

Objectifs pédagogiques :

- Installer et configurer l'IDE afin de communiquer avec une BDD « MYSQL »
- Créer une interface graphique dynamique avec lecture et écriture de données depuis/vers une BDD

Sommaire de ce document :

I.	Création de la base de données.....	1
II.	Accès à la BDD.....	1
III.	Mise en œuvre d'un affichage graphique dans une fenêtre.....	1
IV.	Annexe 1 : Qt sous Windows configuration de l'accès Mysql :.....	5
V.	Annexe 2 : Qt sous Ubuntu configuration de l'accès Mysql :	2

I. Créer et Lancer une base de données graphiquement

1. Créer , sous les deux environnements linux et windows, Wamp (pour windows) Workbench (pour linux), une base de données contenant une table 'jeu' avec une clé primaire int, et deux colonnes prénom et nom.

2. Ajoutez deux personnes : Karim BENZEMA, et Antoine GRIEZMANN.

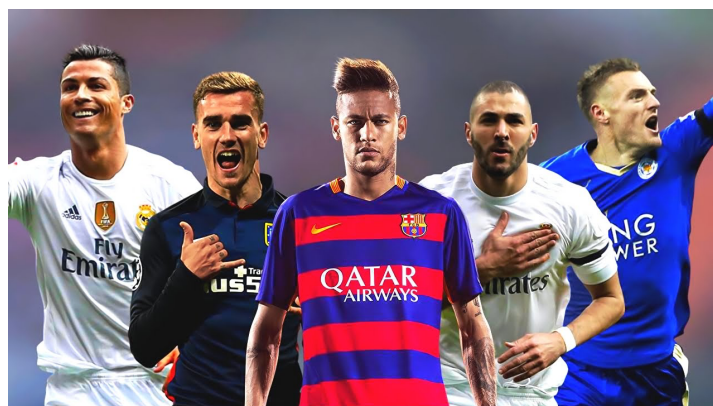
`INSERT INTO `jeu`(`id`, `Nom`, `club`) VALUES (1,'Ronaldo', 'Madrid'),(2,Grizou,'barca');`

II. Accès à la BDD

1. En vous aidant de l'annexe 1. Accédez à la BDD et affichez les données de la table jeu sous Linux.
2. En vous aidant de l'annexe 2. Accédez à la BDD et affichez les données de la table jeu sous Windows.

III. Mise en œuvre d'un affichage graphique des requêtes sur la BDD.

1. Présenter un Diagramme de classe UML en vous appuyant sur ceux des TP précédant qui vous permettra de créer une interface graphique de lecture et écriture dans la BDD afin d'afficher le club d'un joueur et son âge (Lecture BDD) mais aussi de saisir la note sur la prestation de ce joueur lors de son dernier match (Écriture BDD).
2. Créer l'interface graphique sur les deux environnements Linux puis Windows.



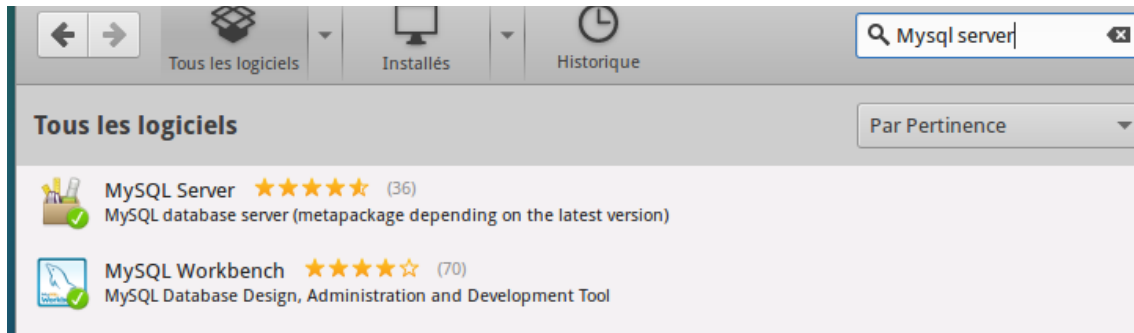
IV. Annexe 1 : Qt sous Ubuntu configuration de Qt et du driver Mysql :

Pour cette partie sous linux :

Dans un premier temps nous allons nous servir de la base de données que vous avez conçue dans les TP précédents

Dans un second temps :

*Vous pourrez gérer le serveur depuis un client **workbench** (ou autre gestionnaire de SGBD graphique). Si le serveur MySQL est déjà installé pas besoin de l'installer, sinon il faut l'installer.*



1. Installation de Qt sous linux :

Installer Qt via le lien sur le site officiel. <https://www.qt.io/download-qt-installer>

Faites seulement l'Installation de base !!!

2. Installation du driver MySQL :

Taper la commande suivante sous le terminal.

```
sudo apt-get update && sudo apt-get upgrade
```

Taper la commande suivante sous le terminal. Ceci installera les outils et bibliothèques nécessaires à faire fonctionner QT sous linux.

```
sudo apt-get -y install build-essential openssl libssl-dev libssl1.0 libgl1-mesa-dev  
libqt5x11extras5 '^libxcb.*-dev' libx11-xcb-dev libglu1-mesa-dev libxrender-dev libxi-dev  
libxkbcommon-dev libxkbcommon-x11-dev
```

Créer un projet en vous aidant des éléments suivants :**1 : requeteSQL.pro**

- ▼ requeteSQL
 - requeteSQL.pro
 - ▼ En-têtes
 - widget.h
 - ▼ Sources
 - main.cpp
 - widget.cpp
 - ▼ Formulaires
 - widget.ui

```

1  #-----
2  #
3  # Project created by QtCreator
4  #
5  #-----
6
7  QT      += core gui sql
8
9  greaterThan(QT_MAJOR_VERSION, 4): QT += widgets
10
11 TARGET = requeteSQL
12 TEMPLATE = app
13
14
15 SOURCES += main.cpp\
16 |         widget.cpp
17
18 HEADERS += widget.h
19
20 FORMS    += widget.ui
21

```

2 : Widget.h

- ▼ requeteSQL
 - requeteSQL.pro
 - ▼ En-têtes
 - widget.h
 - ▼ Sources
 - main.cpp
 - widget.cpp
 - ▼ Formulaires
 - widget.ui

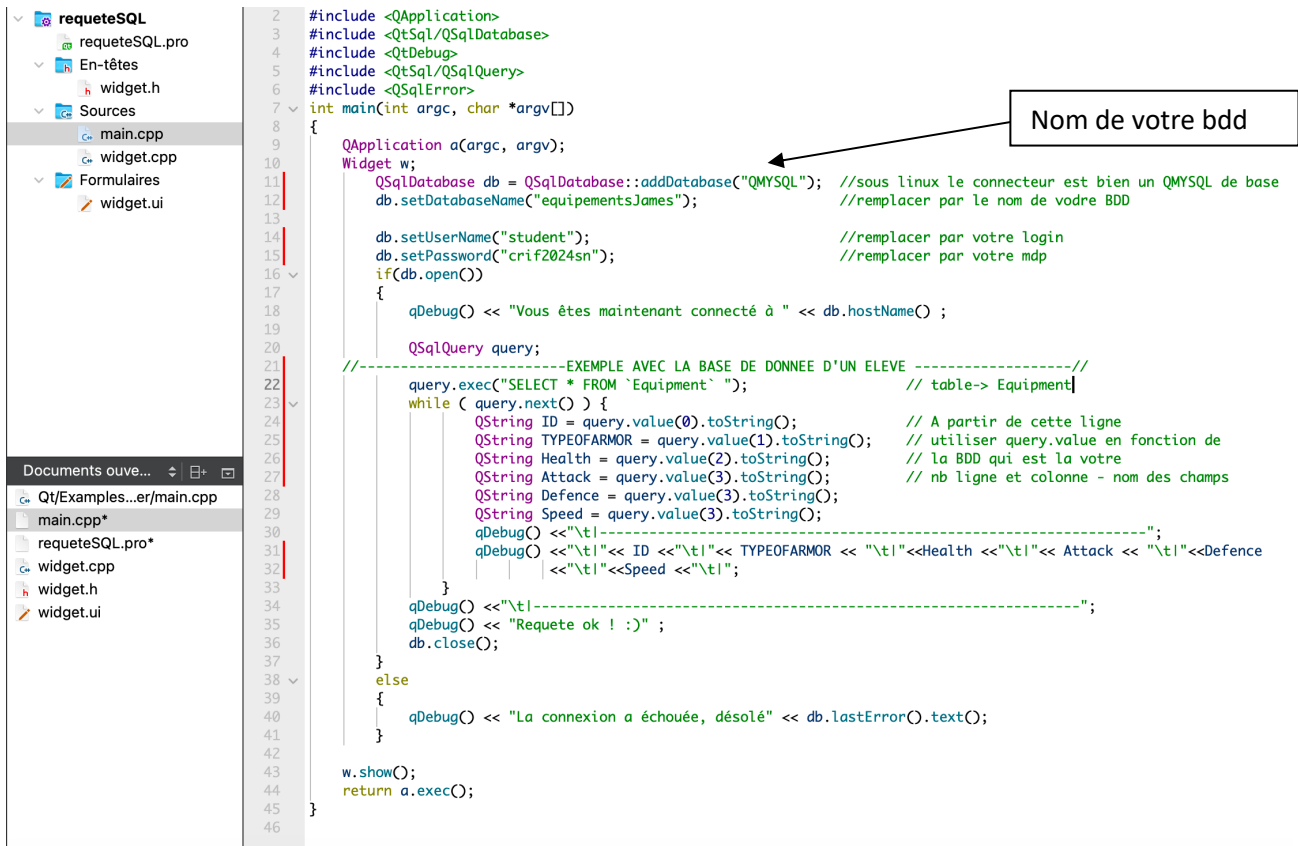
```

1  #ifndef WIDGET_H
2  #define WIDGET_H
3
4  #include <QWidget>
5
6  namespace Ui {
7  class Widget;
8  }
9
10 class Widget : public QWidget
11 {
12     Q_OBJECT
13
14 public:
15     explicit Widget(QWidget *parent = 0);
16     ~Widget();
17
18 private:
19     Ui::Widget *ui;
20 };
21
22 #endif // WIDGET_H
23

```

3 : main.cpp

3. Accéder à la base de données créée graphiquement avec le gestionnaire de SGBD (Workbench) installé à la partie II de la page 1 du TP.



```

1  requeteSQL
2  requeteSQL.pro
3  En-têtes
4  widget.h
5  Sources
6  main.cpp
7  widget.cpp
8  Formulaires
9  widget.ui
10
11 #include <QApplication>
12 #include <QtSql/QSqlDatabase>
13 #include <QtDebug>
14 #include <QtSql/QSqlQuery>
15 #include <QSqlError>
16 int main(int argc, char *argv[])
17 {
18     QApplication a(argc, argv);
19     Widget w;
20     QSqlDatabase db = QSqlDatabase::addDatabase("QMYSQL"); //sous linux le connecteur est bien un QMYSQL de base
21     db.setDatabaseName("equipementsJames"); //remplacer par le nom de votre BDD
22
23     db.setUserName("student"); //remplacer par votre login
24     db.setPassword("crif2024sn"); //remplacer par votre mdp
25     if(db.open())
26     {
27         qDebug() << "Vous êtes maintenant connecté à " << db.hostName() ;
28
29         QSqlQuery query;
30         //-----EXEMPLE AVEC LA BASE DE DONNEE D'UN ELEVE -----//
31         query.exec("SELECT * FROM `Equipment` "); // table-> Equipment
32         while ( query.next() ) {
33             QString ID = query.value(0).toString(); // A partir de cette ligne
34             QString TYPEOFARMOR = query.value(1).toString(); // utiliser query.value en fonction de
35             QString Health = query.value(2).toString(); // la BDD qui est la votre
36             QString Attack = query.value(3).toString(); // nb ligne et colonne - nom des champs
37             QString Defence = query.value(3).toString();
38             QString Speed = query.value(3).toString();
39             qDebug() << "\t|-----";
40             qDebug() << "\t|<< ID << "\t|<< TYPEOFARMOR << "\t|<< Health << "\t|<< Attack << "\t|<< Defence
41             << "\t|<< Speed << "\t|";
42         }
43         qDebug() << "\t|-----";
44         qDebug() << "Requete ok ! :) " ;
45         db.close();
46     }
47     else
48     {
49         qDebug() << "La connexion a échouée, désolé" << db.lastError().text();
50     }
51
52     w.show();
53     return a.exec();
54 }

```

Nom de votre bdd

V. Annexe 2 : Qt sous Windows configuration de l'accès Mysql :

1. Installation du connecteur Mysql / ODBC pour convertir les données ODBC vers Mysql

Pour faire fonctionner le lien avec la base de données il faut Installer le connecteur ODBC:

<http://dev.mysql.com/downloads/connector/odbc/>

mysql-connector-odbc-5.3.4-win32.msi

2. Connexion grâce au driver QODBC3 Tests des drivers :

Voici un autre exemple un peu plus laconique mais très ressemblant de celui sous linux de l'annexe 1.

Nb. Dans le fichier .pro de votre projet, ajouter : QT += core gui sql

Accès aux données :

```
#include "mainwindow.h"
#include <QApplication>
#include <QSqlDatabase>
#include <QtDebug>
#include <QSqlQuery>

int main(int argc, char *argv[])
{
    QApplication a(argc, argv);
    MainWindow w;

    QSqlDatabase db = QSqlDatabase::addDatabase("QODBC3");


    db.setDatabaseName("Driver={MySQL ODBC 5.3 ANSI Driver};DATABASE=test;");
    db.setUserName("root");
    if (db.open())
    {
        qDebug() << "Vous êtes maintenant connecté à " << db.hostName() ;

        QSqlQuery query;

        query.exec("SELECT * FROM `jeu`");
        while ( query.next() ) {
            QString prenom = query.value(1).toString();
            QString club = query.value(2).toString();
            qDebug() << prenom << club;
        }
        qDebug() << "Requete ok ! :)" ;
        db.close();
    }
    else
    {
        qDebug() << "La connexion a échouée, désolé" ;
    }
    w.show();

    return a.exec();
}
```

Nom de votre bdd



Résultat Attendu :

Vous êtes maintenant connecté à ""
 "Ronaldo" "Madrid"
 "Grizou" "Barça"

Requête ok ! :)



Ajout de driver pour problème de non-détection et de chargement du driver ODBC :

<https://youtu.be/qVdhPUI1Fio?si=u2CGOEUyMxKlewC>