JDBC

© María Teresa González Aparicio (maytega@uniovi.es)

Indice

- Definición de JDBC
- Pasos para la conexión
- Ejemplo de conexión

© María Teresa González Aparicio (maytega@uniovi.es)

Java Database Connectivity (JDBC)

- **Permitir la conexión con el servidor de bases de datos PostgreSQL.**
- **Usar el driver PostgreSQL JDBC.**
  - https://jdbc.postgresql.org/download.html (última visita 01/04/2022)
- **Usar las clases standard de Java especificadas en JDBC API y los interfaces para conectar a la base de datos.**
- **Usar métodos para realizar consultas y actualizar datos.**

Pasos para la conexión

- **Una conexión entre una aplicación Java y la base de datos necesita realizar los siguientes pasos:**

  1. **Cargar el driver para la conexión con la base de datos.**
  2. **Crear la conexión. El objeto de conexión requiere un formato URL con el nombre de la máquina, el número de puerto y el nombre de la base de datos.**
  3. **Ejecutar las sentencias SQL, para ello se necesita crear un objeto mediante una sentencia SQL.**
  4. **Retornar un conjunto de resultados (resultset), donde el conjunto de tuplas (registros) del resultado de la consulta son almacenados, para que posteriormente puedan ser gestionados.**
  5. **Cerrar la conexión.**

Ejemplo de conexión (I)

1. **Cargar el driver:**

```
import java.sql.Connection;
import java.sql.DriverManager;
```

2. **Crear la conexión:**

```
private static String USUARIO = "postgres";

private static String PASSWORD = "admin";

private static String NOMBRE_BASEDATOS = "universidad";

private static String STRING_ CONEXION =

"jdbc:postgresql://localhost:5432/"+ NOMBRE_BASEDATOS;

Connection con =
DriverManager.getConnection(CONEXION_STRING ,
 USUARIO, PASSWORD);
```


© María Teresa González Aparicio (maytega@uniovi.es)

Ejemplo de conexión (II)

3. Ejecutar la sentencia SQL:

3.1. Se necesita importar

```
import java.sql.PreparedStatement;
import java.sql.ResultSet;
import java.sql.Statement;
```

3.2. Crear un objeto con la sentencia SQL

```
//Crear la consulta SQL
StringBuilder query = new StringBuilder();
query.append("SELECT *  ");
query.append("FROM grado");
```

3.3. Ejecutar la consulta

```
//Objeto necesario para enviar las sentencias a la base de datos
Statement st = con.createStatement();
//executeQuery retorna el resultado de la consulta y se guarda en resultSet
ResultSet rs = st.executeQuery(query.toString());
```


© María Teresa González Aparicio (maytega@uniovi.es)

Ejemplo de conexión (II)

3. Ejecutar la sentencia SQL:

3.1. Se necesita importar

```
import java.sql.PreparedStatement;
import java.sql.ResultSet;
import java.sql.Statement;
```

Cuidado!
Dejar siempre un espacio en blanco antes de las últimas comillas si la consulta continua.

3.2. Crear un objeto con la s

```
//Crear la consulta SQL
StringBuilder query = new StringBuilder();
query.append("SELECT * ");
query.append("FROM grado");
```

3.3. Ejecutar la consulta

```
//Objeto necesario para enviar las sentencias a la base de datos
Statement st = con.createStatement();
//executeQuery retorna el resultado de la consulta y se guarda en resultSet
ResultSet rs = st.executeQuery(query.toString());
```


© María Teresa González Aparicio (maytega@uniovi.es)

Ejemplo de conexión (III)

4. Manejo del resultado:

Obtener el número de columnas de la tabla proporcionada como resultado

```
int numeroColumnas = rs.getMetaData().getColumnCount();
```

Construir un string con el nombre de la cabecera de cada columna

```
StringBuilder cabeceras = new StringBuilder();
for(int i = 1; i < numeroColumnas ; i++)
headers.append(rs.getMetaData().getColumnName(i) + "\t");
headers.append(rs.getMetaData().getColumnName(numeroColumnas));
```

Presenta el string con el nombre de las cabeceras

```
System.out.println(headers.toString());
```


© María Teresa González Aparicio (maytega@uniovi.es)

Ejemplo de conexión (IV)

4. Manejo del resultado:

```
StringBuilder result = null;
```

Mientras existan tuplas (registros) para ser procesados en resultset

```
while (rs.next()) {
        result = new StringBuilder();
```

Procesa la tupla, y se concatena el valor de cada atributo (columna)

```
    for (int i = 1; i < numeroColumnas ; i++)
            result.append(rs.getObject(i) + "\t");
```

Concatena el número de atributos (columnas)

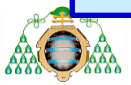
```
    result.append(rs.getObject(numeroColumnas));
```

Presenta el valor de la tupla completa (fila)

```
    System.out.println(result.toString()); }
```

No hay ninguna tupla (fila) como resultado de la consulta

```
if(result == null)  System.out.println("No hay datos");
```



© María Teresa González Aparicio (maytega@uniovi.es)

Ejemplo de conexión (V)

5. Cerrar la conexión:

Cerrar el objeto ResultSet

```
rs.close();
```

Cerrar el objeto Statement

```
st.close();
```

Cerrar el objeto de conexión

```
con.close();
```

Cerrar las conexiones en orden inverso a como fueron creadas

Ejemplo de conexión completo (I)

```
public static void ejemplo1() throws SQLException{
Connection con = getConnection();

//Crear la sentencia SQL
StringBuilder query = new StringBuilder();
query.append("SELECT * " );
query.append("FROM modulo");
 //Objeto necesario para enviar las sentencias a la base de datos
      Statement st = con.createStatement();
      //executeQuery retorna el resultado de la consulta y se guarda en resultSet
      ResultSet rs = st.executeQuery(query.toString());

//objeto de tipo Statement, para enviar las instrucciones a la BBDD
Statement st = con.createStatement();

//executeQuery devuelve un resultado que se almacena en resultSet
ResultSet rs = st.executeQuery(query.toString());
presentaResultados(rs);

//IMPORTANTE: Cerrar todos los objetos relacionados con la BBDD
rs.close();
st.close();
con.close();}
```


© María Teresa González Aparicio (maytega@uniovi.es)

Ejemplo de conexión completo (II)

```
private static Connection getConnection() throws SQLException{

      return DriverManager.getConnection(STRING_CONEXION,USUARIO,PASSWORD);}
```

```
private static void presentaResultados(ResultSet rs) throws SQLException {
    int numeroColumnas = rs.getMetaData().getColumnCount();
    StringBuilder headers = new StringBuilder();
    for(int i = 1; i < numeroColumnas ; i++)
         headers.append(rs.getMetaData().getColumnName(i) + "\t");
     headers.append(rs.getMetaData().getColumnName(numeroColumnas));
     System.out.println(headers.toString());
     StringBuilder result = null;
     while (rs.next()) {
          result = new StringBuilder();
          for(int i = 1; i < numeroColumnas ; i++)
              result.append(rs.getObject(i) + "\t");
          result.append(rs.getObject(numeroColumnas));
          System.out.println(result.toString()); }
if(result == null)
    System.out.println("No hay datos"); }
```

Sentencias con Parámetros

3. Ejecutar la sentencia SQL: Cuando algunos datos de la consulta necesitan ser parametrizados, entonces la sentencia SQL necesita ser creada de la siguiente manera:

3.1. Crear un objeto con la sentencia SQL

```
//Crear la consulta SQL
StringBuilder query = new StringBuilder();
query.append("SELECT *  ");
query.append("FROM profesor where profesor_nombre = ?");
```

3.2. Preparar la sentencia

```
System.out.println("Introduce el nombre del profesor: ");
String nombre = ReadString();
st.setString(1, nombre);
//executeQuery retornará un resultSet
ResultSet rs = st.executeQuery();
```


© María Teresa González Aparicio (maytega@uniovi.es)

Sentencias con Parámetros

3. Ejecutar la sentencia SQL: Cuando algunos datos de la consulta necesitan ser parametrizados, entonces la sentencia SQL necesita ser creada de la siguiente manera:

Si la consulta necesitase más parámetros, entonces

- **Escribir '?' donde se ubique el parámetro**.
- **Parámetros se enumeran de 1 en adelante.**
- **Después, inicializa cada valor del parámetro como sigue:**

  st.setString(<posición_parámetro_dentro_consulta>, <nombre_variable>)

3.2. Prepara cia

```
System.out.println("Introduce el nombre del profesor: ");
String nombre = ReadString();
st.setString(1, nombre);
//executeQuery retornará un resultSet
ResultSet rs = st.executeQuery();
```

Sentencias con Parámetros

```
public static void ejemplo2() throws SQLException{
        Connection con = getConnection();
        //Creacion de la consulta SQL
        StringBuilder query = new StringBuilder();
        query.append("SELECT * ");
        query.append("FROM profesor WHERE profesor_nombre = ?");
        //Objeto Statement para enviar sentencias a la base de datos
        PreparedStatement st = con.prepareStatement(query.toString());
        System.out.println("Introduce el nombre del profesor: ");
        String nombre = ReadString();
        st.setString(1, nombre);
        //executeQuery retornará un resultSet
        ResultSet rs = st.executeQuery();
        presentaResultados(rs);
        rs.close();
        st.close();
        con.close(); }
```


© María Teresa González Aparicio (maytega@uniovi.es)

Funciones – Input data

```
import java.util.Scanner;

private static String LeerString(){
        return new Scanner(System.in).nextLine();
}

private static int LeerInt(){
        return new Scanner(System.in).nextInt();
}
```


© María Teresa González Aparicio (maytega@uniovi.es)

JDBC

© María Teresa González Aparicio (maytega@uniovi.es)