JDBC_II

Ejecución de sentencias de descripción de datos

- ¿Qué pasa si desconocemos la estructura de las tablas de una base de datos? Es decir, ¿cuáles son los campos de una tabla?¿Cómo son las restricciones de integridad? Etc.
- La interfaz **DatabaseMetaData** proporciona dicha información. Los métodos que ofrece son:
  - **getTables()** → Información sobre tablas y vistas
  - **getColumns()** → Información sobre columnas de una tabla
  - **getPrimaryKeys()** → Información sobre columnas que forman la clave primaria de una tabla
  - **getExportedKeys()** → Información sobre claves ajenas (foráneas) que utilizan la clave primaria de una tabla (que apuntan a la tabla)
  - **getImportedKeys()** → Información sobre claves ajenas de una tabla
  - **getProcedures()** → Información sobre procedimientos almacenados.

```
import java.sql.*;
public class EjemploDatabaseMetadata {
	public static void main (String [] args) {
	try
	{
		Class.forName("com.mysql.cj.jdbc.Driver");
		Connection conexion = DriverManager.getConnection
			("jdbc:mysql://localhost/ejemplo","root","austria");
		DatabaseMetaData dbmd = conexion.getMetaData();
		ResultSet resul = null;
		String nombre = dbmd.getDatabaseProductName();
		String driver = dbmd.getDriverName();
		String url = dbmd.getURL();
		String usuario = dbmd.getUserName();
		System.out.printf("Nombre: %s %n", nombre);
		System.out.printf("Nombre: %s %n", driver);
		System.out.printf("Nombre: %s %n", url);
		System.out.printf("Nombre: %s %n", usuario);
		resul = dbmd.getTables(null,"ejemplo",null,null);
		while (resul.next()){
			String catalogo = resul.getString(1);
			String esquema = resul.getString(2);
			String tabla = resul.getString(3);
			String tipo = resul.getString(4);
			System.out.printf("%s - Catalogo: %s, Esquema: %s, Nombre: %s %n", tipo,catalogo, esquema, tabla);
		}
		conexion.close();
	}
	catch (ClassNotFoundException cn) {cn.printStackTrace();}
	catch (SQLException e) {e.printStackTrace();}
	}
}
```

Método getTables()

- Devuelve un objeto ResultSet que proporciona información sobre las tablas y vistas de la base de datos. Su sintaxis es:

```
public abstract ResultSet getTables( String catalogo, String esquema,
                                     String patronDeTabla, String tipos[ ]) throws SQLException
```

- El significado de los parámetros es:
  - Primer parámetro: catálogo de la base de datos. Al poner null indicamos el catálogo actual.
  - Segundo parámetro: esquema de la base de datos. Al poner null indicamos esquema actual.
  - Tercer parámetro: es un patrón en el que se indica el nombre de las tablas que queremos que obtenga el método. Se puede utilizar el carácter guión bajo o porcentaje. Así, por ejemplo, “de%” obtendría todas las tablas que empiezan por “de”.
  - Cuarto parámetros: un array de String en el que indicamos qué tipos de objetos queremos obtener. Por ejemplo: TABLE, VIEW, etc. Al poner null nos devolverá todos los tipos.

- Cada fila de ResultSet que devuelve **getTables()** tiene información sobre una tabla. La descripción de cada columna tiene las siguientes columnas:
  - TABLE_CAT → Columna1: el nombre del catálogo al que pertenece la tabla
  - TABLE_SCHEM → Columna2: el nombre del esquema al que pertenece la tabla
  - TABLE_NAME → Columna3: el nombre de la tabla o vista
  - TABLE_TYPE → Columna4: tipo TABLE o VIEW
  - REMARKS → Columna5: comentarios
  - Y más: TYPE_CAT, TYPE_SCHEM, TYPE_NAME, SELF_REFERENCING_COL_NAME, REF_GENERATION

método getColumns()

- Devuelve un objeto ResultSet con información sobre las columnas de una tabla o tablas. La descripción de cada columna tiene las siguientes columnas: TABLE_CAT, TABLE_SCHEM, TABLE_NAME, COLUMN_NAME, DATA_TYPE, TYPE_NAME, COLUMN_SIZE, BUFFER_LENGHT, DECIMAL_DIGITS, NUM_PREC_RADIX, NULLABLE, REMARKS, COLUMN_DEF, SQL_DATA_TYPE, SQL_DATETIME_SUB, CHAR_OCTET_LENGTH, ORDINAL_POSITION, IS_NULLABLE, IS_AUTOINCREMENT, etc.

```
public abstract ResultSet getColumns( String catalogo, String Esquema, String patronNombreDeTabla, String patronNombreDeColumna) throws SQLException
```

- Para patrón de tabla y de columna se puede utilizar el porcentaje o el guión bajo.

Más métodos de DatabaseMetaData

- **getPrimaryKeys()** → devuelve la lista de columnas que forman la clave primaria de la tabla especificada
- **getExportedKeys()** → devuelve la lista de todas las claves ajenas que utilizan la clave primaria de la tabla especificada.
- **getImportedKeys()** → devuelve la lista de claves ajenas existentes en la tabla indicada
- **getProcedures()** → devuelve la lista de procedimientos y funciones almacenadas.

problemas JDBC_II_1

1. Prueba el programa del ejemplo pero contra tu MV.
2. Genera un programa en Java que muestre el nombre, el tipo, el tamaño y si puede ser nulo o no, de las columnas de la tabla departamentos.
3. Genera un programa que devuelva la clave primaria de la tabla departamentos y la clave ajena que apunta a la tabla departamentos. NOTA: revisa que estén creadas las claves; tanto la primaria como la foránea.
4. Busca información sobre la interfaz **ResultSetMetaData** y realiza un programa utilizando dicha interfaz que obtenga el número de columnas y el tipo de columnas devueltos por la consulta “SELECT * FROM DEPARTAMENTOS”.

Ejecución DML y DDL

- Al crearse un objeto **Statement** se crea un espacio de trabajo para crear consultas SQL, ejecutarlas y para recibir los resultados. Una vez creado el objeto se pueden usar los siguientes métodos:
  - **ResultSet executeQuery(String)** → para sentencias SELECT
  - **int executeUpdate(String)** → se utiliza para sentencias que no devuelvan un **ResultSet** como son las sentencias DML: INSERT, UPDATE Y DELETE; y las sentencias DDL: CREATE, DROP y ALTER. El método devuelve un entero indicando el número de filas que se vieron afectadas y, en caso de las sentencias DDL, devuelve 0.
  - **boolean execute(String)** → se puede utilizar para ejecutar cualquier sentencia SQL. Tanto las que devuelven un **ResultSet** (SELECT), como para las que devuelven el número de filas afectadas (INSERT, UPDATE, DELETE) y para las de definición de datos (CREATE). El método devuelve *true* si devuelve un **ResultSet** (para recuperar las filas será necesario llamar al método **getResultSet()**) y *false* si se trata de actualizaciones o no hay resultados; en este caso se usará el método **getUpdateCount()** para recuperar el valor devuelto de filas afectadas.

Ejemplo: INSERT

```
import java.sql.*;
public class EjemploInsert {
	public static void main (String[] args){
	try{
		Class.forName("com.mysql.cj.jdbc.Driver");
		Connection conexion = DriverManager.getConnection ("jdbc:mysql://localhost/ejemplo","root","austria");
		String dep = args[0];
		String dnombre = args[1];
		String loc = args[2];
		String sql = String.format("INSERT INTO depart VALUES (%s, '%s', '%s')", dep, dnombre, loc);
		System.out.println(sql);
		Statement sentencia = conexion.createStatement();
		int filas = sentencia.executeUpdate(sql);
		System.out.printf("Filas afectadas: %d %n", filas);
		sentencia.close();
		conexion.close();
	}
	catch (ClassNotFoundException cn) {cn.printStackTrace();}
	catch (SQLException e) {e.printStackTrace();}
	}
}
```

Problemas JDBC_II_2

1. Realiza un programa en Java que suba el salario a los empleados de un departamento. El programa recibirá el número de departamento y el incremento.
2. Realiza un programa que cree una vista (de nombre “totales”) que contenga por cada departamento el número de departamento, el nombre, el número de empleados que tiene y el salario medio.
3. Crea un programa Java que inserte un empleado en la tabla *emple*, el programa recibe del usuario los valores a insertar. Los argumentos que recibe son: EMP_NO, APELLIDO, OFICIO, DIR, SALARIO, COMISION, DEPT_NO. Antes de insertar se deben realizar una serie de comprobaciones:

- Que el departamento exista en la tabla depart, si no existe no se inserta
- Que el número del empleado no exista, si existe no se inserta
- Que el salario sea mayor que cero, si no lo es, no se inserta
- Que el director (dir → “jefe” del empleado) exista, si no existe no se inserta
- El apellido y el oficio no pueden ser nulos
- La fecha de alta del empleado es la fecha actual.

Cuando se inserte la fila visualizar mensaje y si no se inserta visualizar el motivo.