SQLite et Android

SQLite

SQLite est un moteur de bases de données libre qui implémente la plus part des fonctionnalités du SQL92(standard). SQLite utilise donc la plus part des commandes SQL (CREATE, INSERT, UPDATE, DELETE, et SELECT).

SQLite ne nécessite pas de serveur de bases de données pour fonctionner. Vous pouvez télécharger la dernière version de SQLite à https://www.sqlite.org/download.html

Types de données : SQLite n'accepte que 5 types de données :

Type	**Définition**
NULL	Valeur vide
INTEGER	Entier signé
REAL	Nombre réel
TEXT VARCHAR	Champ texte
BLOB	Champ binaire (image)

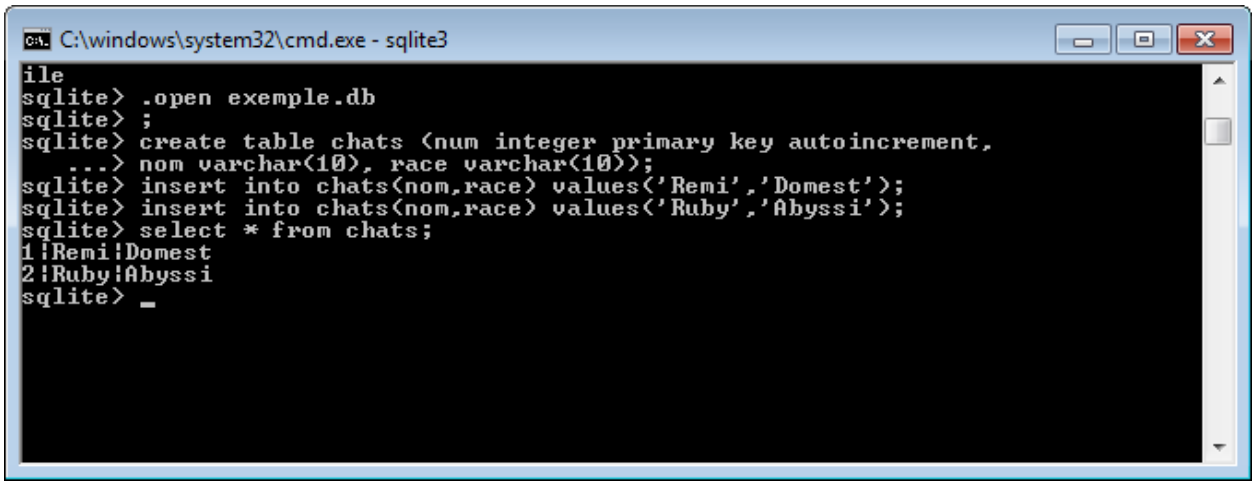
Vous pouvez utiliser SQLite pour vos projets qui utilisent une base de données locale. Il suffit de télécharger la dll SQLite3 . La création de la BD pourra se faire par l'invite commandes ou en téléchargeant (et installant) l'outil SQLiteBrowser. Les accès à la BD SQlite peuvent se faire par JAVA, C# et Android etc…

Pour utiliser SQLite avec Android, vous n'avez pas besoins de télécharger et d'exécuter quoi que ce soit puisque ce moteur de bases de données vient avec Android. Il vous suffit d'utiliser les classes nécessaires d'Android pour créer et manipuler une base de données SQLite.

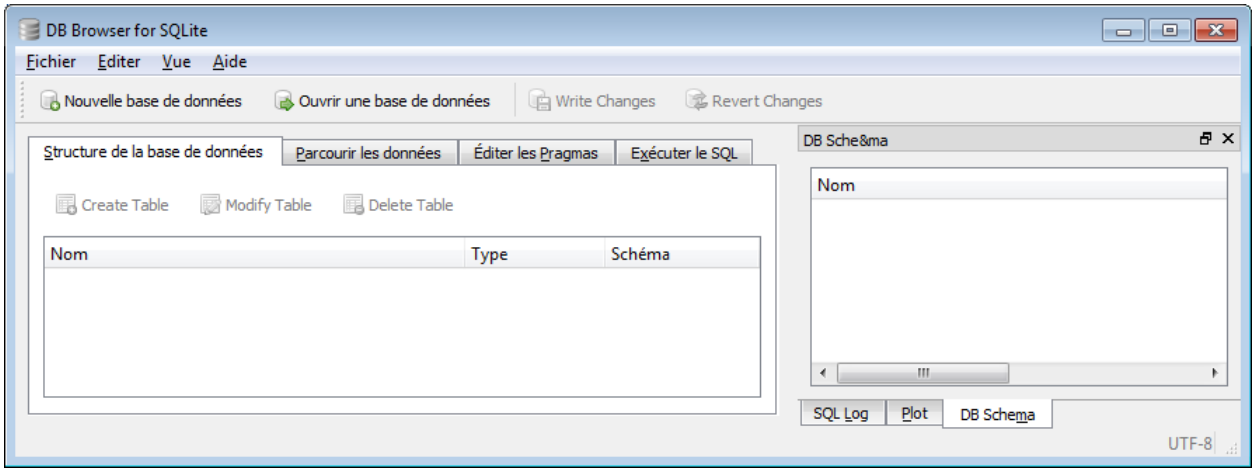
Pour la version pc, vous pouvez créer la BD de deux façons :

Par l'invite de commande

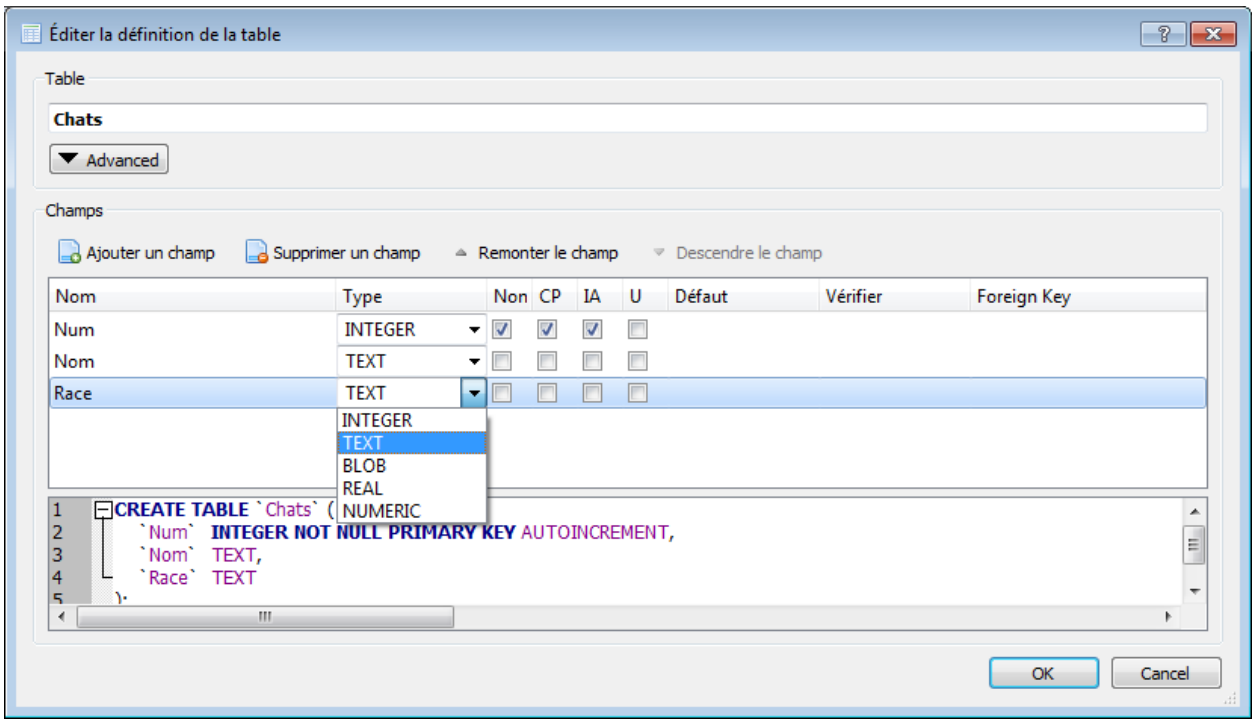
- open Nombd ouvre une base de données existante ou créer une base de donnée si celle-ci n'existe pas. (Remarquez le point devant open).

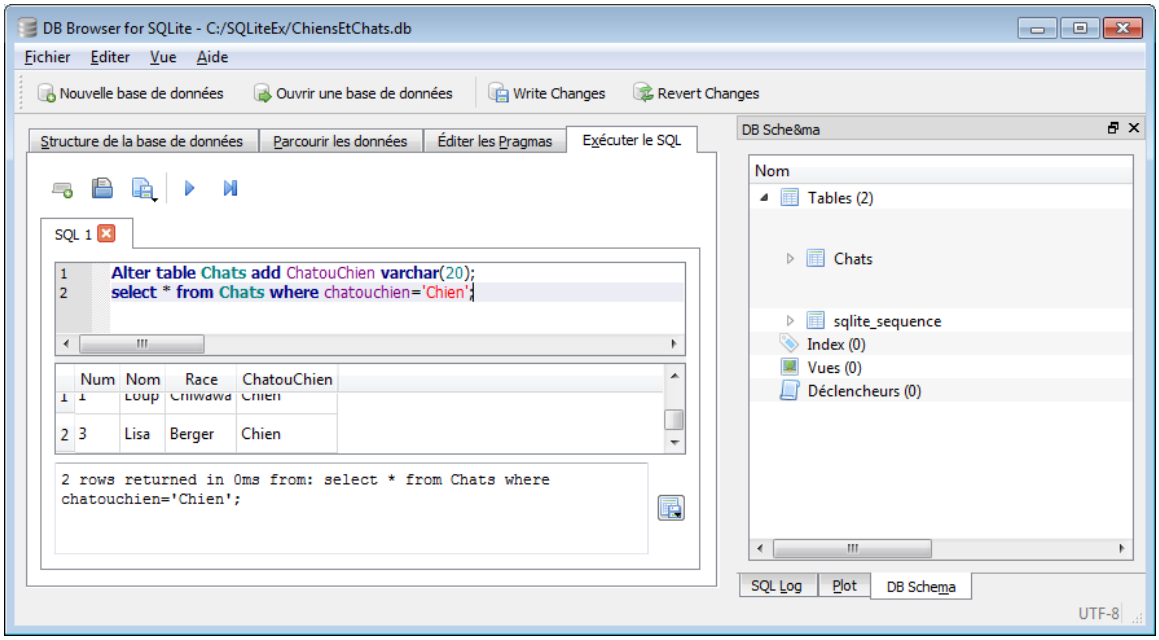


Vous pouvez, encore plus facile, télécharger et installer SqliteBrowser à l'adresse http://sqlitebrowser.org/ pour créer et gérer vos base de données SQLite.



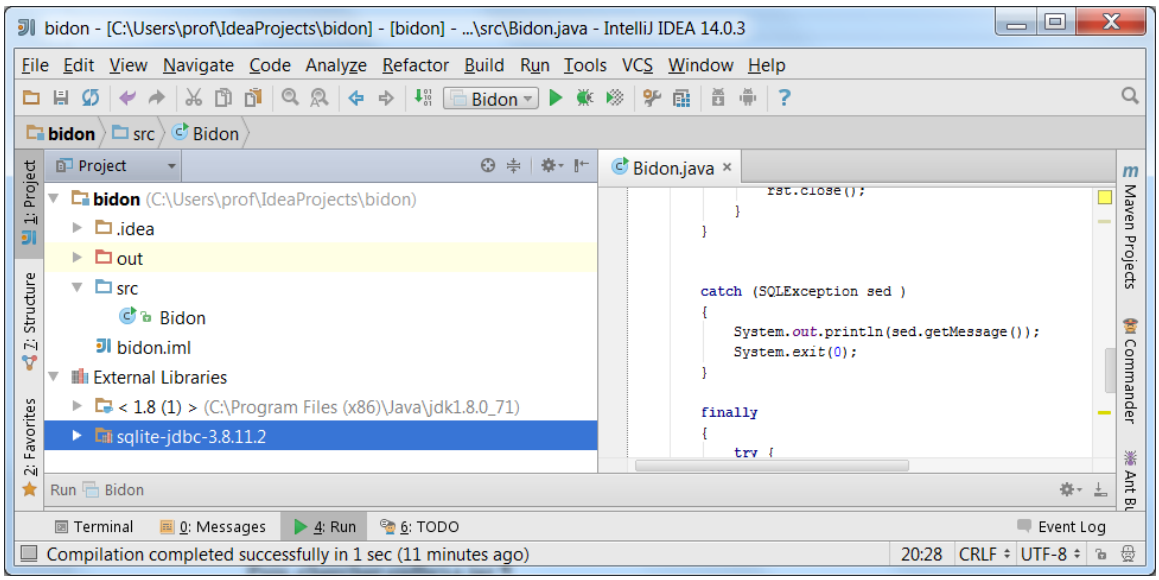
Par l'onglet Nouvelle base de données, vous pouvez créer votre base de données et créer et gérer vos tables.


Éditer la définition de la table
Table
Chats
Advanced
Champs
Ajouter un champ
Supprimer un champ
Remonter le champ
Descendre le champ
Nom
Type
Non
CP
IA
U
Défaut
Vérifier
Foreign Key
Num
INTEGER
Nom
TEXT
Race
TEXT
INTEGER
TEXT
BLOB
REAL
NUMERIC
CREATE TABLE `Chats` (
`Num` INTEGER NOT NULL PRIMARY KEY AUTOINCREMENT,
`Nom` TEXT,
`Race` TEXT
);
OK
Cancel


DB Browser for SQLite - C:/SQLiteEx/ChiensEtChats.db
Fichier
Editer
Vue
Aide
Nouvelle base de données
Ouvrir une base de données
Write Changes
Revert Changes
Structure de la base de données
Parcourir les données
Éditer les Pragmas
Exécuter le SQL
SQL 1
Alter table Chats add ChatouChien varchar(20);
select * from Chats where chatouchien='Chien';
Num
Nom
Race
ChatouChien
2
3
Lisa
Berger
Chien
2 rows returned in 0ms from: select * from Chats where chatouchien='Chien';
DB Schema
Nom
Tables (2)
Chats
sqlite_sequence
Index (0)
Vues (0)
Déclencheurs (0)
SQL Log
Plot
DB Schema
UTF-8

JDBC et SQlite : Voir le cours «Jdbc en Bref»

Par le menu Fichier, Project Structure, à l'onglet Libraire, Ajouter



SQLite sur Android :

Deux méthodes populaires pour accéder et exploiter à une base de données SQLite avec Android. Une qui utilise surtout la classe SQLiteDatabase (proche de la bd) et celle qui utilise la classe SQLiteOpenHelper et SQLiteDatabase (proche de la prog)

La méthode qui sera détaillée ici est celle qui utilise la classe SQLiteDatabase.

La classe SQLiteDatabase a des méthodes qui permettent de faire une gestion complète d'une base de données SQLite sur appareils mobiles Android.

Voici quelques méthodes de cette classe.

Méthodes	Rôle
execSQL(String sql, Object[] bindArgs)	Execute une instruction SQL qui n'est pas un SELECT
rawQuery(String sql, String[] selectionArgs)	Execute une requête SQL de type SELECT et retourne un ensemble de résultat dans un curseur : CURSOR
openOrCreateDatabase(String name, int mode, CursorFactory factory)	Retourne un SQLiteDatabe. Les paramètres sont : nom pour la base de données. Il Peut être null. Le Context Mode, ici (Context. MODE_PRIVATE) qui veut dire que seule votre activité a droit d'y accéder. C'est le mode par défaut

Les curseurs sont des variables (zone mémoire) utilisées pour récupérer le résultat de requêtes de type SELECT

Les curseurs sont des objets qui contiennent les résultats d'une recherche dans une base de données. Ce sont en fait des objets qui fonctionnent comme les tableaux .Ils contiennent les colonnes et lignes qui ont été renvoyées par la requête. On y accède ligne par ligne puis colonne par colonne.

Un objet de type Curseur possède un pointeur sur l'enregistrement courant. À la réception de cet objet, le pointeur se trouve devant le premier enregistrement.

Voici quelques méthodes de l'objet Curseur :

Pour vous déplacer entre les enregistrements, vous pouvez utiliser les méthodes

moveToFirst () et **moveToNext ().** (MoveToPrevious() et moveToLast() pour une lecture inverse)

La méthode **isAfterLast()** permet de vérifier si la fin du résultat de la requête a été atteint.

getCount() permet d'obtenir le nombre d'enregistrements de la requête.

Cursor fournit les méthodes get * (), par exemple getLong (columnIndex), getString (columnIndex) pour accéder aux données de l'enregistrement courant. Le "columnIndex" est l'index de la colonne à accéder.

Cursor fournit également la méthode getColumnIndexOrThrow (String) qui permet d'obtenir l'index de colonne pour un nom de colonne de la table.

Un curseur doit être fermé avec la méthode **close().**

Exemple d'application utilisant SQLite et Android

Contenu du fichier activity_main.xml

```
<?xml version="1.0" encoding="utf-8"?>
<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
   android:id="@+id/myLayout"
   android:stretchColumns="0"
   android:layout_width="fill_parent"
   android:layout_height="fill_parent">
```

```
<TableLayout
    android:layout_width="fill_parent"
    android:layout_height="fill_parent">

    <TableRow
      android:layout_width="fill_parent"
      android:layout_height="fill_parent">
      <TextView android:text="Gestion des joueurs"
        android:layout_column="1"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        />
    </TableRow>

  <TableRow
    android:layout_width="fill_parent"
    android:layout_height="fill_parent">
    <TextView android:text="@string/Num"
      android:layout_column="1"
      android:layout_width="wrap_content"
      android:layout_height="wrap_content"
      />
    <EditText android:id="@+id/edtNum"
      android:inputType="number"
      android:layout_column="2"
      android:layout_width="150dp"
      android:layout_height="40dp"
      />
  </TableRow>
```

```
<TableRow
   android:layout_width="fill_parent"
   android:layout_height="fill_parent">
   <TextView android:text="@string/nom"
      android:layout_column="1"
      android:layout_width="wrap_content"
      android:layout_height="wrap_content"
      />

   <EditText android:id="@+id/edtNom"
      android:inputType="text"
      android:layout_column="2"
      android:layout_width="150dp"
      android:layout_height="40dp"
      />
</TableRow>



<TableRow
   android:layout_width="fill_parent"
   android:layout_height="fill_parent">
   <TextView android:text="@string/prenom"
      android:layout_column="1"
      android:layout_width="wrap_content"
      android:layout_height="wrap_content"
      />
   <EditText android:id="@+id/edtPrenom"
      android:inputType="text"
```

```
        android:layout_column="2"
        android:layout_width="150dp"
        android:layout_height="40dp"
        />
    </TableRow>

    <TableRow
      android:layout_width="fill_parent"
      android:layout_height="fill_parent">
      <Button       android:id="@+id/btnAjouter"
        android:text="@string/ajouter"
        android:layout_column="1"
        android:layout_width="100dp"
        android:layout_height="40dp"
        />
      <Button       android:id="@+id/btnSupprimer"
        android:text="@string/supprimer"
        android:layout_column="2"
        android:layout_width="100dp"
        android:layout_height="40dp"
        android:layout_centerVertical="true"

        />
    </TableRow>

    <TableRow
      android:layout_width="fill_parent"
      android:layout_height="fill_parent">
      <Button   android:id="@+id/btnAfficher"
```

```
        android:text="@string/afficher"
        android:layout_column="1"
        android:layout_width="100dp"
        android:layout_height="40dp"
        />
    <Button   android:id="@+id/btnAffichertout"
        android:text="@string/affichertout"
        android:layout_width="100dp"
        android:layout_height="40dp"
        android:layout_column="2" />
</TableRow>
    <TableRow
    android:layout_width="fill_parent"
    android:layout_height="fill_parent">
    <Button   android:id="@+id/btnSuivant"
        android:text="@string/suivant"
        android:layout_column="1"
        android:layout_width="100dp"
        android:layout_height="40dp"
        />
    <Button   android:id="@+id/btnPrcedent"
        android:text="@string/precedent"
        android:layout_width="100dp"
        android:layout_height="40dp"
        android:layout_column="2" />
</TableRow>
```

```
  <TableRow
     android:layout_width="fill_parent"
     android:layout_height="fill_parent">
  <Button   android:id="@+id/btnPremier"
     android:text="@string/premier"
     android:layout_column="1"
     android:layout_width="100dp"
     android:layout_height="40dp"
     />
  </TableRow>
  </TableLayout>
</RelativeLayout>
```

Contenu du fichier strings.xml

```
<resources>
   <string name="app_name">SqliteExemple</string>
   <string name="Num">Numéro: </string>
   <string name="nom">Nom: </string>
   <string name="prenom">Prenom: </string>
   <string name ="ajouter">  Ajouter</string>
   <string name ="supprimer">  Supprimer</string>
   <string name ="afficher">  Afficher</string>
   <string name ="affichertout">  ToutAfficher</string>
<string name ="suivant">  Suivant</string>
   <string name ="precedent">  Precedent</string>
   <string name ="premier">  Premier</string>
</resources>
```

Contenu MainActivity.java

```
package com.example.salihayacoub.sqliteexemple;
import android.app.Activity;
import android.app.AlertDialog.Builder;
import android.content.Context;
import android.database.Cursor;
import android.database.sqlite.SQLiteDatabase;
import android.os.Bundle;
import android.view.View;
import android.view.View.OnClickListener;
import android.widget.Button;
import android.widget.EditText;

public class MainActivity extends Activity implements OnClickListener {
//Déclarations
   EditText edtNom, edtPrenom, edtNum;
   Button btnAfficher, btnSupprimer, btnAjouter, btnAffichertout;
   Button btnSuivant, btnPrcedent, btnPremier;
   SQLiteDatabase bd;
     Cursor c1;  // utilisé par les boutons suivant et précédent

   @Override
   public void onCreate(Bundle savedInstanceState) {
      super.onCreate(savedInstanceState);
      setContentView(R.layout.activity_main);
```

```
        edtNom = (EditText) findViewById(R.id.edtNom);
        edtPrenom = (EditText) findViewById(R.id.edtPrenom);
        edtNum = (EditText) findViewById(R.id.edtNum);

       btnAjouter = (Button) findViewById(R.id.btnAjouter);
        btnSupprimer = (Button) findViewById(R.id.btnSupprimer);
        btnAfficher = (Button) findViewById(R.id.btnAfficher);
        btnAffichertout = (Button) findViewById(R.id.btnAffichertout);
       btnSuivant =(Button)findViewById(R.id.btnSuivant);
         btnPrecdent =(Button)findViewById(R.id.btnPrcedent);
         btnPremier =(Button)findViewById(R.id.btnPremier);

        btnAjouter.setOnClickListener(this);
        btnSupprimer.setOnClickListener(this);
        btnAffichertout.setOnClickListener(this);
        btnSupprimer.setOnClickListener(this);
        btnAfficher.setOnClickListener(this);
       btnSuivant.setOnClickListener(this);
        btnPrecdent.setOnClickListener(this);
        btnPremier.setOnClickListener(this);

        bd = openOrCreateDatabase("GestionJoueurs", Context.MODE_PRIVATE,null);
        bd.execSQL("CREATE TABLE IF NOT EXISTS Joueurs(num integer primary key autoincrement,
nom VARCHAR,prenom VARCHAR);");
    }
    public void onClick(View view)
    {
```

```
if(view==btnAjouter)
    {
   if(edtNom.getText().toString().trim().length()==0||
         edtPrenom.getText().toString().trim().length()==0)
   {
      AfficheMessage("Erreur", "Entrer toutes les valeurs");
      return;
   }
   bd.execSQL("INSERT INTO joueurs(nom,prenom) VALUES('" + edtNom.getText() +
         "','" + edtPrenom.getText()  + "');");
   AfficheMessage("Succès", "Information ajouter");
   videTexte();
}// fin bouton ajouter
//----------DébutAfficher TOUT
if(view==btnAffichertout)
{
   Cursor c=bd.rawQuery("SELECT * FROM Joueurs",null);
   if(c.getCount()==0)
   {
      AfficheMessage("Erreur", "Aucune donnée");
      return;
   }
   StringBuffer buffer=new StringBuffer();
   while(c.moveToNext())
   {
      buffer.append("Numéro: "+c.getInt(0)+"\n");
      buffer.append("Non: "+c.getString(1)+"\n");
      buffer.append("Prénom: "+c.getString(2)+"\n");
   }
```

```
        c.close();
        AfficheMessage("Les joueurs", buffer.toString());
    }
    //----fin afficher tout


//Bouton supprimer
    if(view==btnSupprimer)
    {
        if(edtNum.getText().toString().trim().length()==0)
        {
            AfficheMessage("Erreur", "Entrer un numéro de joueur");
            return;
        }
        Cursor c=bd.rawQuery("SELECT * FROM Joueurs WHERE num="+ edtNum.getText(), null);
        if(c.moveToFirst())
        {
            bd.execSQL("DELETE FROM Joueurs WHERE num=" + edtNum.getText() );
            AfficheMessage("Succès", "Information détruite");
        }
        else
        {
            AfficheMessage("Erreur", "Numéro invalide");
        }
        videTexte();
    } // fin supprimer
```

```
// Bouton afficher le premier
            if(view==btnPremier) {
          try{
          c1 = bd.rawQuery("SELECT * FROM Joueurs", null);
          if (c1.getCount() == 0) {
             AfficheMessage("Erreur", "Aucune donnée");
             return;
         }
       if(c1.moveToFirst())
            {
        edtNom.setText(c1.getString(1));
        dtPrenom.setText(c1.getString(2));
            }
       } // fin du try
     catch(Exception se){
      Toast.makeText(MainActivity.this, "message"+ se.getMessage().toString(),
Toast.LENGTH_LONG).show();
                    }
  } // Fin du premier

// bouton suivant :
       if(view ==btnSuivant)
      {
         if(c1.moveToNext())
         edtNom.setText(c1.getString(1));
         edtPrenom.setText(c1.getString(2));
      }
```

```
// bouton précedent
   if(view ==btnPrecdent)
      {
         if(c1.moveToPrevious())
            edtNom.setText(c1.getString(1));
         edtPrenom.setText(c1.getString(2));
      } // fin précedent

//Bouton afficher un Enregistrement
if(view==btnAfficher) {
         Cursor c = bd.rawQuery("SELECT * FROM Joueurs where num= "+ edtNum.getText(),
null);
         if (c.getCount() == 0) {
            AfficheMessage("Erreur", "Aucune donnée");
            return;
         }
         if(c.moveToFirst())
         {
            edtNom.setText(c.getString(1));
            edtPrenom.setText(c.getString(2));
         }
     }// fin Afficher

   }//Fin du onClick
```

```
  public void AfficheMessage(String titre,String message)
  {
     Builder builder=new Builder(this);
     builder.setCancelable(true);
     builder.setTitle(titre);
     builder.setMessage(message);
     builder.show();
  }

  public void videTexte()
  {
     edtNum.setText("");
     edtNom.setText("");
     edtPrenom.setText("");
  }
}//fin
```

Sources :

https://www.sqlite.org/quickstart.html

Notes de cours de Denis Brunet.

http://developer.android.com/reference/android/database/sqlite/SQLiteDatabase.html

http://developer.android.com/reference/android/content/Context.html

https://openclassrooms.com/courses/creez-des-applications-pour-android

http://www.univ-orleans.fr/lifo/Members/Jean-Francois.Lalande/enseignement/android/cours-android.pdf

Travail final : Partie 2

Pondération 8%

Date de remise le 19 mai avant 16h10

1- Directement sur votre téléphone/tablette, créer la table circuits suivantes en utilisant SQLite

Colonnes	Types et contrainte
IdCircuit	integer, primary key. autoincrement
VilleDepart	Varchar
VilleArivee	Varchar
Prix	REAL
Duree	integer

2- Créer l'application JDBC-Android qui permet :
   a. Créer la base de données
   b. D'ajouter un enregistrement
   c. De supprimer un enregistrement
   d. Rechercher un circuit selon la ville de départ. S'il y a plusieurs circuits on doit pouvoir se déplacer d'un enregistrement à l'autre par des boutons suivant et précedent.

Important : Le cours sur SQLite et Android sera présenté le lundi 09 mai.