EJECUCIÓN DE SENTENCIAS DE MANIPULACIÓN DE DATOS

Javier García-Retamero Redondo

OBJETIVO

- El objetivo principal es ejecutar consultas sobre los datos almacenados.
- Utilizaremos la interfaz **Statement** (sentencia)

EJECUCIÓN DE SENTENCIAS DE MANIPULACIÓN DE DATOS

SENTENCIAS SIMPLES

CONSULTAS SENCILLAS

- **ResulSet executeQuery(String):**
  - Se utiliza para sentencias SELECT que recuperan datos de un único ResulSet.

```
String sql=“SELECT * FROM depart";
Connection conexion = DriverManager.getConnection(……);

Statement sentencia = conexion.createStatement ();
Resulset resul= sentencia.executeQuery(sql);
```

EJECUCIÓN DE SENTENCIAS DE MANIPULACIÓN DE DATOS

DML Y DDL

DML Y DDL

- **int executeUpdate(String):**
  - Se utiliza para:
    - DML (insert, delete y update): Devuelve el número de filas afectadas.
    - DDL (create, drop, alter): Devuelve 0

```
String sql="insert into departamentos values (" + dep + ","'+dnombre+"',"'+loc+"')";
Connection conexion = DriverManager.getConnection(.....);

Statement sentencia = conexion.createStatement ();
int filas = sentencia.executeUpdate(sql);
```

EJECUCIÓN DE SENTENCIAS DE MANIPULACIÓN DE DATOS

CUALQUIER SENTENCIA

CUALQUIER SENTENCIA

- **boolean execute(String):**
  - Vale para ejecutar cualquier sentencia SQL.
  - Devuelve TRUE: Cuando devuelve un Resulset
    - Utilizar getResulset() para obtener el resulset devuelto
  - Devuelve FALSE: Cuando devuelve un recuento de actualizaciones o no hay resultados.
    - Utilizar getUpdateCount() para recuperar el valor devuelto

CUALQUIER SENTENCIA

```
String sql="SELECT * FROM departamentos";
Connection conexion = DriverManager.getConnection(……);
Statement sentencia = conexion.createStatement ();

Boolean valor = sentencia.execute(sql);

if (valor) {
        Resulset rs = sentencia.getResulset();
        ……Recorremos el resulset
} else {
        int f = sentencia.getUpdateCount();
}
```

EJECUCIÓN DE SENTENCIAS DE MANIPULACIÓN DE DATOS

SENTENCIAS PREPARADAS

SENTENCIAS PREPARADAS

- Podemos utilizar marcadores de posición para evitar tener que concatenar cadenas de caracteres a la hora de formar la cadena SQL.
- De esta forma precompilaremos la sentencia una vez y le pasaremos los valores en tiempo de ejecución.
- Con Statement normales, se compila la sentencia cada vez que se ejecuta.

SENTENCIAS PREPARADAS

Sentencia con Statement	Sentencia con Preparedstatement
String sql="insert into departamentos values (" + dep + ",'"+dnombre+"','"+loc+"')";	String sql="insert into departamentos values (?, ?, ?)";
Statement sentencia=conexion.createStatement();	PreparedStatement sentencia=conexion.prepareStatement(sql);
	La sentencia se precompila…
	Asigna los valores, del tipo correcto, a cada interrogación: sentencia.setInt (1, dep); sentencia.setString (2 , dnombre); sentencia.setString (3 , loc); Si la variable del paréntesis no fuera del mismo tipo, utilizaríamos: Integer.parseInt(v), … Float.parseFloat(v)
int filas = sentencia.executeUpdate(sql); //Aquí cada vez que se ejecuta se compila	int filas = sentencia.executeUpdate(); //Se ejecuta la sentencia ya precompilada con los valores suministrados

SENTENCIAS PREPARADAS

Método	Tipo SQL
setString (i,String)	VARCHAR
setBoolean(i,boolean)	BIT
setByte(i,byte)	TINYINT
setShort(i,short)	SMALLINT
setInt (i,int)	INTEGER
setLong (i,long)	BIGINT
setFloat (i,float)	FLOAT
setDouble(i,double)	DOUBLE
setDate (i,Date)	DATE
setNull(i, int tipoSQL)	Pone a null una posición de un tipo especificado en java.sql.Types