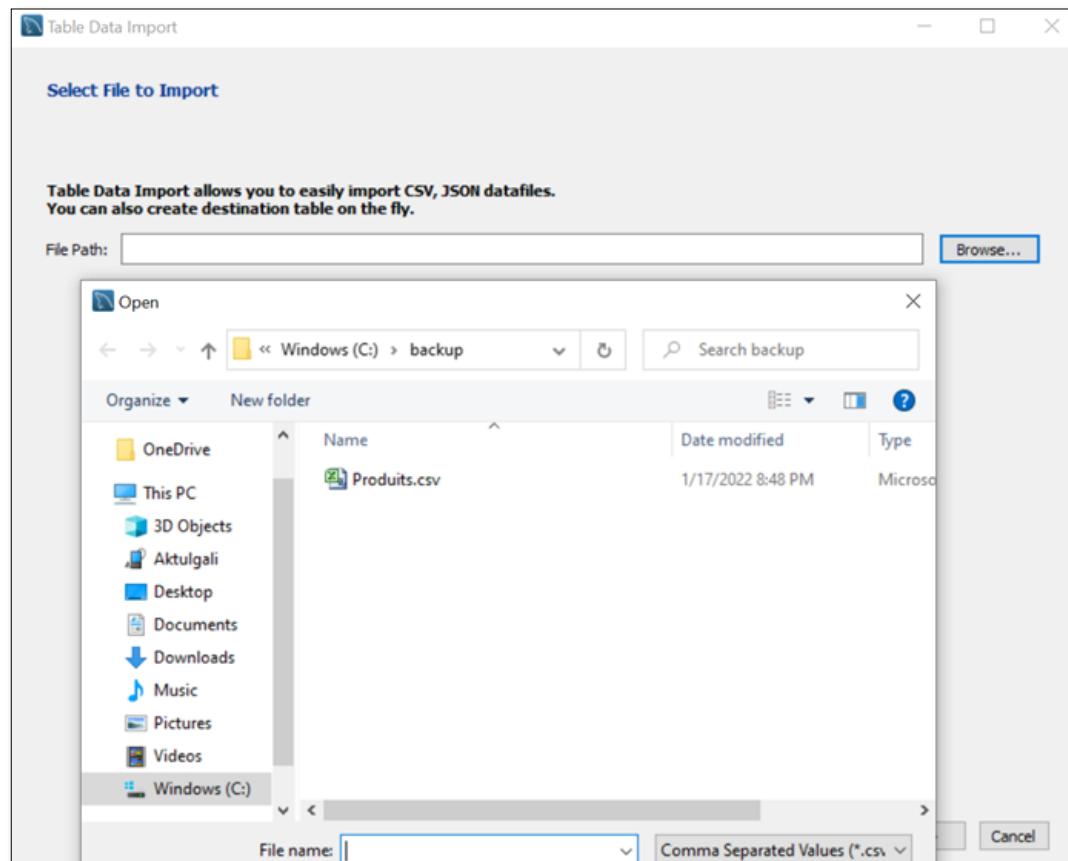
03 - Administrer une base de données

Importation



Importer des données vers une table :

3. Naviguer vers le fichier qui contient les données.

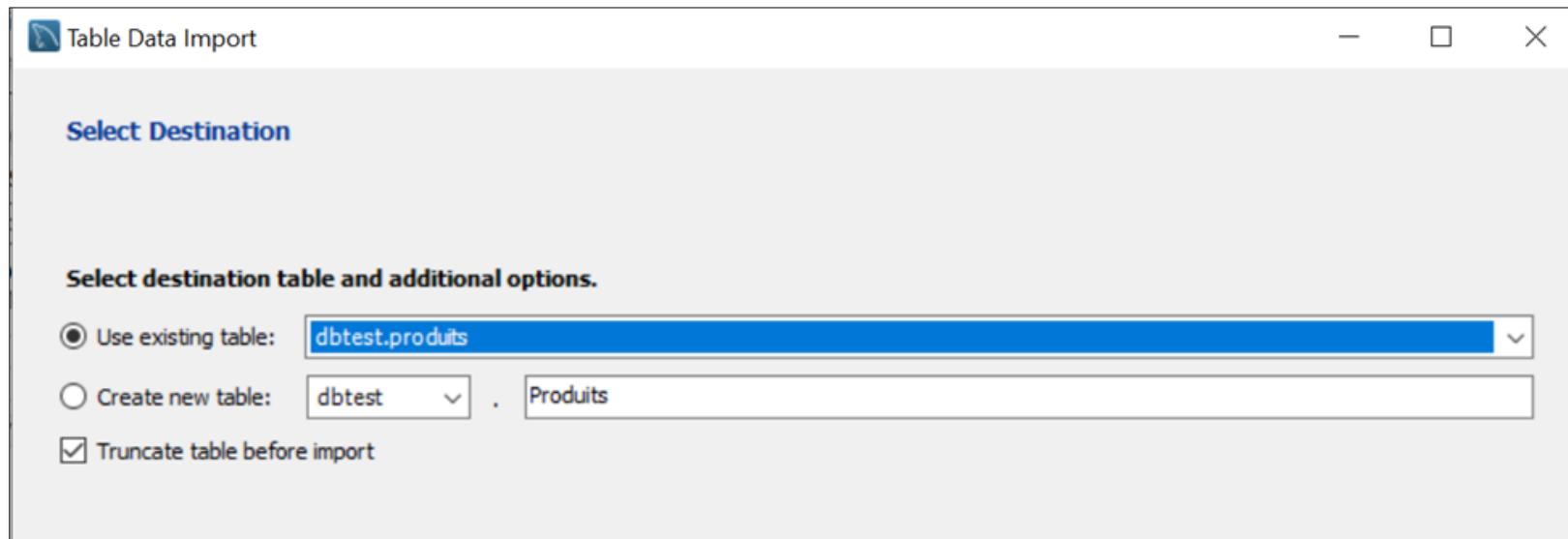


03 - Administrer une base de données

Importation

Importer des données vers une table :

4. Vous pouvez choisir d'importer les données vers une table qui existe déjà, ou en créer une nouvelle.

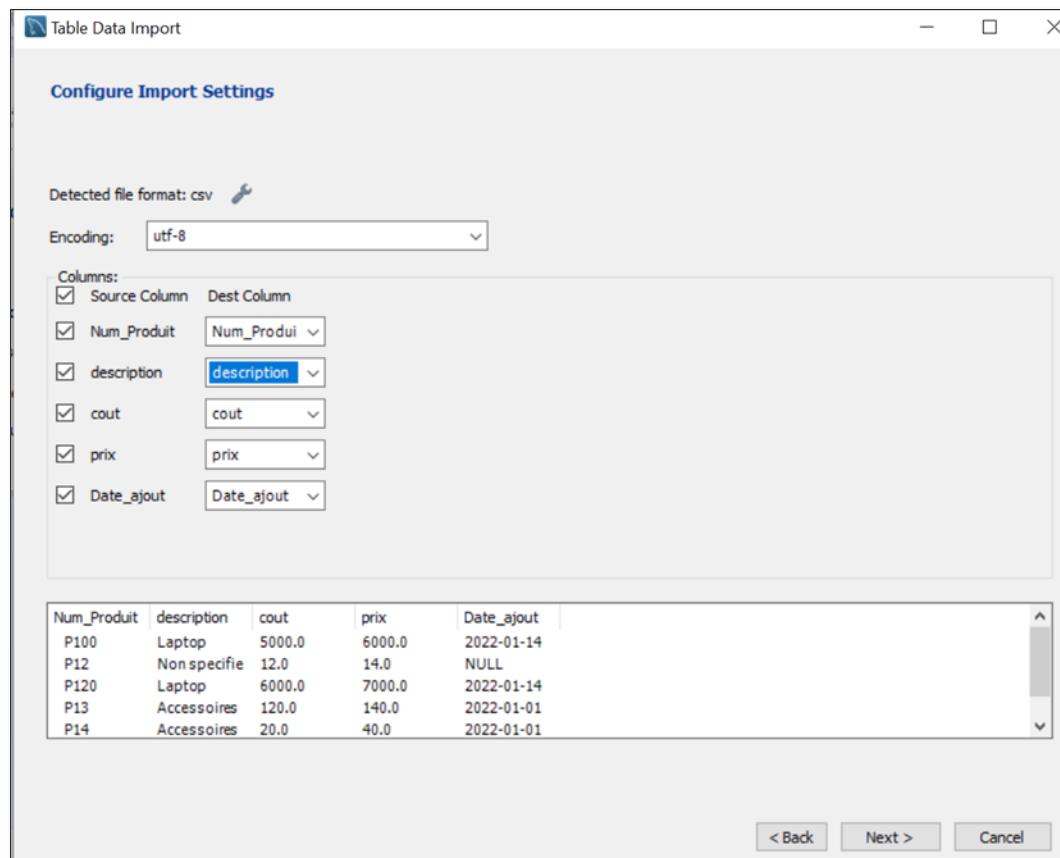


03 - Administrer une base de données

Importation

Importer des données vers une table :

5. Ensuite vous pouvez vérifier que le type du fichier choisis a été bien détecté et réaliser le mapping des colonnes du fichier avec celle de la table.

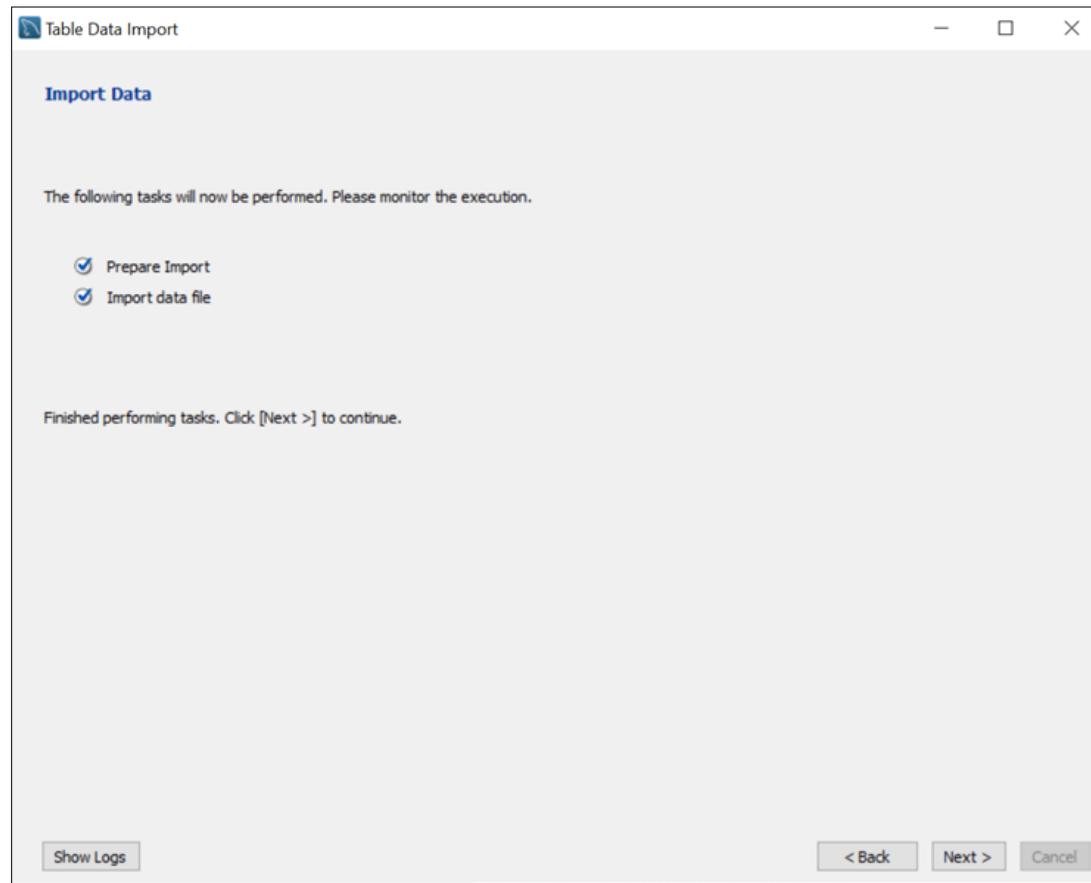


03 - Administrer une base de données

Importation

Importer des données vers une table :

6. Cliquer sur Next pour réaliser le Import.





CHAPITRE 3

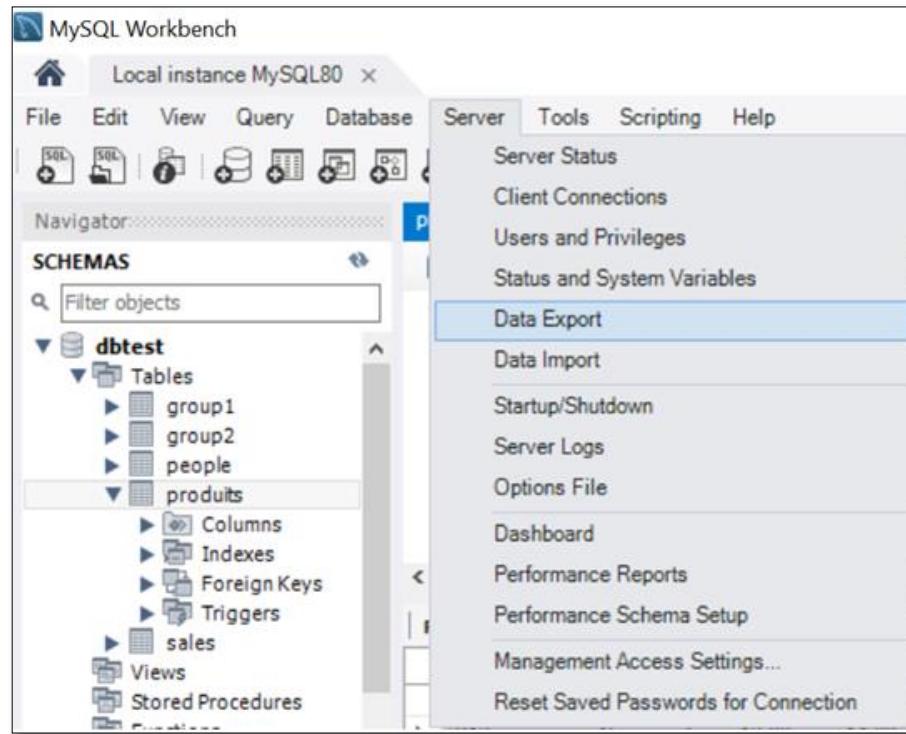
Administrer une base de données

1. Backup/Restore
2. Importation
- 3. Exportation**
4. Commandes de création des comptes utilisateurs
5. Commandes de gestion des priviléges de base

03 - Administrer une base de données

Exportation

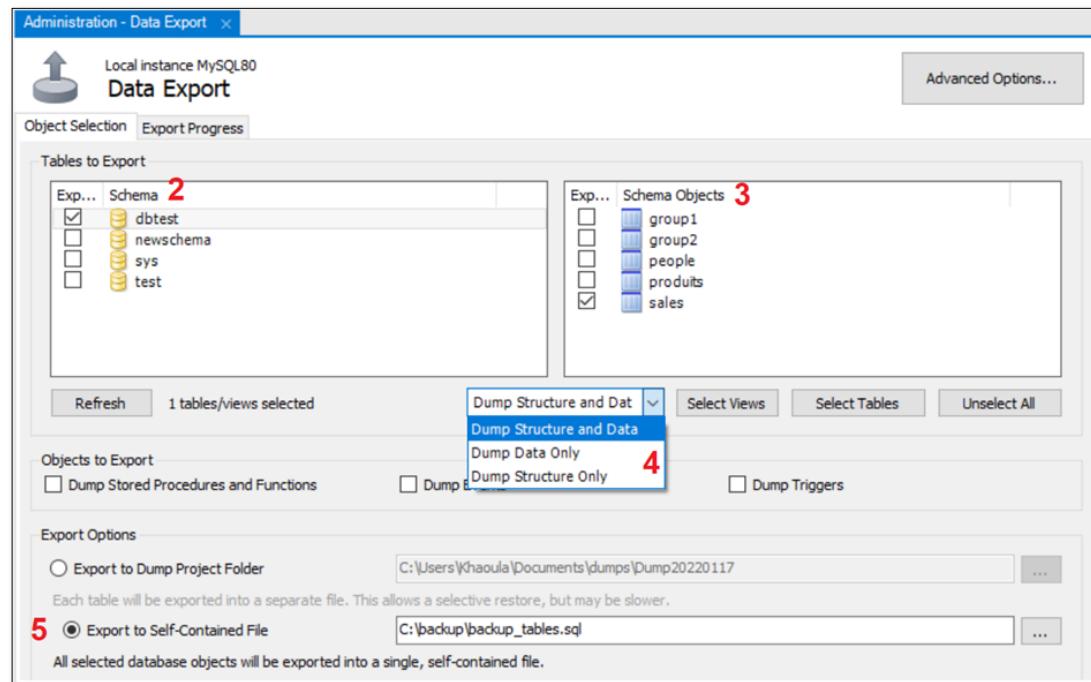
- L'utilitaire Export data dans Workbench permet d'exporter les données d'une base de données en suivant ces étapes :
1. Ouvrez MySQL Workbench. Dans la liste des MySQL Connexions, choisissez votre base de données, puis naviguer dans le menu Server et cliquer sur « Export Data »



03 - Administrer une base de données

Exportation

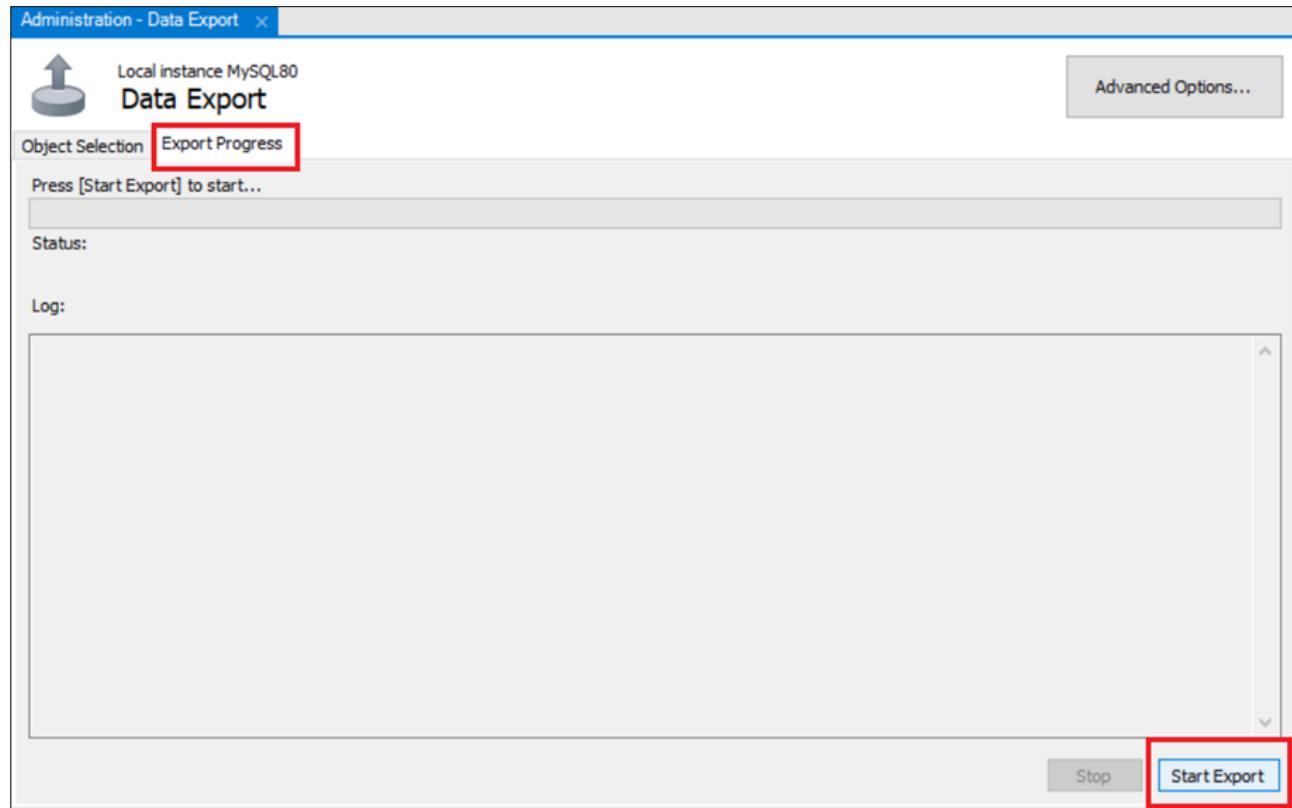
- Sur le volet « Objet Sélection »
 2. Choisissez le Schéma qui contient les données à exporter
 3. Choisissez les objets à exporter (Tables, Vues..)
 4. La liste déroulante permet de préciser s'il s'agit d'exporter la structure des objets sélectionnés, les données qu'ils contiennent ou les deux. On peut aussi choisir d'exporter d'autres objets comme les procédures stockées, les Triggers..
 5. Sélectionnez le fichier cible qui va contenir le script des données exportées



03 - Administrer une base de données

Exportation

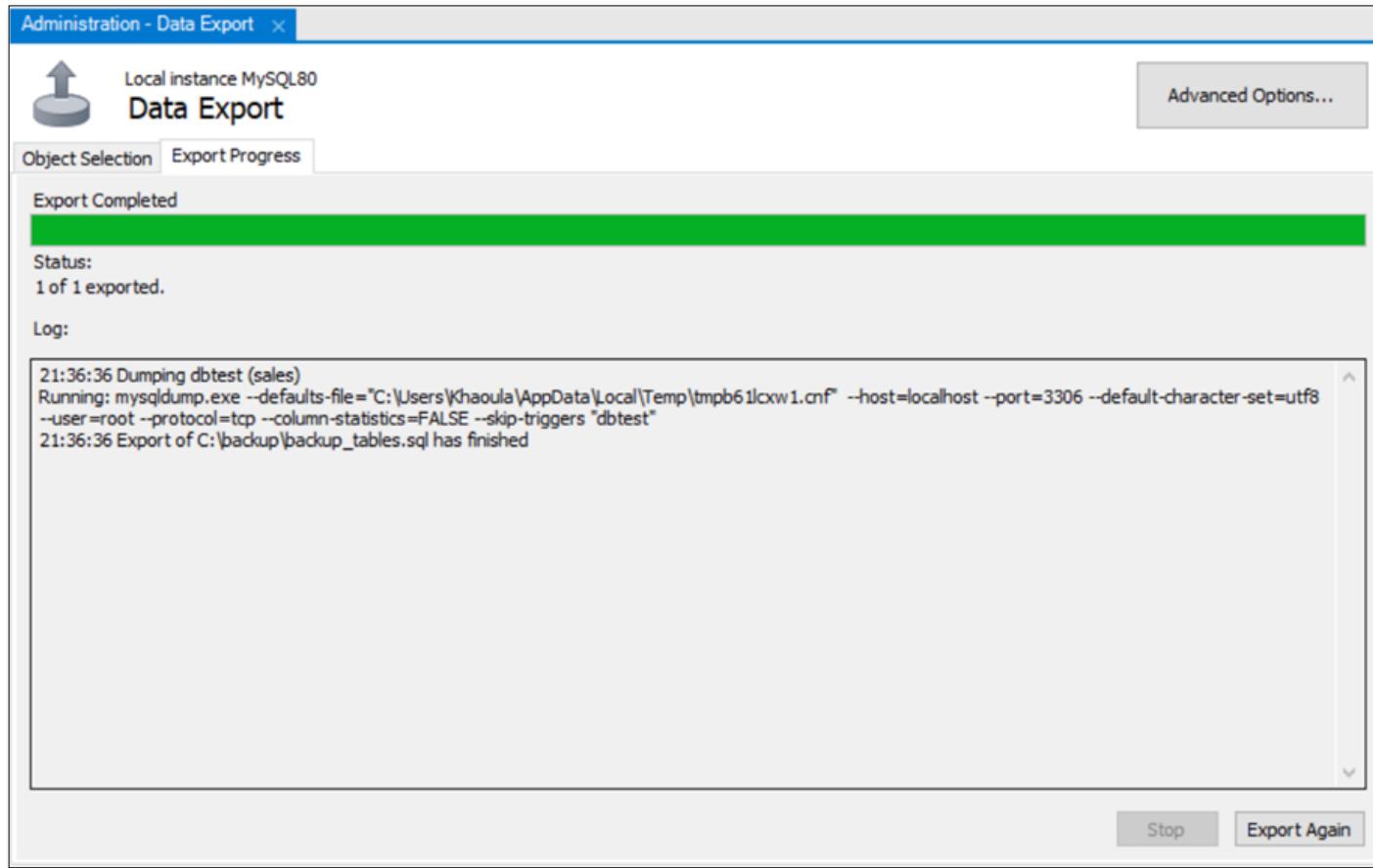
- Sur le volet « Export Progress »
 - Cliquez sur « Start Export » pour commencer le process



03 - Administrer une base de données

Exportation

7. Le système confirme l'export :

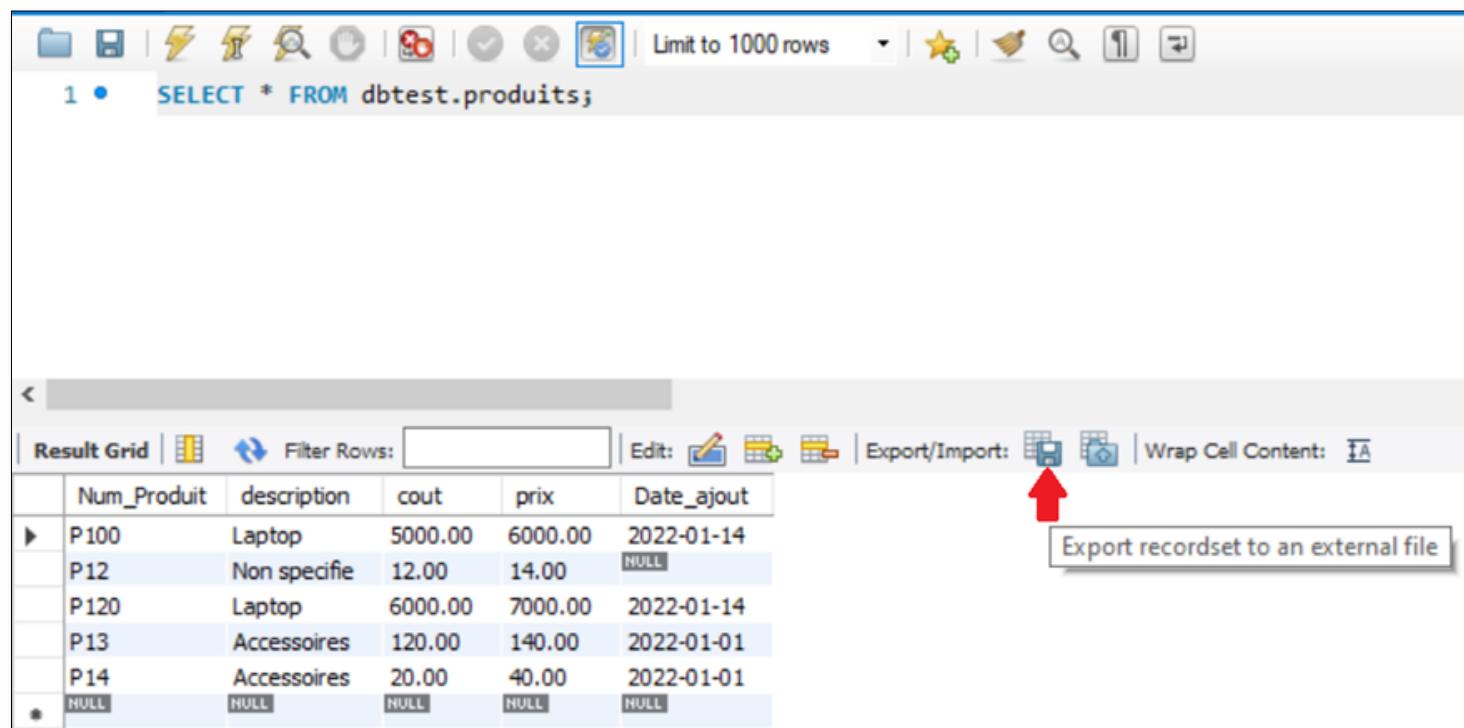


03 - Administrer une base de données

Exportation

Exporter les données d'une table :

- MySQL Workbench fournit un outil pour exporter les données d'une table. Voici les étapes à suivre :
 - Ouvrez la table de laquelle vous voulez exporter des données
 - Cliquez sur l'icône « Export recordset to an external file »

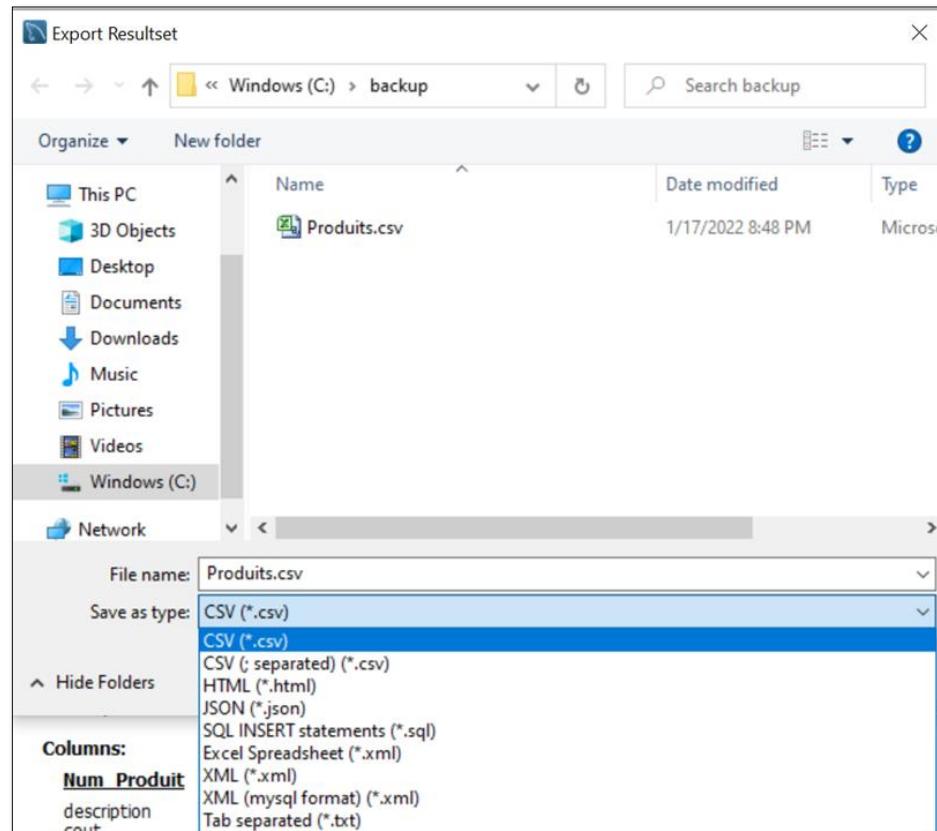


03 - Administrer une base de données

Exportation

Exporter les données d'une table :

3. Choisissez le type et l'emplacement du fichier data à créer, et cliquer sur Ok





CHAPITRE 3

Administrer une base de données

1. Backup/Restore
2. Importation
3. Exportation
4. **Commandes de création des comptes utilisateurs**
5. Commandes de gestion des priviléges de base

03 - Administrer une base de données

Commandes de création des comptes utilisateurs



- On utilise la commande **CREATE USER** pour créer de nouveaux utilisateurs dans le serveur de base de données MySql.
- Voici la syntaxe de base de **CREATE USER** :

```
CREATE USER [IF NOT EXISTS] nom_compte  
IDENTIFIED BY 'mot_de_passe';
```

- Dans cette expression il faut spécifier :
 - **nom_compte** : Il s'agit du nom du compte à créer et se compose en général de deux parties sous cette forme :
`nom_utilisateur@nom_host`
 - « **nom_utilisateur** » est le nom de l'utilisateur. Et « **nom_host** » est le nom de l'hôte à partir duquel l'utilisateur se connecte au serveur MySQL. La partie nom d'hôte du nom de compte est optionnelle. Si elle est omise, l'utilisateur peut se connecter depuis n'importe quel hôte.
 - **Mot_de_passe** : Le mot de passe relatif au nouveau compte.
- **CREATE USER** crée un nouvel utilisateur sans aucun privilège.

03 - Administrer une base de données

Commandes de création des comptes utilisateurs



Exemple :

- Sur la ligne de commande MySQL, on liste les utilisateurs existants : `mysql> select user from mysql.user`

• Résultat

```
mysql> select user from mysql.user;
+-----+
| user |
+-----+
| mysql.infoschema |
| mysql.session |
| mysql.sys |
| root |
+-----+
4 rows in set (0.20 sec)
```

- `mysql> create user Ahmad@localhost identified by 'Monmot2p@ss'`
- On peut vérifier sur la table mysql.user la création du nouveau

```
mysql> select user from mysql.user;
+-----+
| user |
+-----+
| Ahmad |
| mysql.infoschema |
| mysql.session |
| mysql.sys |
| root |
+-----+
5 rows in set (0.00 sec)
```

03 - Administrer une base de données

Commandes de création des comptes utilisateurs



Exemple :

- Afin de tester le login, on ouvre une autre session MySQL avec le compte « Ahmad » , en utilisant la commande suivante : **Mysql – u Ahmad –p**
- Puis on saisit le mot de passe : **Monmot2p@ss**

```
C:\Program Files\MySQL\MySQL Server 8.0\bin>mysql -u Ahmad -p
Enter password: *****
```

- On peut vérifier les bases de données auxquelles Ahmad peut accéder :

```
mysql> show databases;
+-----+
| Database      |
+-----+
| information_schema |
+-----+
1 row in set (0.01 sec)
```



CHAPITRE 3

Administrer une base de données

1. Backup/Restore
2. Importation
3. Exportation
4. Commandes de création des comptes utilisateurs
5. **Commandes de gestion des privilèges de base**

03 - Administrer une base de données

Commandes de gestion des privilèges de base



Attribution des privilèges (GRANT)

- La commande CREATE USER crée un ou plusieurs comptes d'utilisateurs sans privilèges. Pour qu'un utilisateur puisse accéder aux objets de base de données, il faut d'abord lui accorder des privilèges . Ceci se fait à l'aide de la commande GRANT.
- Voici la syntaxe générique de GRANT :

```
GRANT privilege [,privilege],...  
ON privilege_level  
TO nom_utilisateur;
```

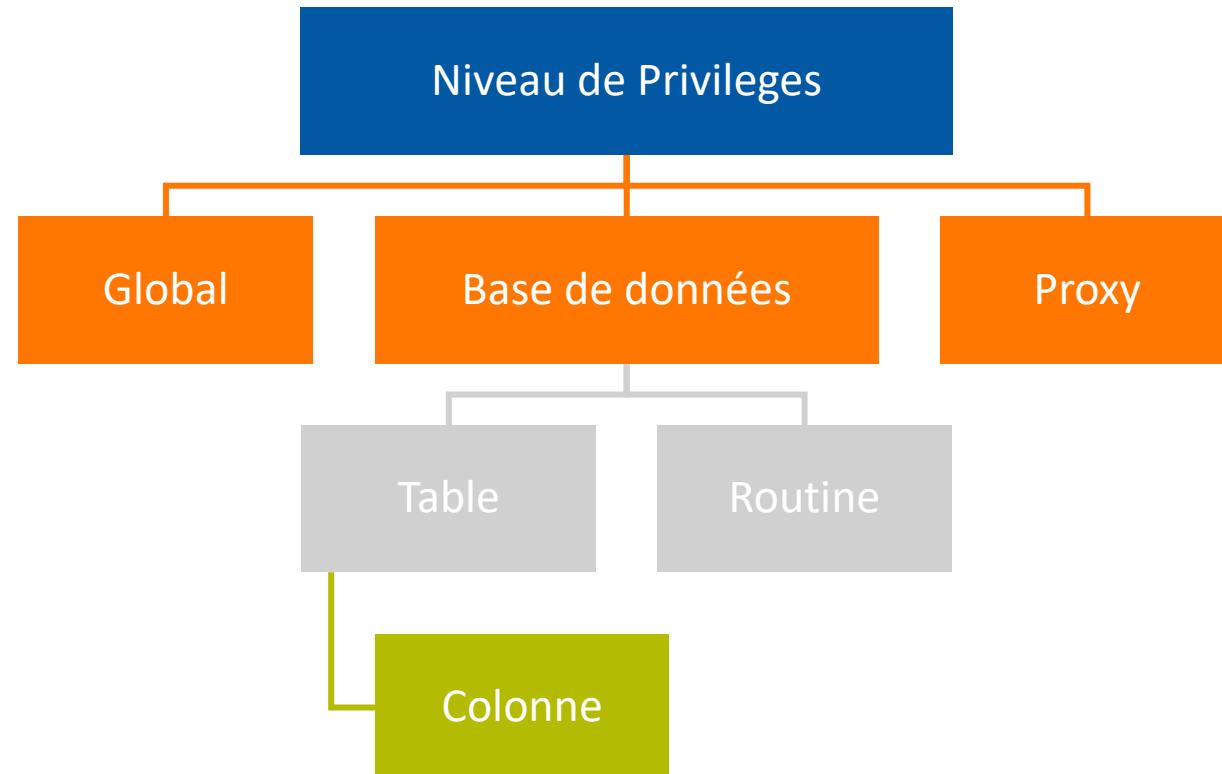
- Pour donner des privilèges au compte « nom_utilisateur », il faut spécifier le ou les privilèges à donner (SELECT, DELETE, UPDATE, ALL...) et sur quel niveau les appliquer.

03 - Administre une base de données

Commandes de gestion des privilèges de base

Attribution des privilèges (GRANT) : Les niveaux de privilèges

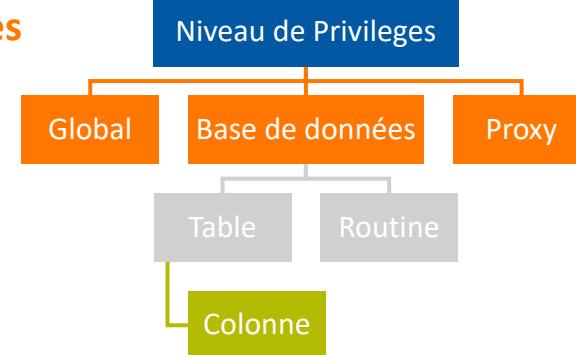
Il existe différents niveaux de privilèges dans MySql :



03 - Administre une base de données

Commandes de gestion des privilèges de base

Attribution des privilèges (GRANT) : Les niveaux de privilèges



1. Niveau global :

- Les privilèges globaux s'appliquent à toutes les bases de données d'un serveur MySQL. Pour attribuer des privilèges globaux, vous utilisez la syntaxe `*.*`.
- Exemple :

```
GRANT UPDATE
```

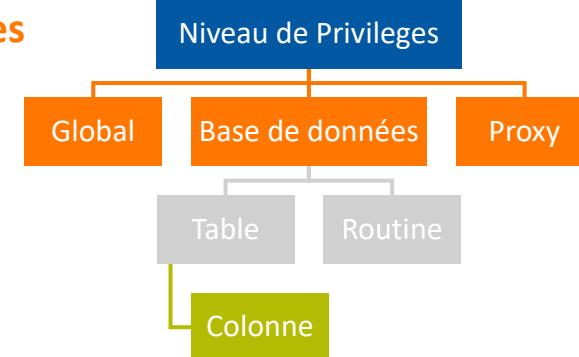
```
ON *.*
```

```
TO Ahmad@localhost;
```

03 - Administrer une base de données

Commandes de gestion des privilèges de base

Attribution des privilèges (GRANT) : Les niveaux de privilèges

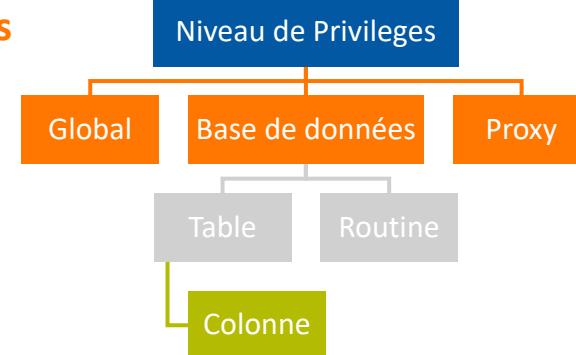


2. Niveau Base de données :

- Ces privilèges s'appliquent à tous les objets d'une base de données. Pour attribuer des privilèges au niveau de la base de données, on utilise la syntaxe : ON nom_base_de_données.*
- **Exemple :**

```
GRANT INSERT  
ON dbtest.*  
TO Ahmad@localhost;
```

Attribution des privilèges (GRANT): Les niveaux de privilèges



3. Niveau Table :

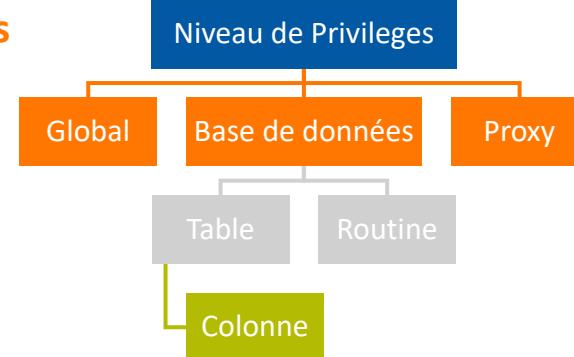
- Les privilèges de table s'appliquent à toutes les colonnes d'une table. Pour attribuer des privilèges au niveau de la table, on utilise la syntaxe ON nom_base_de_données.nom_table.
- **Exemple :**

```
GRANT INSERT,DELETE  
ON dbtest.Produits  
TO Ahmad@localhost;
```

03 - Administre une base de données

Commandes de gestion des privilèges de base

Attribution des privilèges (GRANT): Les niveaux de privilèges



4. Niveau Colonne :

- Les privilèges de colonne s'appliquent à des colonnes uniques dans une table. Vous devez spécifier la ou les colonnes pour chaque privilège.

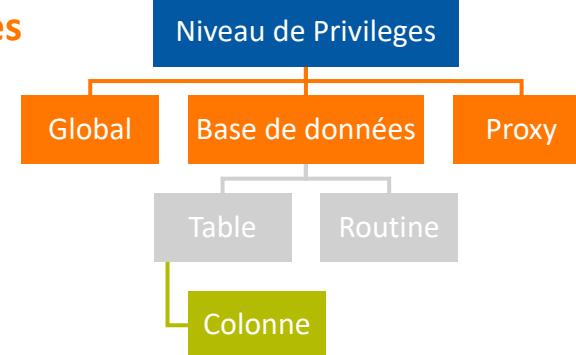
Exemple :

```
GRANT  
  
SELECT (Num_Produit, Description, Date_ajout),  
UPDATE (Prix)  
ON dbtest.Produits  
TO Ahmad@localhost;
```

03 - Administrer une base de données

Commandes de gestion des privilèges de base

Attribution des privilèges (GRANT) : Les niveaux de privilèges

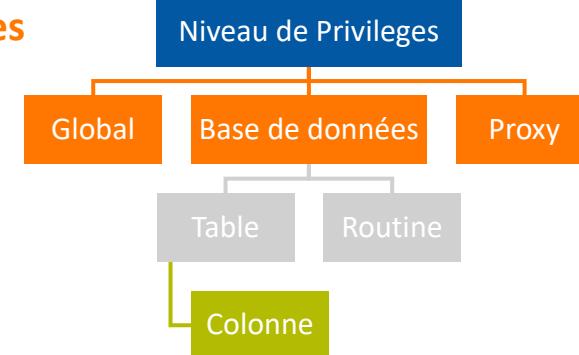


5. Niveau Routine :

- Les privilèges de routine s'appliquent aux procédures et fonctions stockées.
- **Exemple :**

```
GRANT EXECUTE  
ON PROCEDURE CalculPrix  
TO Ahmad@localhost;
```

Attribution des privilèges (GRANT) : Les niveaux de privilèges



6. Niveau Proxy :

- Les privilèges d'utilisateur proxy permettent à un utilisateur externe d'être un proxy pour un autre, c'est-à-dire, d'avoir les privilèges du deuxième utilisateur. En d'autres termes, l'utilisateur externe est un "utilisateur proxy" (un utilisateur qui peut usurper l'identité ou devenir un autre utilisateur) et le deuxième utilisateur est un "utilisateur mandaté" (un utilisateur dont l'identité peut être reprise par un utilisateur proxy).
- **Exemple :**

```
GRANT PROXY  
ON root  
TO Ahmad@localhost;
```

03 - Administre une base de données

Commandes de gestion des privilèges de base



Attribution des privilèges (GRANT): Les niveaux de privilèges

Le tableau suivant illustre les privilèges autorisés les plus utilisés pour les instructions GRANT et REVOKE La liste exhaustives des privilèges MySql est consultable sur le lien suivant : <https://dev.mysql.com/doc/refman/8.0/en/privileges-provided.html>

Privilège	Description	Niveau					
		Global	B. Données	Table	Colonne	Routine	Proxy
ALL PRIVILEGES	Accorde tous les privilèges au niveau spécifié sauf GRANT OPTION						
ALTER	Autorise l'utilisation de ALTER TABLE	X	X	X			
ALTER ROUTINE	Autorise Alter et Drop des procédures et fonctions stockées	X	X			X	
CREATE	Autorise la création des bases de données et tables	X	X	X			
CREATE ROUTINE	Autorise la création des procédures et fonctions stockées	X	X				
CREATE TEMPORARY TABLES	Autorise la création des tables temporaires: CREATE TEMPORARY TABLE	X	X				
CREATE USER	Autorise l'utilisation de: CREATE USER, DROP USER, RENAME USER et REVOKE ALL PRIVILEGES						
CREATE VIEW	Autorise la création et modification des vues.	X	X	X			
DELETE	Autorise l'utilisation de DELETE.	X	X	X			
DROP	Autorise la suppression des bases de données et des tables.	X	X	X			
EXECUTE	Autorise l'exécution des routines.	X	X	X			
GRANT OPTION	Autorise l'utilisateur à accorder ou révoquer des privilèges d'autres comptes	X	X	X		X	X
INDEX	Autorise la création et suppression des index	X	X	X			
INSERT	Autorise l'utilisation de INSERT	X	X	X	X		
PROCESS	Autorise l'exécution de SHOW PROCESSLIST.	X					
PROXY	Autorise l'utilisation de PROXY						
REFERENCES	Autorise la création des clés étrangères	X	X	X	X		
SELECT	Autorise l'utilisation SELECT	X	X	X	X		
SHUTDOWN	Autorise l'utilisation de la commande mysqladmin shutdown	X					
UPDATE	Autorise l'utilisation de UPDATE	X	X	X	X		

Révocation des privilèges (REVOKE)

- Afin de supprimer un ou plusieurs privilèges donnés à des utilisateurs, on utilise la commande REVOKE suivant cette syntaxe :

```
REVOKE  
    Privilege1 [,privilege2] ..  
    ON [type_objet] privilege_level  
    FROM utilisateur1 [, utilisateur2] ...;
```

- Il faut spécifier :
 - Une liste de privilèges séparés par des virgules qu'on veut révoquer d'un compte d'utilisateur après le mot-clé REVOKE ;
 - Le type d'objet et le niveau de privilège après le mot-clé ON ;
 - Un ou plusieurs comptes d'utilisateur dont vous souhaitez révoquer les privilèges dans la clause FROM.

03 - Administrer une base de données

Commandes de gestion des privilèges de base



Révocation des privilèges (REVOKE)

Exemples :

```
REVOKE  
ALL , GRANT OPTION  
FROM Ahmad@localhost;
```

```
REVOKE SELECT,UPDATE,DELETE  
ON dbtest.*  
FROM Ahmad@localhost, str@localhost;
```

- L'instruction REVOKE prend effet selon le niveau de privilège :
 - **Niveau global** : Les modifications prennent effet lorsque le compte utilisateur se connecte au serveur MySQL dans les sessions suivantes. Les modifications ne sont pas appliquées à tous les utilisateurs actuellement connectés.
 - **Niveau base de données** : Les modifications prennent effet après la prochaine instruction USE.
 - **Niveaux table et colonne** : Les modifications prennent effet sur toutes les requêtes suivantes.