Université Nice Sophia Antipolis

H. Affes, J.Y. Tigli,
Polytech'Nice – Sophia
BAT4
2013-2014

# Travaux pratiques
# Utilisation MySql dans un programme C#

## 1 Rappel

Une clé primaire :

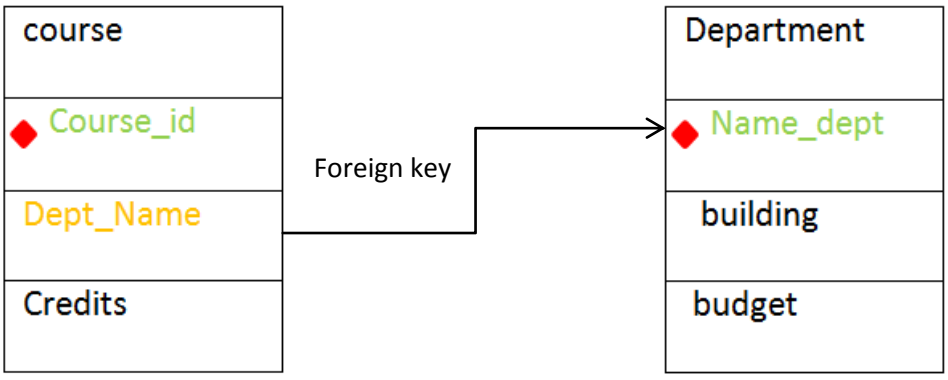
Une clé primaire (***Primary key PK***) est une contrainte d'unicité qui permet d'identifier de manière unique un enregistrement dans une table. Une clé primaire peut être composée d'un ou de plusieurs champs de la table. Deux lignes distinctes de la table ne peuvent pas avoir les mêmes valeurs pour les champs définis au niveau de la clé primaire. Il est possible de définir pour une même table plusieurs contraintes d'unicité, mais au plus une seule clé primaire. Une clé primaire est choisie parmi les clés candidates.

Une clé étrangère :

Une clé étrangère (***FOREIGN key PK***) identifie une colonne ou un ensemble de colonnes d'une table comme référençant une colonne ou un ensemble de colonnes d'une autre table (la table référencée). Les colonnes de la table référencée doivent faire partie d'une contrainte de **clé primaire** ou d'une **contrainte d'unicité** ***(Unique Index UQ)***.

## 2 Exemple :

Considérant 2 tables *cours* et *département* : Comme chaque cours appartient à un département spécifique. On définit cette relation par une clé étrangère

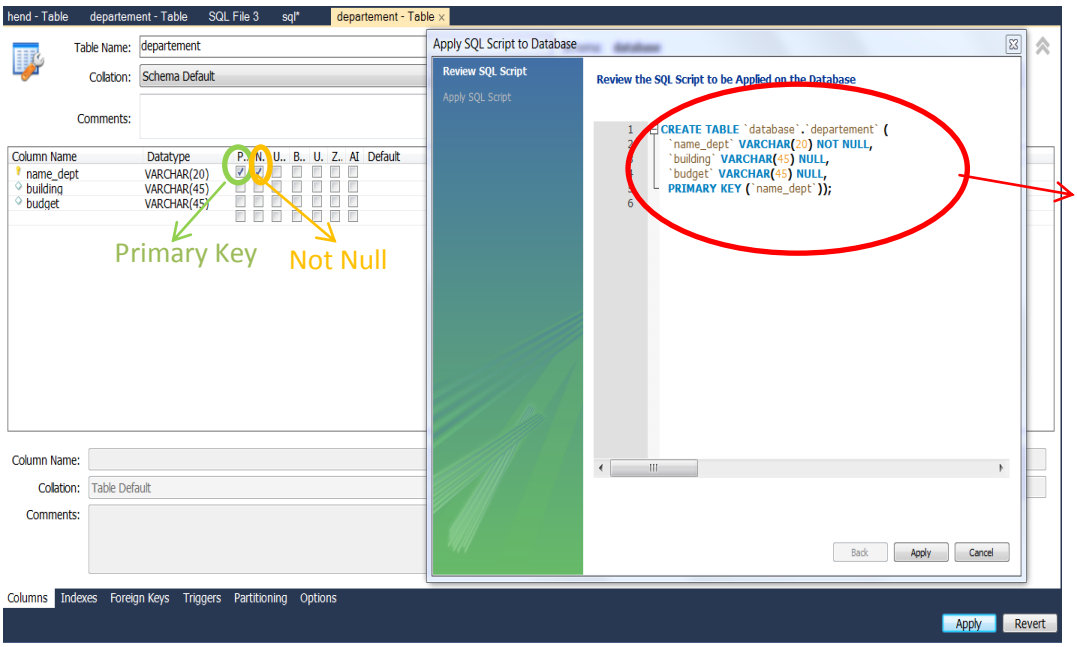



**Modèle physique de données**

La manpulation avec mysql est montrée dans les figures suivantes :

H. Affes, J.Y. Tigli,
Polytech'Nice – Sophia
BAT4
2013-2014

# Travaux pratiques
# Utilisation MySql dans un programme C#




Script SQL Associé à la création graphique d'une table

Figure 1 : Création de la table département

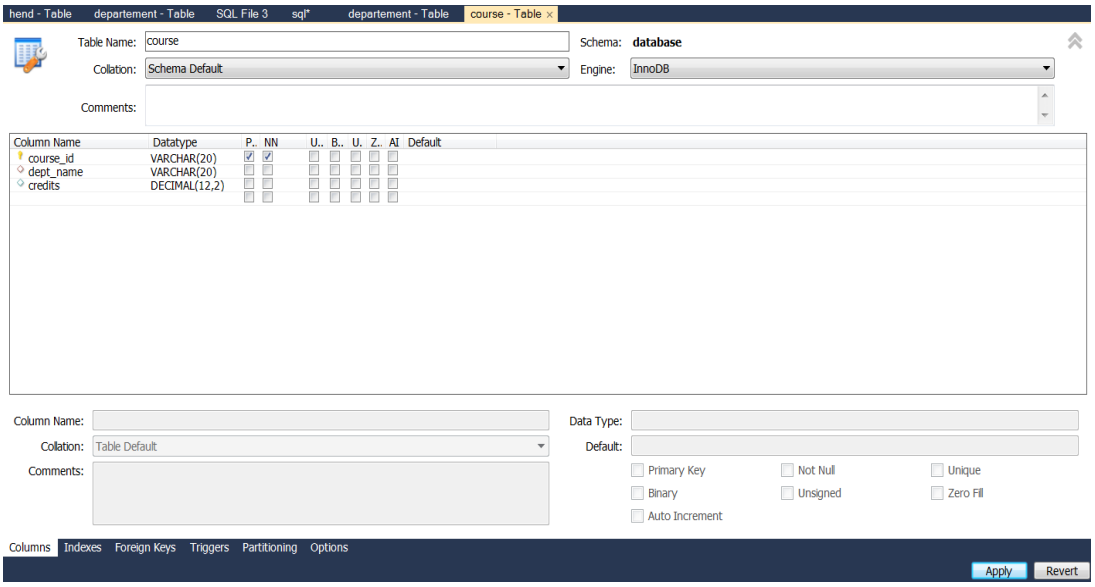



Figure 2: Création de la table course

# Travaux pratiques
# Utilisation MySql dans un programme C#

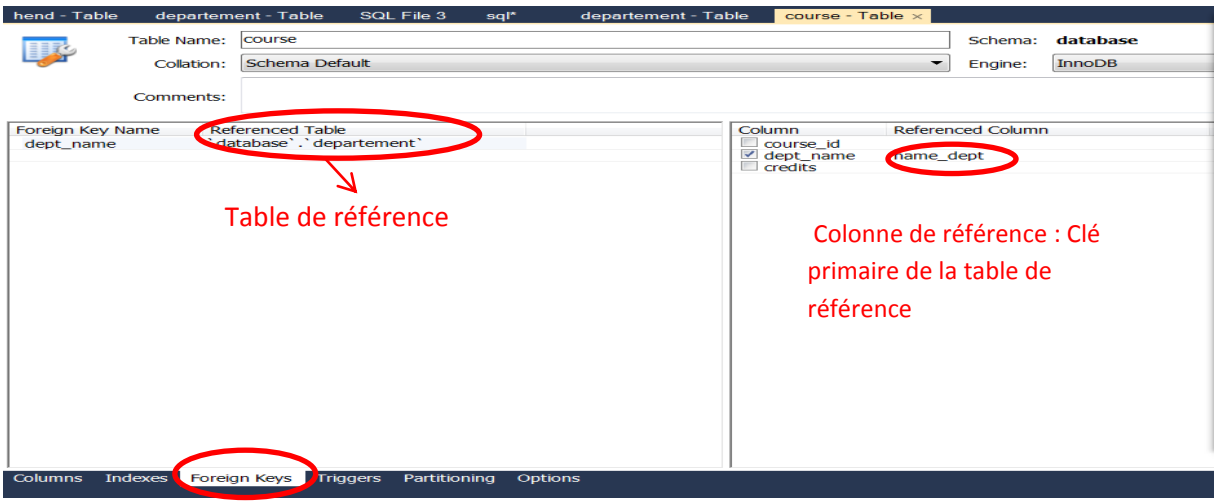



Figure 3: insertion d'une clé étrangère

Il faut que le type de la colonne de la clé étrangère et le type de la colonne de référence soient identiques.

# Travaux pratiques
# Utilisation MySql dans un programme C#

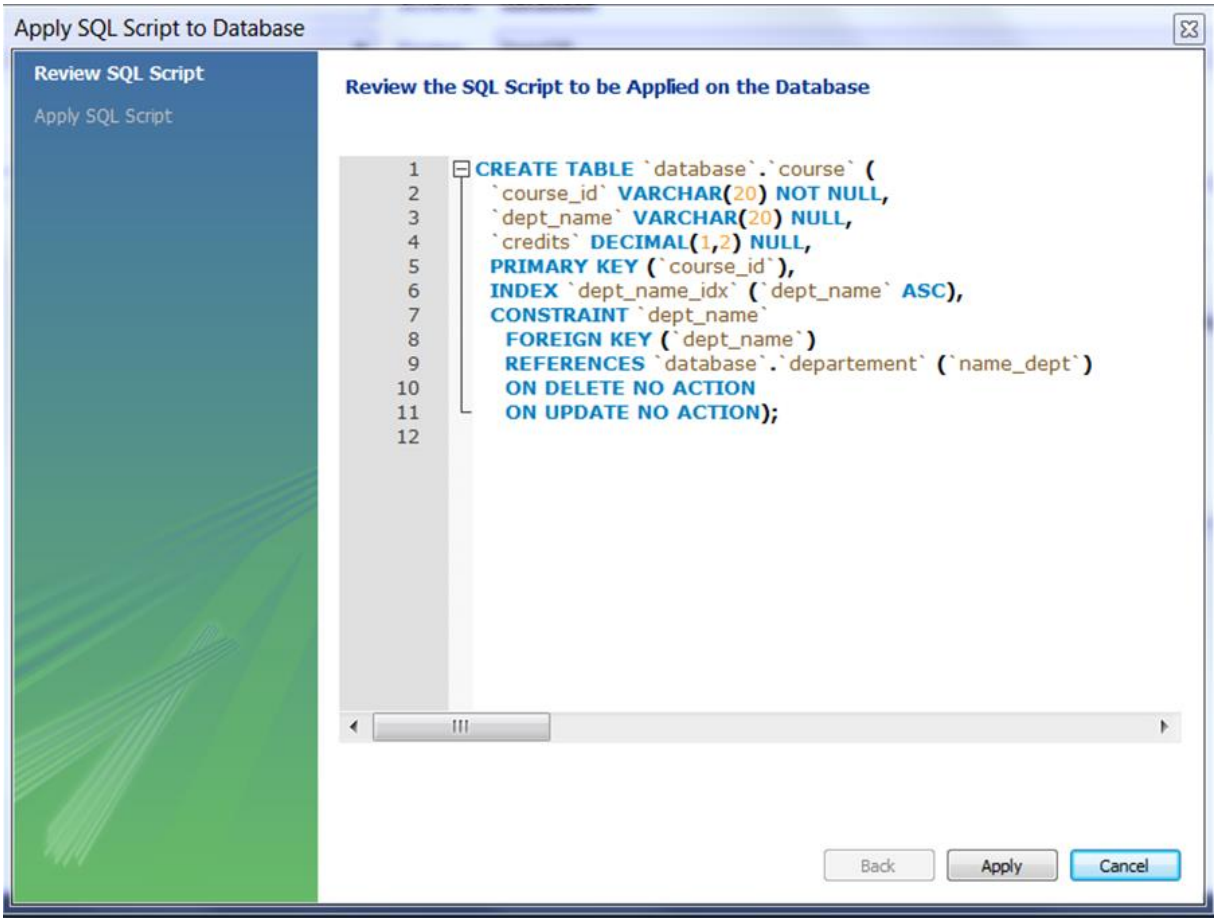



Figure 4: Script Sql associé à la creation du table course

## 3 Exercice 1 :

Créez la base de données relative aux travaux pratiques de synthèse Programmation Orientée Objet sous c#
https://www.tigli.fr/doku.php?id=cours:cours_environnement_de_programmation_bat4

Université Nice Sophia Antipolis

H. Affes, J.Y. Tigli,
Polytech'Nice – Sophia
BAT4
2013-2014

# Travaux pratiques
# Utilisation MySql dans un programme C#

## 4 Exercice 2 :

Pour connecter la base de données MySql en C#, il faut ajouter la référence de MySql comme il est illustré dans le figure 5 :

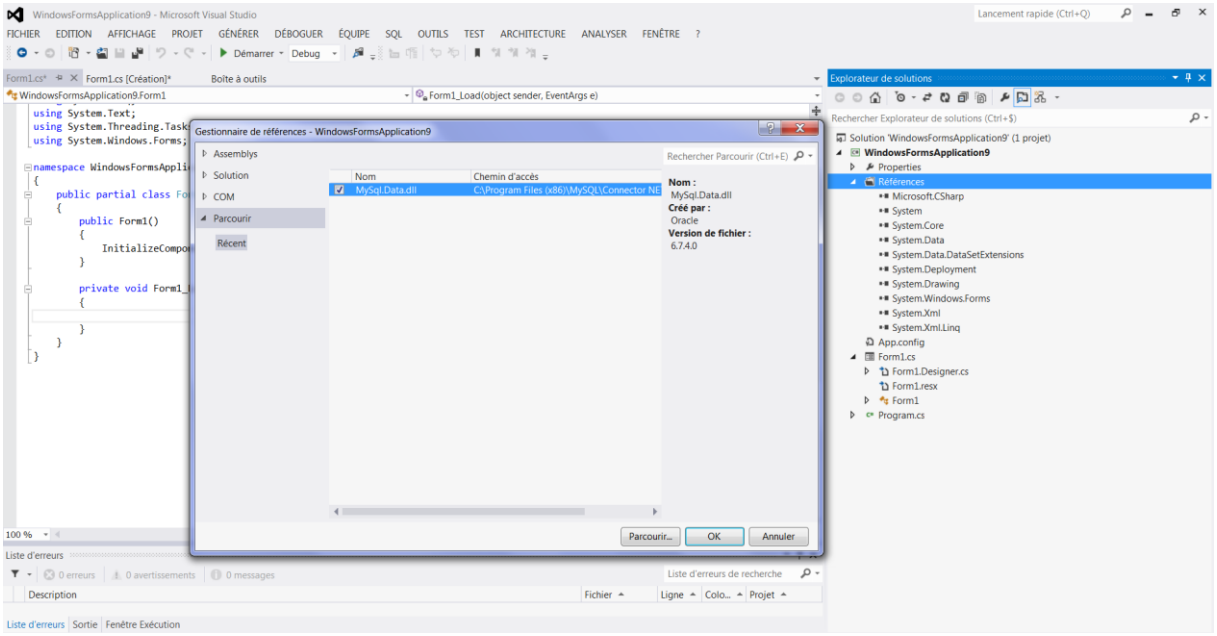



Figure 5: Ajout de la référence MySql.data

Ajouter la librairie **`MySql.Data.MySqlClient`** qui contient tout ce qu'il faut pour manipuler les bases de données MySql

`Les étapes pour ouvrir une connexion d'une base de données :`

La déclaration et l'initialisation des variables :

- `connection`: une instance de classe `MySqlConnection` utilisée pour ouvrir une connexion à la base de données
- `server`: indique où le serveur est hébergé, dans notre cas « localhost ».
- `database`: le nom de la base des données utilisée
- `uid`: MySQL username.
- `password`: MySQL password (que vous avez utilisé dans l'installation)
- `connectionString`: contient la chaine de caractère pour se connecter à la base de données

# Travaux pratiques
# Utilisation MySql dans un programme C#

```
class DBConnect
{
    private MySqlConnection connection;
    private string server;
    private string database;
    private string uid;
    private string password;

    //Constructor
    public DBConnect()
    {
        Initialize();
    }

    //Initialize values
    private void Initialize()
    {
        server = "localhost";
        database = "connectcsharptomysql";
        uid = "username";
        password = "password";
        string connectionString;
        connectionString = "SERVER=" + server + ";" + "DATABASE=" +
                database + ";" + "UID=" + uid + ";" + "PASSWORD=" + password + ";";

        connection = new MySqlConnection(connectionString);
    }
```

Figure 6: Code de la connexion à une base de données

Il faut toujours ouvrir une connexion avant d'interroger la (ou les) table(s) et fermer la tout de suite après la manipulation, pour libérer les ressources et indiquer que cette connexion n'est plus nécessaire.

# Travaux pratiques
# Utilisation MySql dans un programme C#

```
//open connection to database
private bool OpenConnection()
{
    try
    {
        connection.Open();
        return true;
    }
    catch (MySqlException ex)
    {
        //When handling errors, you can your application's response based
        //on the error number.
        //The two most common error numbers when connecting are as follows:
        //0: Cannot connect to server.
        //1045: Invalid user name and/or password.
        switch (ex.Number)
        {
            case 0:
                MessageBox.Show("Cannot connect to server.  Contact administrator");
                break;

            case 1045:
                MessageBox.Show("Invalid username/password, please try again");
                break;
        }
        return false;
    }
}

//Close connection
private bool CloseConnection()
{
    try
    {
        connection.Close();
        return true;
    }
    catch (MySqlException ex)
    {
        MessageBox.Show(ex.Message);
        return false;
    }
}
```

Figure 7: code d'ouverture et fermeture de la base de données

Généralement, `Insert`, `update` et `delete` sont utilisés pour écrire et changer dans la base de données, tant que `Select` est utilisé pour lire les données.

Pour cette raison, il existe différentes types des méthodes pour exécuter les différentes commandes

- `ExecuteNonQuery`: Utilisé pour exécuter une commande qui n'a pas une valeur de retour par exemple `Insert`, `update` or `delete`.
- `ExecuteReader:` Utilisé pour exécuter une commande qui a comme valeur de retour 0 ou plusieurs données, par exemple `Select.`
- `ExecuteScalar`: Utilisé pour exécuter une commande qui a comme une seule valeur par exemple `Select Count(*).`

# Travaux pratiques
# Utilisation MySql dans un programme C#

Pour exécuter une commande il faut faire ces étapes :

1. Ouvrir une connexion de la base de données
2. Création de la commande MySQL : une instance de `MySqlCommand`
3. Exécuter une commande en utilisant : `ExecuteNonQuery`
4. Fermer la connexion.

```
//Insert statement
public void Insert()
{
    string query = "INSERT INTO tableinfo (name, age) VALUES('John Smith', '33')";

    //open connection
    if (this.OpenConnection() == true)
    {
        //create command and assign the query and connection from the constructor
        MySqlCommand cmd = new MySqlCommand(query, connection);

        //Execute command
        cmd.ExecuteNonQuery();

        //close connection
        this.CloseConnection();
    }
}
```

Figure 8: exemple d'exécution une commande 'insert'

1) Créer des interfaces pour afficher et ajouter des nouveaux Fournisseur, Salarié, Livre et Auteur.
2) Créer une interface qui permet d'afficher pour chaque livre son auteur et son fournisseur

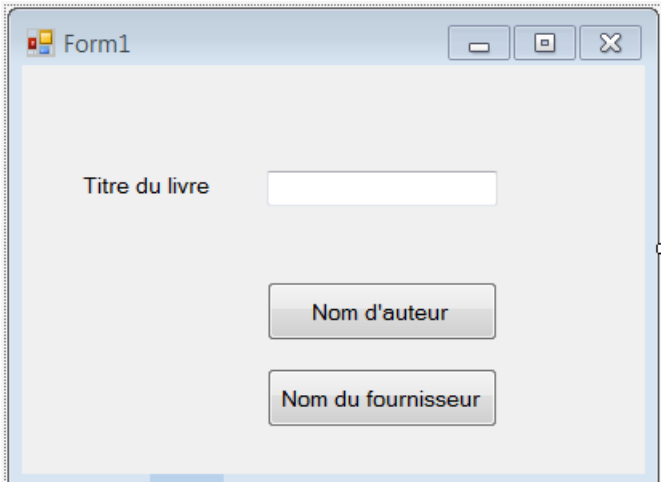



Figure 9: Exemple d'interface

H. Affes, J.Y. Tigli,
Polytech'Nice – Sophia
BAT4
2013-2014

# Travaux pratiques
# Utilisation MySql dans un programme C#

3) Une interface qui permet la mise à jour après la modification d'une table ainsi que la suppression d'une ligne d'une table