# Programación Interactiva JDBC

Escuela de Ingeniería de Sistemas y Computación Facultad de Ingeniería Universidad del Valle



#### Motivación

- JDBC fue diseñado para mantener sencillas las cosas sencillas. Esto significa que el API JDBC hace muy sencillas las tareas diarias de una base de datos, como una simple sentencia **SELECT**.
- Esta sección nos llevará a través de ejemplos que utilizan el JDBC para ejecutar sentencias SQL comunes, para que podamos ver lo sencilla que es la utilización del API JDBC básico.

#### **Pasos**

- Establecer la conexion
  - Cargar Drivers
    - » Class.forName("sun.jdbc.odbc.JdbcOdbcDriver");
  - Hacer la conexión
    - » Connection con = DriverManager.getConnection(url, "Log", "Pass");
    - » String url = "jdbc:odbc:Fred"; Connection con = DriverManager.getConnection(url, "Fernando", "J8");

#### Crear Tablas

#### Crear Tablas

Statement stmt = con.createStatement();

```
String createTableCoffees = "CREATE TABLE COFFEES" +
"(COF_NAME VARCHAR(32), SUP_ID INTEGER, " +
" PRICE FLOAT, SALES INTEGER, TOTAL INTEGER)";
stmt.executeUpdate(createTableCoffees);
```

```
stmt.executeUpdate("CREATE TABLE COFFEES " +
"(COF_NAME VARCHAR(32), SUP_ID INTEGER, " +
" PRICE FLOAT, SALES INTEGER, TOTAL INTEGER)");
```

#### Insertar Valores a las Tablas

```
Statement stmt = con.createStatement();
stmt.executeUpdate("INSERT INTO COFFEES " +
  "VALUES ('Espresso', 150, 9.99, 0, 0)");
stmt.executeUpdate("INSERT INTO COFFEES " +
  "VALUES ('Colombian_Decaf', 101, 8.99, 0, 0)");
stmt.executeUpdate("INSERT INTO COFFEES " +
  "VALUES ('French_Roast_Decaf', 49, 9.99, 0, 0)");
```

#### Seleccionar Valores a las Tablas

stmt. executeQuery ("SELECT \* FROM COFFEES");

stmt. executeQuery ("SELECT COF\_NAME, PRICE" + "FROM COFFEES WHERE PRICE < 9.00 ");

String query = "SELECT COF\_NAME, PRICE " + "FROM COFFEES";

#### Seleccionar Valores a las Tablas

```
String query = "SELECT COF_NAME, PRICE" +
  "FROM COFFEES";
ResultSet rs = stmt.executeQuery(query);
while (rs.next()) {
  String s = rs.getString("COF_NAME");
  Float n = rs.getFloat("PRICE");
  System.out.println(s + " " + n);
```

```
String updateString = "UPDATE COFFEES SET SALES = 75 " +
"WHERE COF_NAME LIKE 'Colombian'";

stmt.executeUpdate(updateString);

PreparedStatement updateSales;

String updateString = "update COFFEES " +
"set SALES = ? where COF_NAME like ?";

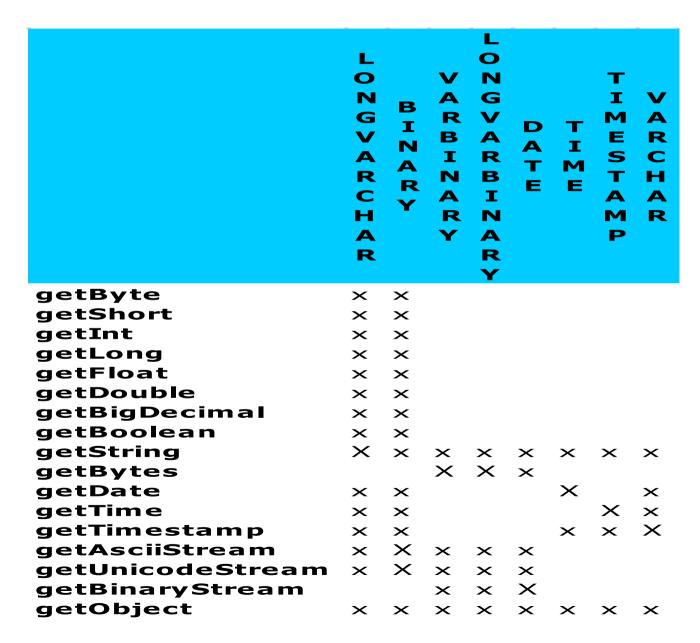
updateSales = con.prepareStatement(updateString);
```

```
int [] salesForWeek = {175, 150, 60, 155, 90};
String [] coffees = {"Colombian", "French_Roast", "Espresso",
  "Colombian_Decaf", "French_Roast_Decaf");
int len = coffees.length;
for(int i = 0; i < len; i++) {
     updateSales.setInt(1, salesForWeek[i]);
     updateSales.setString(2, coffees[i]);
     updateSales.executeUpdate();
```

	T I N Y I N T	5 M A L L I N	I N T G E R	B I G I N T	R E A L	F L O A T	D O U B L E	D E C I M A L	N U M E R I C	B I T	C H A R
getByte	X	X	X	X	X	X	X	X	X	X	×
getShort	X	X	X	X	X	X	X	X	X	X	X
getInt	X	X	X	X	X	X	X	×	X	X	×
getLong	X	X	×	X	X	X	X	×	X	X	×
getFloat	X	X	×	X	X	X	X	×	X	X	×
getDouble	×	×	×	×	×	X	X	X	X	×	X
getBigDecimal	×	×	×	×	×	×	X	X	X	X	X
getBoolean	×	×	×	×	×	×	X	×	X	X	X
getString getBytes	×	×	×	×	×	×	×	×	×	×	×
getDate											×
getTime											×
getTimestamp											×
getAsciiStream											×
getUnicodeStream getBinaryStream											X
getObject								X	×		

X Recomendado Para recuperar datos JDBC

x se puede usar para recuperar datos JDBC



X Recomendado Para recuperar datos JDBC

x se puede usar para recuperar datos JDBC