Exercices SQL avec MySQL

Table des matières

Section 1 Généralités

1.1 Contenu du document

- Ce document décrit des exercices de base sur l'emploi de requêtes SQL avec l'emploi du SGBD (gestionnaire de base de données) MySQL.
- Ces exercices se feront d'abord avec l'outil phpMyAdmin
- Des requêtes SQL seront ensuite exécutées depuis une fenêtre commande.
- Une étude de cas complète ces exercices.

Vous allez créer un document de type Word et y mettre au fil de l'eau les réponses aux divers exercices.

Ce document porte votre nom.

Il contient pour chaque exercice :

- un rappel du numéro de l'exercice
- la commande SQL que vous avez utilisé (cela peut être une copie d'écran)
- le résultat de la commande, sous forme d'une copie d'écran

Ce document est d'abord pour vous et vous sera probablement utile dans le futur.

Je le demanderai sans doute en fin de journée pour voir l'état d'avancement.

Document unique, sous forme Word ou pdf à l'adresse (pas de collections de png ou jpeg...)

contact@pragma-tec.fr

Section 2 Installation de l'environnement

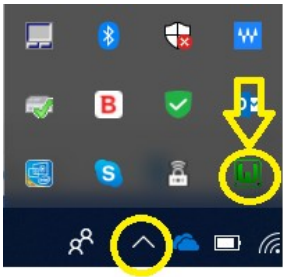
Nous allons avoir besoin du SGBD MySQL puis de l'outil phpMyAdmin.

2.1 Environnement MySQL et php

Pour MySQL il existe plusieurs moyens de l'installer :

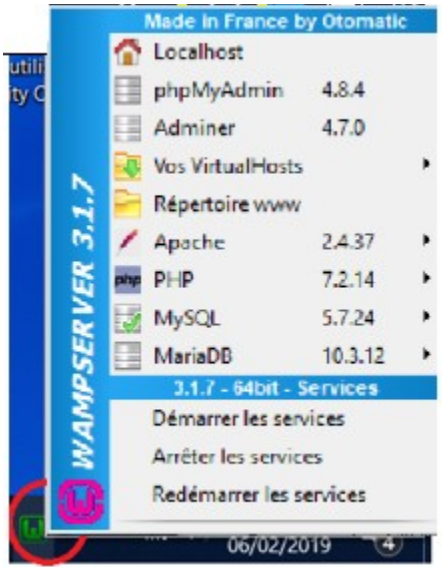
- Installation de WampServer http://www.wampserver.com/ pour Windows
- Ou installation de xampp https://www.apachefriends.org/fr/download.html pour Windows, Linux, Mac.
  Si le poste ne dispose pas de Java (tester en tapant java –version dans une fenêtre commande), il est nécessaire d'installer le package Java (https://www.java.com/fr/download/).

2.1.1 Si WampServer choisi : Lancer WampServer



Pour lancer WampServer, utiliser l'icône du bureau

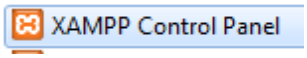
Une fenêtre apparaît



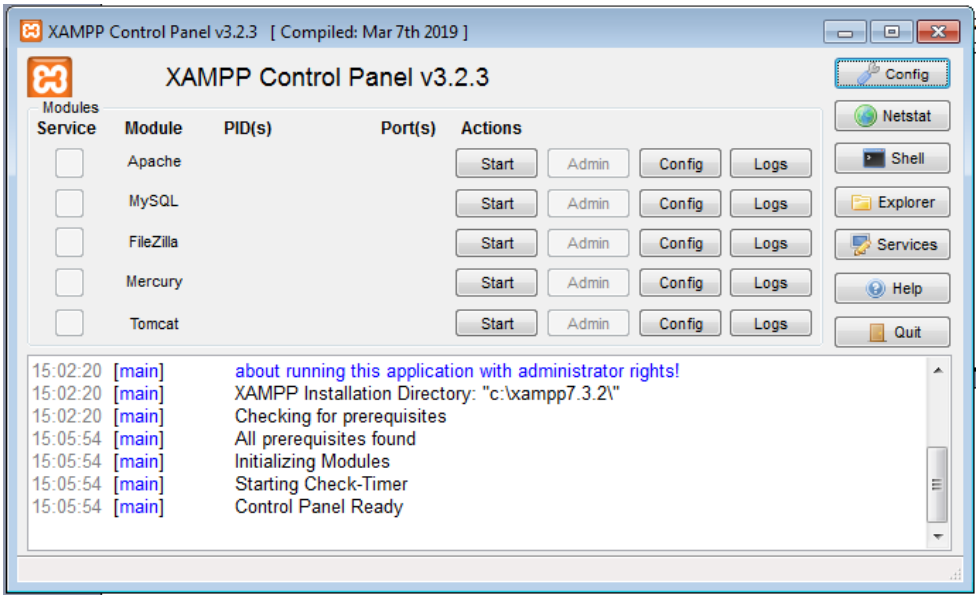
Si l'icône est rouge, cliquez sur « Démarrer les services » puis attendez que l'icône de WAMP devienne verte. Si l'icône reste orange, cliquez sur « Redémarrer les services »

Ensuite phpMyAdmin peut être lancé.

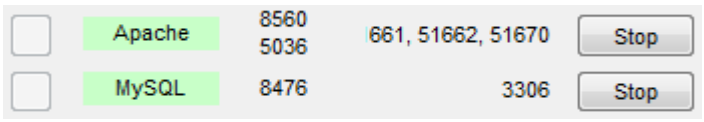
2.1.2 Si Xampp choisi :Lancer Xampp

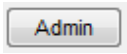
Pour xampp, lancer 

La fenêtre suivante apparaît :



Cliquer sur les boutons Start des lignes Apache et MySQL. Le résultat suivant doit apparaître :



Puis cliquer sur le bouton 

de la ligne MySQL pour lancer phpMyAdmin.

Si erreur **XAMPP - Port 80 in use by "Unable to open process" with PID 4**

Taper dans une fenêtre cmd :

'**Net stop was /y**'

Puis lancer l'application services.msc

Désactiver les services :

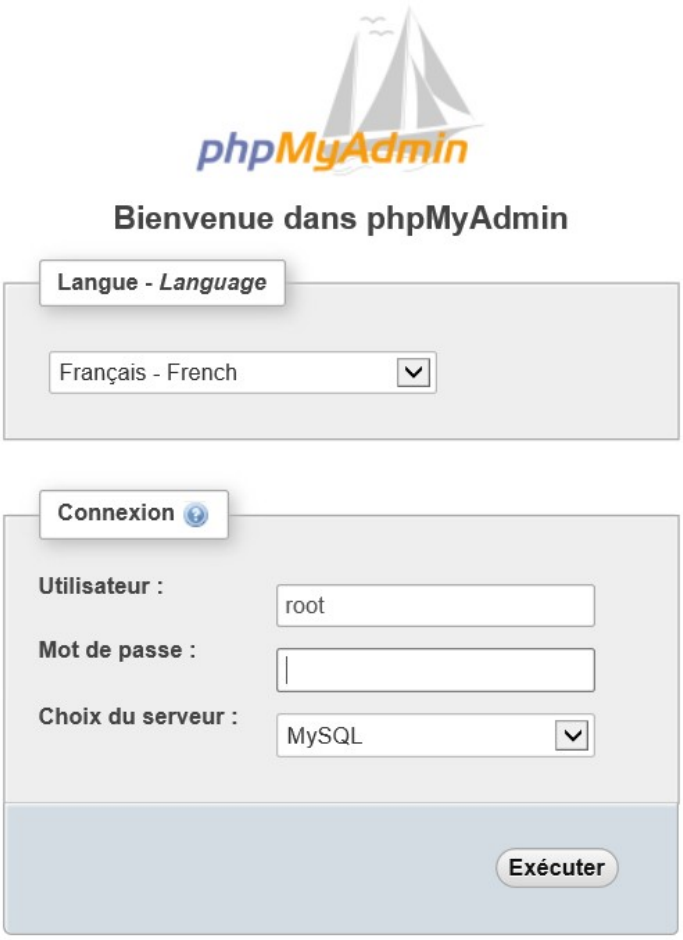
- WWW-Publishing
- Web Deployment Agent Service

Si le problème de conflit avec le port 80 subsiste, se reporter au dernier chapitre.

Section 3 Utiliser MySQL – outil phpMyAdmin

3.1 Premier pas

Quand phpMyAdmin s'ouvre, la partie centrale contient

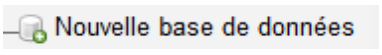


3.2 Se connecter :

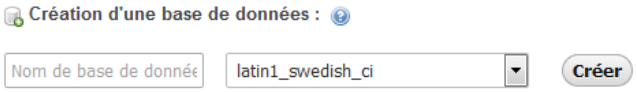
Saisir l'utilisateur root et mot de passe vide. Cliquer sur Exécuter

3.3 Créer une base de données :

Nous allons créer une base de données nommée ExoSQL

Clic sur 

en partie gauche.



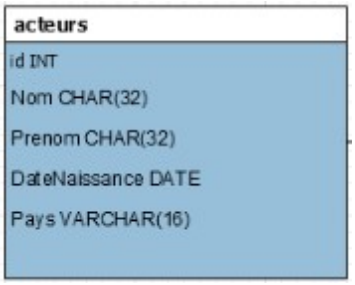
Dans le champ 

saisir ExoSQL, et l'encoding suivant 

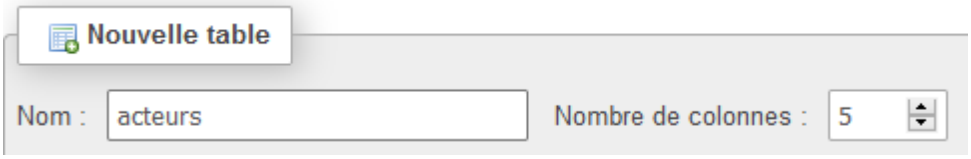
puis 

3.4 Créer une table

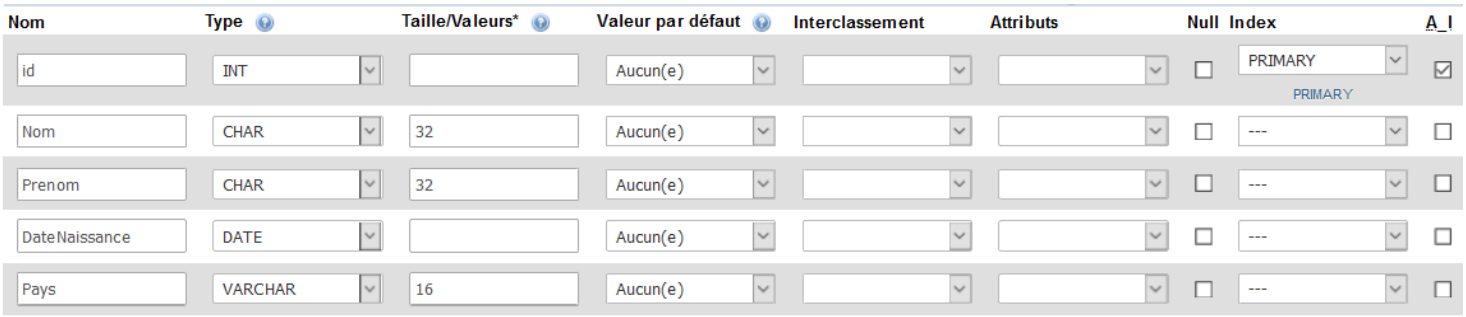
Nous allons créer la table acteurs correspondant à ce dessin :



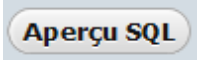
Depuis phpMyAdmin, dans l'onglet Structure, saisir la table acteurs avec 5 colonnes



Saisir les colonnes comme suit :



Choisir comme moteur de stockage InnoDB

Appuyer sur le bouton 

3.5 Appuyer ensuite sur Insérer des lignes

Saisir comme suit dans l'onglet Insérer :

id	int(11)		
Nom	char(32)		Cage
Prenom	char(32)		Nicolas
DateNaissance	date		1964-03-14
Pays	varchar(16)		Angleterre

Colonne	Type	Fonction	Null	Valeur
id	int(11)			
Nom	char(32)			Lambert
Prenom	char(32)			Wilson
DateNaissance	date			1958-04-14
Pays	varchar(16)			France

Aperçu SQL

Puis clic UNE SEULE FOIS sur Exécuter

3.6 Charger une base de données :

Les manipulations précédentes ont permis de se familiariser avec l'environnement et quelques commandes.

Nous allons avoir besoin de plus de données ...

Un contenu existe dans le fichier acteursFilm.sql

Nous allons l'importer :

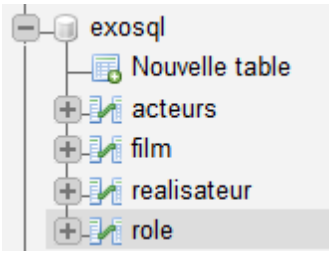
Cliquer sur l'onglet Importer au centre de l'image.

Cliquer sur Parcourir... puis choisir acteursFilm.sql.

Clic sur Exécuter

3.7 Regarder le contenu des tables :

Le résultat doit montrer **l'arborescence** suivante :

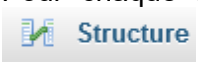


Regarder dans la partie centrale le contenu de chaque table en cliquant sur chaque nom de table.

Notez que c'est l'onglet 

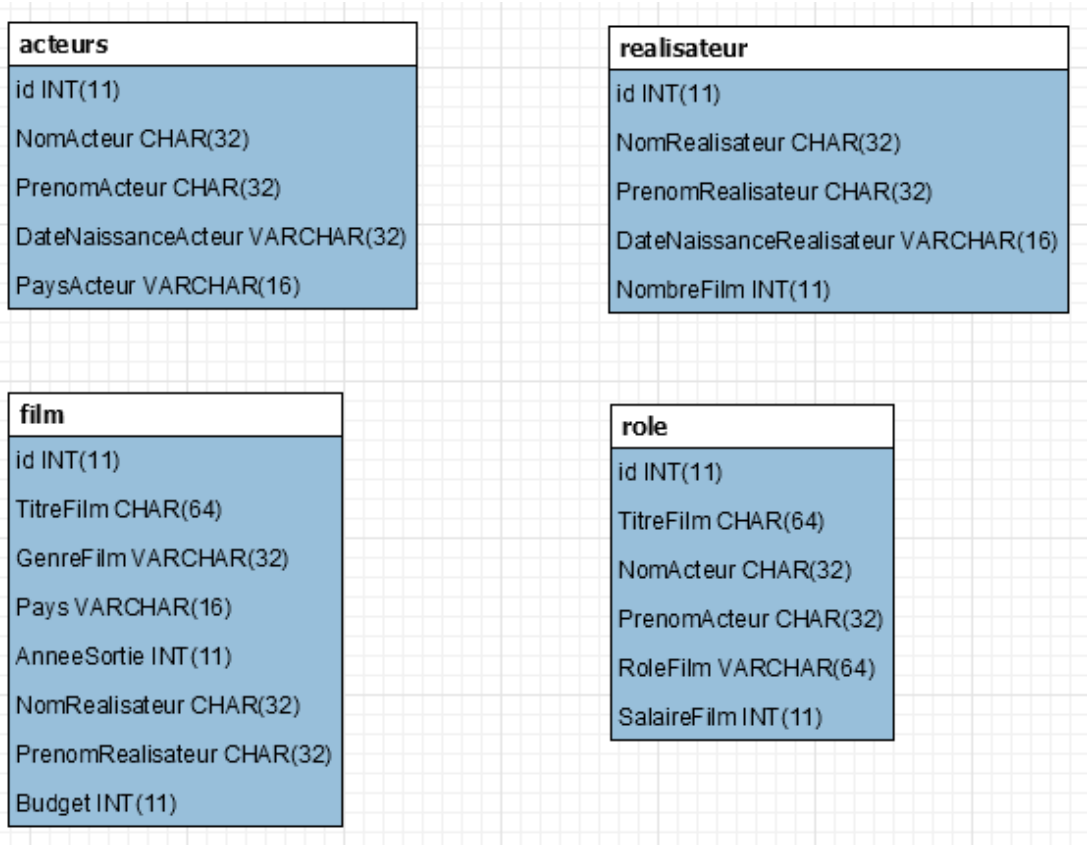
qui affiche le contenu

3.8 Regarder la structure des tables :

Pour chaque table, cliquer sur son nom dans l'arborescence puis sur l'onglet 

Regarder les différents types de données sur le site
https://www.w3schools.com/sql/sql_datatypes.asp

La structure peut être représentée comme suit :


acteurs
id INT(11)
NomActeur CHAR(32)
PrenomActeur CHAR(32)
DateNaissanceActeur VARCHAR(32)
PaysActeur VARCHAR(16)
realisateur
id INT(11)
NomRealisateur CHAR(32)
PrenomRealisateur CHAR(32)
DateNaissanceRealisateur VARCHAR(16)
NombreFilm INT(11)
film
id INT(11)
TitreFilm CHAR(64)
GenreFilm VARCHAR(32)
Pays VARCHAR(16)
AnneeSortie INT(11)
NomRealisateur CHAR(32)
PrenomRealisateur CHAR(32)
Budget INT(11)
role
id INT(11)
TitreFilm CHAR(64)
NomActeur CHAR(32)
PrenomActeur CHAR(32)
RoleFilm VARCHAR(64)
SalaireFilm INT(11)

3.9 Rechercher des données

La table acteurs contient :

NomActeur	PrenomActeur	DateNaissanceActeur	PaysActeur
Bale	Christian	30 janvier 1974	Angleterre
Bean	Sean	17 avril 1959	Angleterre
Cage	Nicolas	7 janvier 1964	USA
Caine	Michael	14 mars 1933	Angleterre
Connery	Sean	25 aout 1930	Angleterre
Cotillard	Marion	30 septembre 1975	France
Depardieu	Gerard	27 decembre 1948	France
Diaz	Cameron	30 aout 1972	USA
DiCaprio	Leonardo	11 novembre 1974	USA
Fox	Megan	16 mai 1986	USA
Jackman	Hugh	12 octobre 1968	Australie
Johansson	Scarlett	22 novembre 1984	USA
LaBeouf	Shia	11 juin 1986	USA
MacDowell	Andie	21 avril 1958	USA
MacGregor	Ewan	31 mars 1971	Angleterre
Marsden	James	18 septembre 1973	USA
Murray	Bill	21 septembre 1950	USA
Prime	Optimus	Inconnue	Cybertron
Wilson	Lambert	3 aout 1958	France
Worthington	Sam	2 aout 1976	USA

Nous allons faire des recherches sur certaines lignes (entrées)

Utiliser l'onglet 

pour rechercher :

- Les lignes dont le pays de l'acteur est USA
- Les lignes des acteurs nés en septembre
- Les lignes des acteurs qui NE SONT PAS nés en septembre
- Les lignes dont le pays de l'acteur est USA et nés en 1950

La table role contient :

TitreFilm	NomActeur	PrenomActeur	RoleFilm	SalaireFilm
Equilibrium	Bale	Christian	John Preston	760
Equilibrium	Bean	Sean	Partridge	60
Ghost rider	Cage	Nicolas	Johnny Blaze	15000
Il etait une fois	Marsden	James	Le prince Edouard	500
Inception	Caine	Michael	Miles	75
Inception	Cottillard	Marion	Mall	850
Inception	DiCaprio	Leonardo	Dom Cobb	20000
Jennifers body	Fox	Megan	Jennifer Check	11000
Le prestige	Bale	Christian	Alfred Borden	1700
Le prestige	Caine	Michael	Cutter	350
Le prestige	Jackman	Hugh	Robert Angier	1800
Le prestige	Johansson	Scarlett	Olivia	950
Lhomme au masque de fer	Depardieu	Gerard	Porthos	1
Lhomme au masque de fer	DiCaprio	Leonardo	Louis XIV	3500
Matrix reloaded	Wilson	Lambert	Le Merovingien	10000
Rock	Cage	Nicolas	Stanley Goodspeed	1500
Rock	Connery	Sean	John Patrick Mason	6000
Terminator renaissance	Bale	Christian	John Connor	12000
Terminator renaissance	Worthington	Sam	Marcus Wright	9000
The Box	Diaz	Cameron	Norma Lewis	3000
The Box	Marsden	James	Arthur Lewis	1000
The island	Bean	Sean	Merrick	150
The island	Johansson	Scarlett	Jordan 2delta	900
The island	MacGregor	Ewan	Lincoln 6echo	2000
Transformers	Fox	Megan	Mikaela Banes	7000
Transformers	LaBeouf	Shia	Sam Witwicky	13000
Transformers	Prime	Optimus	Lui-meme	0
Transformers 2 : la revanche	Fox	Megan	Mikaela Banes	12000
Transformers 2 : la revanche	LaBeouf	Shia	Sam Witwicky	14000
Transformers 2 : la revanche	Prime	Optimus	Lui-meme	0
Un jour sans fin	MacDowell	Andie	Rita	250
Un jour sans fin	Murray	Bill	Phil Connors	600
Xmen origins : Wolverine	Jackman	Hugh	Logan	5000

Chercher les lignes :

- Des salaires de Fox dans Jennifers body et celui de Bale dans Le prestige
- Lignes dont le salaire est compris entre 400 et 800

3.10 Modification de table

Ajouter dans la bonne table le film anglais (policier) « Sherlock Holmes » sorti en 2009. L'acteur américain Robert Downey y joue le rôle de Sherlock Holmes et Jude Law celui de John Watson. Pour des salaires respectifs de 2500 et 2000.

3.11 Créer une table

Créer la nouvelle table cinema pour obtenir ce qui suit :

	#	Nom	Type	Interclassement	Attributs	Null	Valeur par défaut
☐	1	**nom**	varchar(20)	utf8_unicode_ci		Non	*Aucun(e)*
☐	2	**codepostal**	int(11)			Non	*Aucun(e)*
☐	3	**film**	varchar(64)	utf8_unicode_ci		Non	*Aucun(e)*
☐	4	**datedebut**	date			Non	*Aucun(e)*
☐	5	**datefin**	date			Non	*Aucun(e)*

Rentrer quelques lignes pour obtenir environ ceci :

nom	codepostal	film	datedebut	datefin
ugc	92520	Equilibrium	2019-04-01	2019-04-09
gaumont	92700	Il etait une fois	2018-04-01	2018-04-09
gaumont	92700	Terminator renaissance	2019-02-01	2019-02-09

3.12 Créer un utilisateur

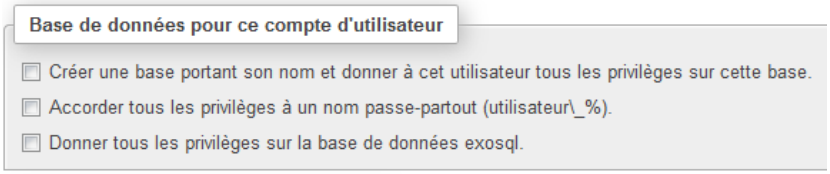
Nous allons créer un utilisateur qui ne peut que regarder le contenu de la BD.

Créer **un nouvel utilisateur**, ne pas toucher à l'utilisateur par défaut root

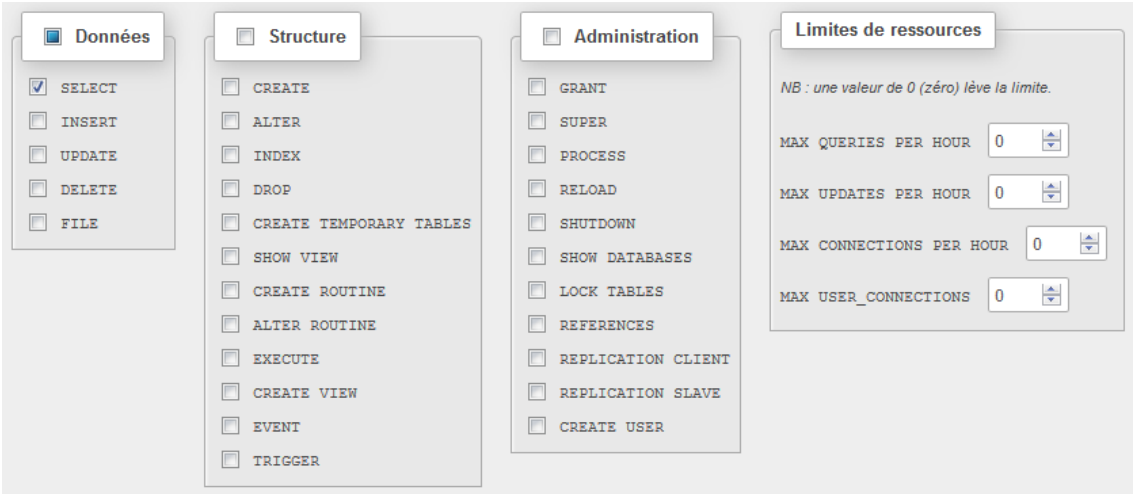
Ouvrir l'onglet Privilèges

Cliquer sur Ajouter un compte d'utilisateur

Décocher les coches suivantes



Dans données, cocher uniquement select



Section 4 Utilisation directe du langage SQL

4.1 Généralités

L'utilisation de phpMyAdmin permet de réaliser des opérations de base de façon pratique, graphique, et assez intuitive. Ce n'est qu'un outil d'investigation parmi d'autres.

MySQL Workbench est un autre outil utilisable pour le développement, en particulier lors de la conception par l'édition de diagrammes.

Une application qui utilise une base de données a ses propres moyens de créer la base de données et de la gérer.

La création d'une base de données peut se faire par l'exécution de scripts SQL (du code dans des fichiers) ou par programme.

La gestion se fait par programme (écrit en php, Java, C# , Python…).

Dans tous les cas il est nécessaire de connaître le langage SQL.

Quelques bases sur le SQL – Search Query Language :

- Toute commande s'appelle une requête
- Une requête se termine par ' ;'
- Si le ' ;' est oublié dans la fenêtre de commande la ligne suivante commence par '->' pour montrer que la suite, terminée par ; est attendue
- Une requête peut être écrite sur plusieurs lignes
- Insensible à la casse . SELECT équivalent à sEleCT

Le site https://www.w3schools.com/sql/sql_syntax.asp va nous aider.

4.2 Lancement du client MySQL

Il faut d'abord ouvrir une fenêtre commande :

Si utilisation de WampServer, ouvrir le menu, choix MySQLDB ->Console MySQLDB

Accepter le mot de passe root, puis touche Entrée pour le mot de passe

Si utilisation de Xampp : dans la fenêtre xampp, cliquer sur 

Dans le fenêtre commande obtenue, taper

```
Mysql –u root –p exosql
```

4.3 Regarder la structure générale de la base de données : show

Commande show

`Show databases ;` montre toutes les bases de données

`Show tables from <nom base> ;` montre toutes les tables d'une base

`Show columns from < nom base.nom tables> ;` montre la constitution des colonnes d'une table

`Show create table <nom table> ;` montre le code SQL qui a permis de créer la table

4.4 Regarder le contenu de la base de données : select

La commande select a une syntaxe très riche

https://dev.mysql.com/doc/refman/8.0/en/select.html

Nous allons explorer quelques cas.

```
Select <le résultat que l'on souhaite voir> from <table> ;
```

Select * from acteurs ;

Select nomacteur, paysacteur from acteurs ;

Il est possible d'appliquer des filtres :

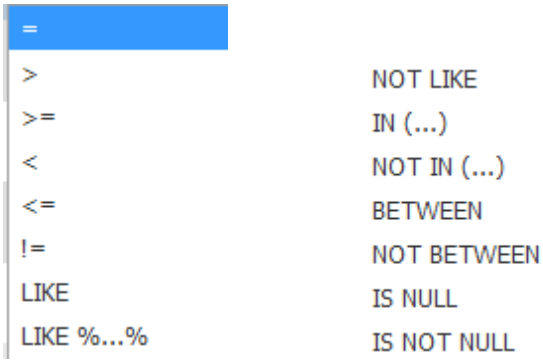
```
Select <le résultat> from <table> where <clause>;
```

La « clause » where définit le filtre, la condition. C'est ce que l'on a vu dans l'onglet 

de PhpMyAdmin

Clause where : where <colonne> <opérateur> <valeur>

Voici une partie des opérateurs :



<valeur> peut être un nombre entier ou flottant. Pour un texte ‘ ou ” doivent encadrer ce texte.

Ecrire les requêtes SQL pour les énoncés suivants

- Les lignes dont le pays de l’acteur est USA (utiliser like ou =)
- Les lignes dont le pays de l’acteur est USA ou France (utiliser or)
- Les lignes des acteurs nés en septembre(utiliser like et %)
- Les lignes des acteurs qui NE SONT PAS nés en septembre (not like)
- Les lignes dont le pays de l’acteur est USA et nés en 1950 (and)

Chercher les lignes :

- Des salaires de Fox dans Jennifers body et celui de Bale dans Le prestige
- Lignes dont le salaire est compris entre 400 et 800 (regarder between sur w3school)

Regrouper des lignes et faire un traitement :

- Afficher la somme des salaires de chaque film (utiliser sum() et group by)
- Afficher la table role par ordre croissant des salaires (order by)

4.5 Assembler/joindre plusieurs tables

Il est possible d’assembler plusieurs tables en une seule, pour l’afficher ou réaliser des traitements.

Ce mécanisme d’assemblage s’appelle **jointure.**

Pour pouvoir joindre 2 tables, il faut trouver des éléments communs dans les 2 tables pour que le SGBD (système de gestion base de données) trouve la correspondance.

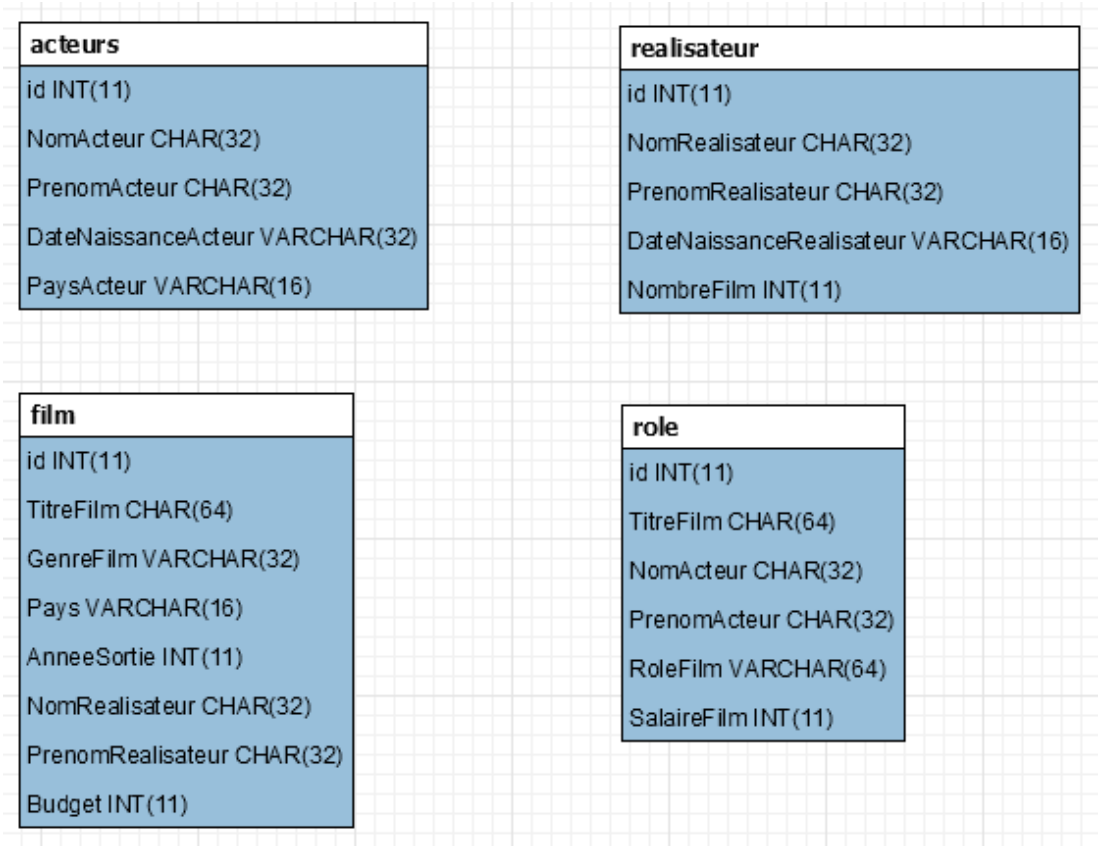
Par exemple pour faire correspondre les tables acteurs et role, nous allons utiliser NomActeur et PrenomActeur qui sont en commun dans les deux tables.

Exemple (ancienne syntaxe) :

```
select acteurs.nomacteur, acteurs.paysacteur, role.titrefilm from acteurs, role where acteurs.nomacteur = role.nomacteur and acteurs.prenomacteur = role.prenomacteur;
```

Même exemple (syntaxe préconisée) :

```
select acteurs.nomacteur, acteurs.paysacteur, role.titrefilm from acteurs inner join role ON acteurs.nomacteur = role.nomacteur and acteurs.prenomacteur = role.prenomacteur;
```



Identifier les liens possibles entre les tables de cette base de données.

Afficher le nom des acteurs et nom des réalisateurs ayant en commun un film.

Afficher le nom des acteurs, leur pays et le nom des réalisateurs ayant en commun un film.(utiliser une double jointure)

4.6 Copier une table, supprimer et créer une table

Pour créer une table qui une copie exacte ou partielle d'une table existante :

(lien http://www.mysqltutorial.org/mysql-copy-table-data.aspx)

```
CREATE TABLE new_table
SELECT col, col2, col3
FROM
    existing_table;
```

```
CREATE TABLE new_table
SELECT col1, col2, col3
FROM
    existing_table
WHERE
    conditions;
```

Créer et copier la table cinema dans la table cinemabis

Supprimer le contenu d'une table : **`truncate table <nom table>;`**

Supprimer le contenu de la table cinemabis

Supprimer une table : **`drop table <nom table>`**

Supprimer la table cinemabis

Créer une table, exemple :

```
CREATE TABLE Persons (
    PersonID int,
    LastName varchar(255),
    FirstName varchar(255),
    Address varchar(255),
    City varchar(255)
);
```

Regarder l'ordre de création de la table cinéma : **`show create table cinema;`**

4.7 Contenu du fichier source acteursFilm.sql

Regardons et commentons le contenu du fichier.

Dans la fenêtre commande pour prendre en compte un fichier source la commande est

Source <chemin/nomfichier.sql>

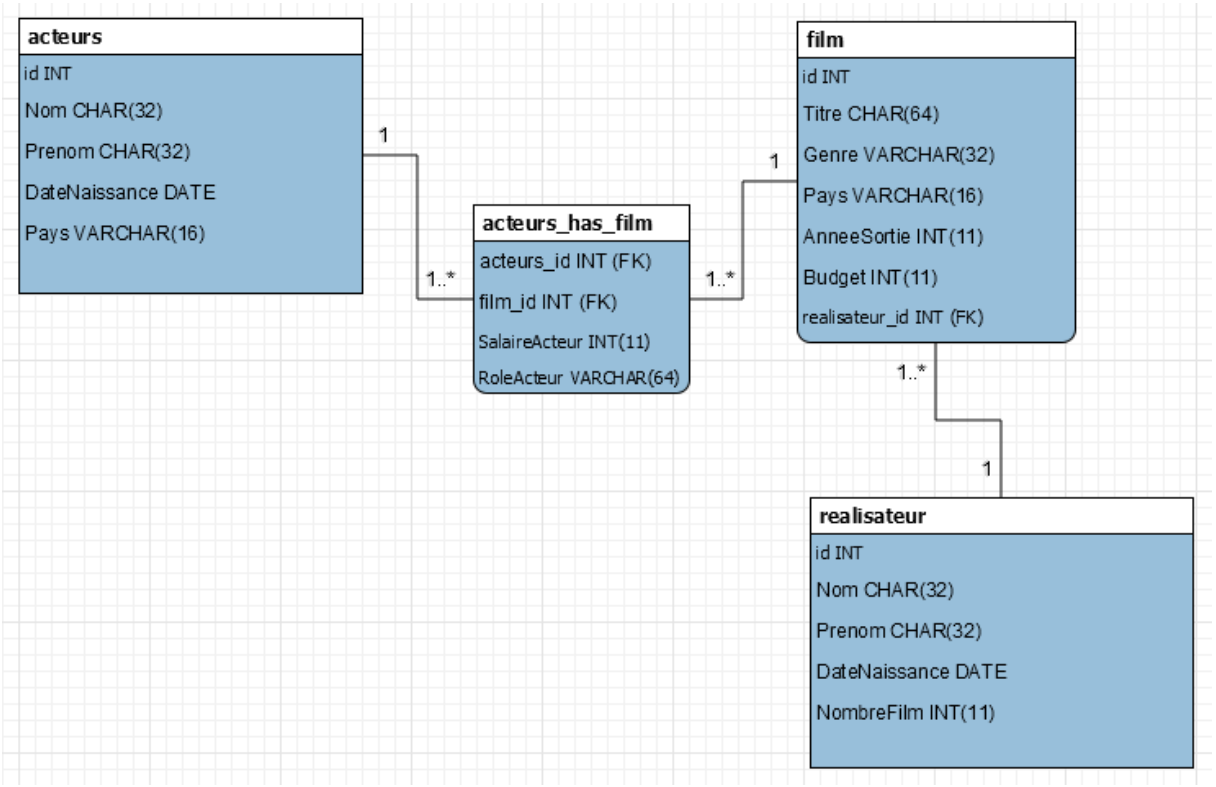
Le chemin et nom du fichier ne doivent pas comporter d'espace et caractères accentués. Le séparateur de chemin est '/' et non pas '\'

4.8 Critique de la conception de cette BD

La conception de cette BD laisse à désirer. Le résultat en est, en particulier :

- Des colonnes sont répétées dans plusieurs tables
- Le nom des colonnes est inadapté
- Il n'y a aucun lien dans le modèle.
- Les dates sont des chaînes de caractères et pas de type DATE

Voici le modèle après révision et simplification (schéma de forme UML):



Cette BD se nomme exosqlplus et est disponible dans le fichier scriptActeurRole.sql
Regardons le contenu.
Les valeurs '1' et '1..*' se nomment la « cardinalité ».

Section 5 Base de données employés

5.1 Chargement de la base

Nous allons travailler essentiellement avec la fenêtre commande.

Vous pouvez utiliser aussi phpMyAdmin pour regarder des résultats. Mais cette application est trop limitée pour ce que nous allons faire.

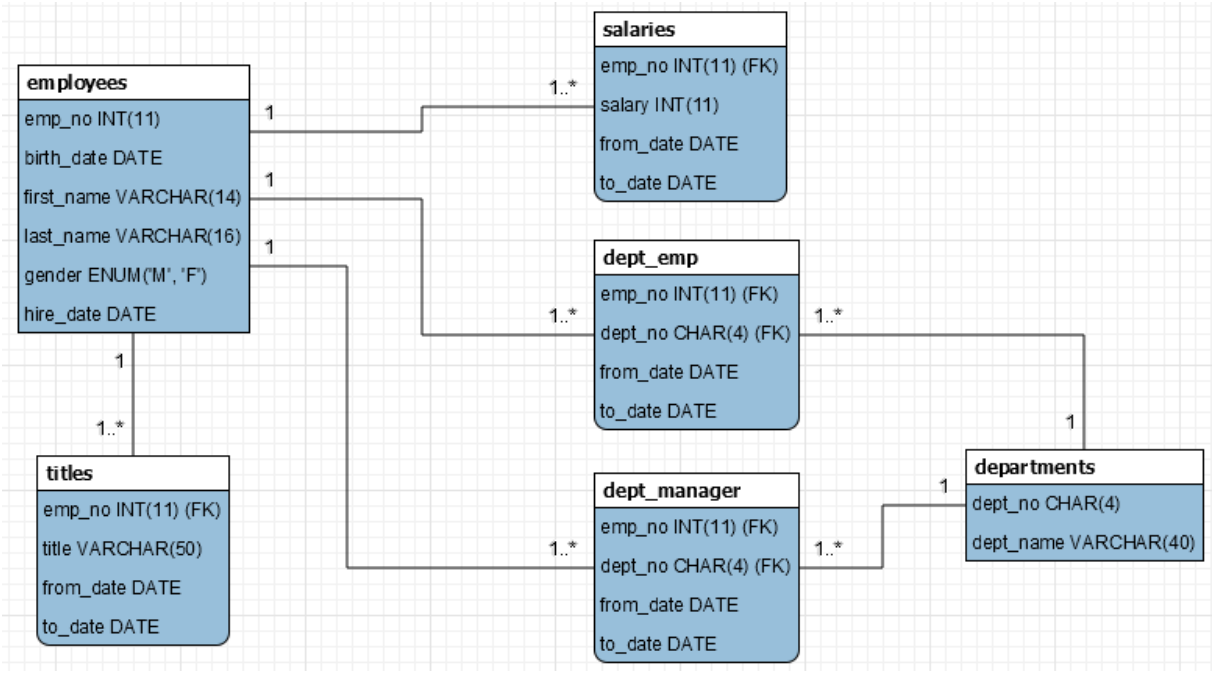
Dans la fenêtre commande charger le source employees.sql

Source <chemin avec / sans espace>/ employees.sql

5.2 Analyser le contenu

Utiliser show tables ; puis des select * sur ces tables ….

Le schéma obtenu par l'outil MysqlWorkBench montre le schéma suivant (format UML)



Ce schéma montre que cette base :

- liste des employés d'une société, table employees (hire date signifie date d'embauche). Emp_no est la clé primaire

- la tables salaries (salaires) permet de tracer l'évolution (l'historique) des salaires des employés. Le 1..* signifie qu'un salarié peut avoir une ou plusieurs lignes de salaire dans cette table.
  emp_no qui fait référence à employees.emp_no est ici une « clé étrangère – foreign key »
- La table titles permet de tracer les différents postes des employés. Noter ici aussi 1..*
- La table dept_emp permet de tracer les différents départements occupés par les salariés. Il y a deux clés étrangères pour les relations entre les tables employees et departments
- La table dept_manager permet d'identifier les employés qui sont des managers.

5.3 Réaliser des requêtes

1. Obtenir la liste des plus vieux employés.
   S'y prendre en plusieurs temps :
   - Utiliser la fonction MIN() dans un select pour récupérer la plus ancienne date.
   - Y ajouter year() pour n'avoir que l'année
   - réaliser un select imbriqué dans lequel le select précédent fait partie du where … courage
2. Idem avec la liste des plus jeunes
3. Obtenir la liste des femmes employées les plus jeunes
4. Obtenir la liste des hommes employés avec la date d'embauche la plus ancienne
5. Afficher les noms et prénoms des managers
6. Afficher l'évolution des salaires de l'employé dont emp_no vaut 10001 ;
7. Rechercher le nom de ce salarié
8. En une seule requête afficher l'évolution des salaires de Facello
9. Afficher les noms et derniers salaires des employés.
   Afficher d'abord les lignes de salaries correspondant aux derniers salaires.
   Exploiter au besoin la valeur 9999 ...
10. Afficher le dernier emploi des salariés
11. Utiliser UNION
    Afficher par un UNION les employés de no 10001 et 10003

Regarder l'emploi de EXIST.

5.4 Procédures stockées

Généralités, chaque partie est reprise en détail.

- Une procédure stockée est un **ensemble d'instructions** que l'on peut exécuter sur commande.
- Une procédure stockée est un objet de la base de données **stocké de manière durable**, au même titre qu'une table. Elle n'est pas supprimée à la fin de la session comme l'est une requête préparée.
- On peut passer des **paramètres** à une procédure stockée, qui peuvent avoir trois sens : `IN`(entrant), `OUT` (sortant) ou `INOUT` (les deux).
- `SELECT ... INTO` permet d'assigner des données sélectionnées à des variables ou des paramètres, à condition que le `SELECT` ne renvoie qu'une seule ligne, et qu'il y ait autant de valeurs sélectionnées que de variables à assigner.
- Les procédures stockées peuvent permettre de **gagner en performance** en diminuant les allers-retours entre le client et le serveur. Elles peuvent également aider à **sécuriser une base de données** et à s'assurer que les traitements sensibles sont toujours exécutés de la même manière.
- Par contre, elle **ajoute à la charge du serveur** et sa syntaxe n'est **pas toujours portable** d'un SGBD à un autre.

Les exemples suivants sont inspirés du site :

https://openclassrooms.com/fr/courses/1959476-administrez-vos-bases-de-donnees-avec-mysql/1971667-procedures-stockees

5.4.1 Créer une procédure

Procédure qui ne contient qu'une requête :

```
1 CREATE PROCEDURE afficher_races_requete()
2     -- pas de paramètres dans les parenthèses
3 SELECT id, nom, espece_id, prix FROM Race;
```

Procédure avec un bloc de requêtes : les mots BEGIN et END sont nécessaires.

```
1 DELIMITER | -- On change le délimiteur
2 CREATE PROCEDURE afficher_races()
3     -- toujours pas de paramètres, toujours des parenthèses
4 BEGIN
5     SELECT id, nom, espece_id, prix
6     FROM Race;  -- Cette fois, le ; ne nous embêtera pas
7 END|            -- Et on termine bien sûr la commande CREATE PROCEDURE par notre nouveau
  délimiteur
```

L'instruction DELIMITER permet de modifier le délimiteur par défaut ' ;' le temps d'une session.

Cette instruction est nécessaire pour que le SGBD ne croit pas que Race; est la fin de la définition de la procédure.

C'est bien 'END|' qui est la fin de la procédure.

Ne pas oublier de remettre ensuite DELIMITER ;

Avant de voir le passage de paramètres, pour appeler une procédure :

CALL nomProcedure() ;

⇒ Dans la base employees, créer une procédure ageMiniMaxi() qui affiche l'âge minimal et maximal des employés, homme et femme.

Procédures avec paramètres :

Il faut indiquer un Sens aux paramètres :

Un paramètre peut être de trois sens différents : entrant (`IN`), sortant (`OUT`), ou les deux (`INOUT`).

- `IN` : c'est un paramètre "entrant". C'est-à-dire qu'il s'agit d'un paramètre dont la valeur est fournie à la procédure stockée. Cette valeur sera utilisée pendant la procédure (pour un calcul ou une sélection, par exemple).

- `OUT` : il s'agit d'un paramètre "sortant", dont la valeur sera établie au cours de la procédure et qui pourra ensuite être utilisé en dehors de cette procédure.
- `INOUT` : un tel paramètre sera utilisé pendant la procédure, verra éventuellement sa valeur modifiée par celle-ci, et sera ensuite utilisable en dehors.

Lorsque l'on crée une procédure avec un ou plusieurs paramètres, chaque paramètre est défini par trois éléments.

- Son sens : entrant, sortant, ou les deux. Si aucun sens n'est donné, il s'agira d'un paramètre `IN` par défaut.
- Son nom : indispensable pour le désigner à l'intérieur de la procédure.
- Son type : `INT`, `VARCHAR(10)`…

Exemple

```
DELIMITER | -- Facultatif si votre délimiteur est toujours |
CREATE PROCEDURE afficher_race_selon_espece (IN p_espece_id INT)
    -- Définition du paramètre p_espece_id
BEGIN
    SELECT id, nom, espece_id, prix
    FROM Race
    WHERE espece_id = p_espece_id;  -- Utilisation du paramètre
END |
DELIMITER ;  -- On remet le délimiteur par défaut
```

Appel de la procédure :

```
CALL afficher_race_selon_espece(1);
SET @espece_id := 2;
CALL afficher_race_selon_espece(@espece_id);
```

SET @espece_id:= 2 ; permet de créer une variable utilisateur dont la durée de vie est la session (non liée aux mécanismes de procédures stockées).

⇒ Dans la base employees, créer une procédure salaireMaxi() qui prend pour paramètre le genre et affiche le salaire correspondant.

Procédure avec paramètre OUT :

D'abord une expérience :

```
1 SELECT id, nom INTO @var1, @var2
2 FROM Animal
3 WHERE id = 7;
4 SELECT @var1, @var2;
```

@var1	@var2
7	Caroline

Le mot INTO redirige un résultat dans une ou plusieurs variables. Chaque variable par contre ne contient qu'une valeur.

Le SELECT précédent ne doit renvoyer qu'une seule ligne.

D'où la procédure avec 2 paramètres, un entrant, un sortant :

```
1 DELIMITER |
2 CREATE PROCEDURE compter_races_selon_espece (p_espece_id INT, OUT p_nb_races INT)
3 BEGIN
4     SELECT COUNT(*) INTO p_nb_races
5     FROM Race
6     WHERE espece_id = p_espece_id;
7 END |
8 DELIMITER ;
```

Appel de la procédure :

```
1 CALL compter_races_selon_espece (2, @nb_races_chats);
```

Afficher le résultat :

```
1 SELECT @nb_races_chats;
```

⇒ Dans la base employees, créer une procédure salaireMaxiOut() qui prend pour paramètre le genre et rend le salaire correspondant dans un autre paramètre de sortie.

Un paramètre de type INOUT combine les deux usages, INTO est aussi utilisé pour changer sa valeur en sortie.

5.4.2 Créer un algorithme dans une procédures

Ce que nous allons voir :

Un bloc d'instructions est délimité par `BEGIN` et `END`. Il est possible d'imbriquer plusieurs blocs d'instructions.

- Une variable locale est définie dans un bloc d'instructions grâce à la commande `DECLARE`. Une fois la fin du bloc d'instructions atteinte, toutes les variables locales qui y ont été déclarées sont supprimées.
- Une structure conditionnelle permet d'exécuter une série d'instructions si une condition est respectée. Les deux structures conditionnelles de MySQL sont `IF` et `CASE`.
- Une boucle est une structure qui permet de répéter une série d'instructions un certain nombre de fois. Il existe trois types de boucles pour MySQL : `WHILE`, `REPEAT` et `LOOP`.
- L'instruction `LEAVE` permet de quitter un bloc d'instructions ou une boucle.
- L'instruction `ITERATE` permet de relancer une itération d'une boucle.

Cette partie est inspirée de

https://openclassrooms.com/fr/courses/1959476-administrez-vos-bases-de-donnees-avec-mysql/1972254-structurez-vos-instructions

Dans une procédure on peut déclarer une variable locale :

```
1 DECLARE nom_variable type_variable [DEFAULT valeur_defaut];
```

Regarder et essayer l'exemple qui suit :

https://openclassrooms.com/fr/courses/1959476-administrez-vos-bases-de-donnees-avec-mysql/1972254-structurez-vos-instructions#/id/r-1989271

Structure conditionnelle : le IF

```
IF condition THEN instructions
[ELSEIF autre_condition THEN instructions
[ELSEIF ...]]
[ELSE instructions]
END IF;
```

⇒ Dans la base employees, adapter la procédure salaireMaxi() qui prend pour paramètre le genre et affiche le salaire correspondant ainsi que 'pour les hommes' ou 'pour les femmes' selon le cas. Utiliser le IF

5.5 Les fonctions

Une fonction est une sorte de procédure qui retourne une seule valeur.

Les paramètres sont tous en IN, le mot n'est donc pas présent.

La définition s'accompagne du type de l'unique valeur de retour.

Exemple :

```
DELIMITER $$
CREATE FUNCTION CustomerLevel(credit DECIMAL(10,2))
RETURNS VARCHAR(20)
DETERMINISTIC
BEGIN
   DECLARE customerLevel VARCHAR(20);

   IF credit > 50000 THEN
                SET customerLevel = 'PLATINUM';
   ELSEIF (credit >= 50000 AND credit <= 10000) THEN
      SET customerLevel = 'GOLD';
   ELSEIF credit < 10000 THEN
      SET customerLevel = 'SILVER';
   END IF;
       -- return the customer level
       RETURN (customerLevel);
END$$
DELIMITER ;
```

5.6 Les CURSOR

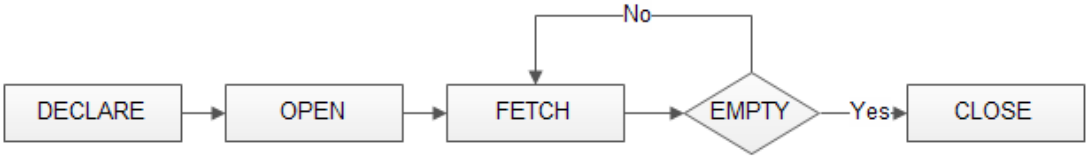
Un curseur peut être employé dans une procédure ou une fonction .

Un Curseur sert à 'décortiquer' un à un les lignes résultantes d'un select.

On ne peut pas mettre à jour le contenu d'un curseur.

Le parcours des données d'un curseur est séquentiel et correspond à l'ordre de restitution fait par le SELECT

Les étapes d'utilisation d'un curseur suit le schéma suivant :




```
DECLARE cursor_name CURSOR FOR SELECT_statement;
```

Cette déclaration doit être après la déclaration des variables locales de la procédure ou fonction.

```
DECLARE CONTINUE HANDLER FOR NOT FOUND SET finished = 1;
```

Ceci permet d'indiquer que quand tous les éléments auront été parcourus, c'est la variable finished qui sera mise à 1. La déclaration de cette variable est à placer avant avec la valeur 0 par défaut.

```
OPEN cursor_name;
```

Cette instruction réalise l'exécution du select associé au curseur. Les données sont prêtes à être lues.

```
FETCH cursor_name INTO variables list;
```

Cette ligne doit être répétée (dans une boucle) jusqu'au moment où finished vaut 1

```
CLOSE cursor_name;
```

Ferme le curseur

Voici un exemple de code : (site https://www.mysqltutorial.org/mysql-cursor/)

```
DELIMITER $$
CREATE PROCEDURE createEmailList ( INOUT emailList varchar(4000))
BEGIN
	DECLARE finished INTEGER DEFAULT 0;
	DECLARE emailAddress varchar(100) DEFAULT "";

	-- declare cursor for employee email
	DECLARE curEmail
		CURSOR FOR
			SELECT email FROM employees;

	-- declare NOT FOUND handler
	DECLARE CONTINUE HANDLER FOR NOT FOUND SET finished = 1;

	OPEN curEmail;

	getEmail: LOOP
		FETCH curEmail INTO emailAddress;
		IF finished = 1 THEN
			LEAVE getEmail;
		END IF;
		-- build email list
		SET emailList = CONCAT(emailAddress,";",emailList);
	END LOOP getEmail;
	CLOSE curEmail;
END$$
DELIMITER ;
```

5.7 Les tables temporaires

- Une table temporaire a au maximum une durée de vie qui est celle de la session.
- Le SGBD ne stocke pas sa définition.
- Une table temporaire est **créée de la même manière qu'une table normale**. Il suffit d'ajouter le mot-clé `TEMPORARY` avant `TABLE`.
- On peut créer une table (temporaire ou non) à partir de la structure d'une autre table avec `CREATE [TEMPORARY] TABLE nouvelle_table LIKE ancienne_table;`.
- On peut créer une table à partir d'une requête `SELECT` avec `CREATE [TEMPORARY] TABLE SELECT ...;`.
- Les tables temporaires permettent de **gagner en performance** lorsque, dans une session on doit exécuter plusieurs requêtes sur un même set de données.
- On peut utiliser les tables temporaires pour créer des **données de test**.
- Enfin, les tables temporaires peuvent être utilisées pour stocker un set de résultats d'une procédure stockée.

Exemple :

```sql
CREATE TEMPORARY TABLE TMP_Animal (
    id INT UNSIGNED PRIMARY KEY,
    nom VARCHAR(30),
    espece_id INT UNSIGNED,
    sexe CHAR(1)
);

DESCRIBE TMP_Animal;
```

SHOW TABLES ; ne montre pas les tables temporaires.

Les opérations qui suivent et concernent une table temporaire ne nécessitent pas le mot TEMPORARY

```sql
ALTER TABLE TMP_Animal
ADD COLUMN date_naissance DATETIME;
```

Pour le DROP TEMPORARY est optionnel, l'utiliser evte de supprimer une vraie table de même nom

```sql
DROP TEMPORARY TABLE TMP_Animal;
```

Si une table temporaire porte le même nom qu'une table permanente, c'est la table temporaire qui est utilisée pour cette session.

Utile pour faire des tests unitaires.

`CREATE TEMPORARY TABLE` et `DROP TEMPORARY TABLE` ne valident pas les transactions comme le font ces instructions pour des tables permanentes.

Utilité de ces tables :

- gain en performance : les données sont disponibles en mémoire
- pour les tests
- pour les procédures stockées

Exemple issu de https://openclassrooms.com/fr/courses/1959476-administrez-vos-bases-de-donnees-avec-mysql/1975343-tables-temporaires#/id/r-1992819 :

```
DELIMITER |
CREATE PROCEDURE table_adoption_non_payee()
BEGIN
    DROP TEMPORARY TABLE IF EXISTS Adoption_non_payee;

    CREATE TEMPORARY TABLE Adoption_non_payee
    SELECT Client.id AS client_id, Client.nom AS client_nom, Client.prenom AS client_prenom,
Client.email AS client_email,
            Animal.nom AS animal_nom, Especе.nom_courant AS espece, Race.nom AS race,
            Adoption.date_reservation, Adoption.date_adoption, Adoption.prix
    FROM Adoption
    INNER JOIN Client ON Client.id = Adoption.client_id
    INNER JOIN Animal ON Animal.id = Adoption.animal_id
    INNER JOIN Espece ON Espece.id = Animal.espece_id
    LEFT JOIN Race ON Race.id = Animal.race_id
    WHERE Adoption.paye = FALSE;
END |
DELIMITER ;

CALL table_adoption_non_payee();

SELECT client_id, client_nom, client_prenom, animal_nom, prix
FROM Adoption_non_payee;
```

5.8 Gestion des exceptions dans une transaction

Voir le site https://gist.github.com/julwong/5ef291cad711846909e1

et https://www.mysqltutorial.org/mysql-stored-procedure/mysql-declare-handler/

DECLARE EXIT HANDLER FOR SQLEXCEPTION ROLLBACK;

fournit une action à réaliser sur une exception

5.9 Triggers

Voir https://www.mysqltutorial.org/mysql-triggers/

Tout comme les procédures stockées, les triggers servent à exécuter une ou plusieurs instructions. Mais à la différence des procédures, il n'est pas possible d'appeler un trigger : un trigger doit être déclenché par un événement.

Un trigger est **attaché à une table** et peut **être déclenché** par :

- une insertion dans la table (requête `INSERT`) ;
- la suppression d'une partie des données de la table (requête `DELETE`) ;
- la modification d'une partie des données de la table (requête `UPDATE`).

Par ailleurs, une fois le trigger déclenché, ses instructions peuvent être exécutées soit juste avant l'exécution de l'événement déclencheur, soit juste après.

```
1  -- Trigger déclenché par l'insertion
2  DELIMITER |
3  CREATE TRIGGER before_insert_animal BEFORE INSERT
4  ON Animal FOR EACH ROW
5  BEGIN
6      -- Instructions
7  END |
8
9  -- Trigger déclenché par la modification
10 CREATE TRIGGER before_update_animal BEFORE UPDATE
11 ON Animal FOR EACH ROW
12 BEGIN
13     -- Instructions
14 END |
15 DELIMITER ;
```

Section 6 Petite étude de cas

6.1 Contexte

Cet exercice est un exercice de conception de base de données.

Il est mis à titre indicatif mais est hors périmètre d'une formation d'initiation au langage SQL.

Il est susceptible d'être remplacé par un exercice plus orienté SQL.

6.2 Cahier des charges

On souhaite modéliser la gestion commerciale d'une entreprise qui vend des produits en démarchant auprès de ses clients (particuliers ou entreprises). L'entreprise possède différents services.

Il existe trois types d'employés :

- les membres de la direction (auxquels on affecte une zone de responsabilité),
- les assistantes commerciales
- et les commerciaux.

Les assistantes ont pour tâche principale de gérer l'emploi du temps d'un nombre prédéterminé de commerciaux (ces derniers sont en relation avec une seule assistante).

Les commerciaux sont les seuls à pouvoir réaliser une commande auprès d'un client.

Chaque employé est rattaché à un service et chaque service possède un responsable.

La France est divisée en secteurs et chaque client appartient à l'un d'entre eux. Un commercial a en charge un ou plusieurs secteurs.

Une commande concerne un client, est réalisée par un commercial et ne peut pas contenir deux lignes de commande qui impliquent le même produit. Une remise peut être effectuée sur une commande.

6.3 Etape 1 : le dictionnaire des données

Il faut prendre le temps de l'analyse du problème. Pas évident de commencer.

Pour aider à la réflexion, établir une liste, un dictionnaire, des données que doit gérer notre système, notre application.

On ne parle pas encore ici de table ni de modèle.

Proposition de présentation avec 2 exemples

Nom	Type	Taille	Commentaire
IdService	N		Identifiant du service
NomService	C	128	Nom du service

Avec Type : C chaîne de caractères, N numérique, D date, L logique, F flottant (décimal)

On peut avoir tendance à confondre données utiles à l'application et lien/relation entre les données.

Par exemple, pour la ligne du cahier des charges

« Chaque employé est rattaché à un service et chaque service possède un responsable. "

Il est fort probable qu'il y ait des données pour qualifier des employés, des données pour qualifier des services mais ne pas décrire ici "est rattaché" et " chaque service possède" car il s'agit de relations qui sont imaginées ensuite dans le diagramme.

6.4 Les différentes tables

A l'aide du dictionnaire établi, imaginer les différentes tables et colonnes nécessaires.

Faire en sorte que les mêmes données ne soient jamais définies à plusieurs exemplaires.

Dessiner sur papier les tables avec quelques données dedans

6.5 Les relations entre tables

Grâce aux dessins ci-dessus, faire un peu d'abstraction et imaginer les liens entre tables.

6.6 Proposition d'une solution

Le ficher etudeCas.sql propose une solution de tables.

Charger ce fichier. Nous allons mettre quelques entrées. Utiliser phpMyAdmin ou requête SQL …

1. créer le service Direction sans responsable
2. Créer l'employé Durand Daniel du service direction
3. Indiquer qu'au service direction Durand Daniel est le responsable
4. Idem pour le service commercial
5. Karl Azerty en est le responsable
6. Il est affecté au secteur « Pays de loire »
7. Créer le produit « compresseur » à 2500€
8. Créer le produit « riveteuse » à 500 €
9. Créer le client Ebasque Albert du secteur pays de Loire, adr rue du béret Perpignan
10. Il passe commande de 1 compresseur et de 3 riveteuses. Il n'a pas de remise

11. afficher le contenu de la commande de Mr Ebasque

Section 7 Problème de port 80 et dll manquantes

Au lancement de WampServer ou xampp , il est possible qu'un message d'erreur vous indique que le port 80 est déjà utilisé. Dans ce cas, vous devez changer le port que va utiliser le serveur Apache ou supprimer l'application qui utilise déjà ce port :

7.1 Identifier l'application qui utilise le port 80

Lancer une fenêtre Cmde

Taper la commande **netstat -a -n -o >result.txt** pour diriger le résultat dans le fichier result.txt

Ouvir le fichier dans un éditeur de texte et chercher la ligne contenant :80 . La dernière information est l'id du processus qui utilise ce port. Utiliser ensuite le gestionnaire de tâches pour identifier cet id. Arrêter le processus si permis.

Sinon vérifier que le port 1030 (exemple) n'est pas utilisé.

7.2 Modifier la configuration d'Apache

Ouvrir le fichier httpd.conf et remplacer le port 80 par 1030.

Relancer le serveur Apache.

7.3 Dll manquantes

L'utilisation de wampserver64 peut nécessiter des dll Visual Studio C++ : dll 110, 120 … manquantes.

Utiliser le fichier vcredist_x64_Allversions.zip fourni.

Arrêter wampserver.

Le dézipper et exécuter les différents installers dans la version croissante des no de version.