Java JDBC

Octavio Robleto

octavio.robleto@gmail.com

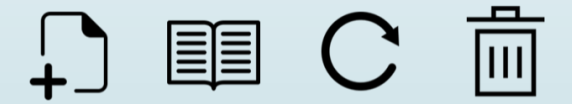
https://orobleto.github.io/octaviorobleto.github.io/

Motor de Base de Datos

- Un motor de Bases de Datos es un conjunto de programas que permiten el almacenamiento, modificación y extracción de la información en una base de datos , además de proporcionar herramientas para añadir, borrar, modificar y analizar los datos.
- Los usuarios pueden acceder a la información usando herramientas específicas de consulta y / o de generación de informes, o bien mediante aplicaciones al efecto.
- En base a esto, podemos definir que los motores de base de datos sirven para **definir** , **construir** y **manipular** una base de datos .

Que podemos hacer con esto?

- **1- Definir una base de datos:** consiste en especificar los tipos de datos, estructuras y restricciones para los datos que se almacenarán .
- **2- Construir una base de datos:** es el proceso de almacenar los datos sobre algún medio de almacenamiento .
- **3- Manipular una base de datos:** incluye funciones como consulta , actualización, etc. de bases de datos.



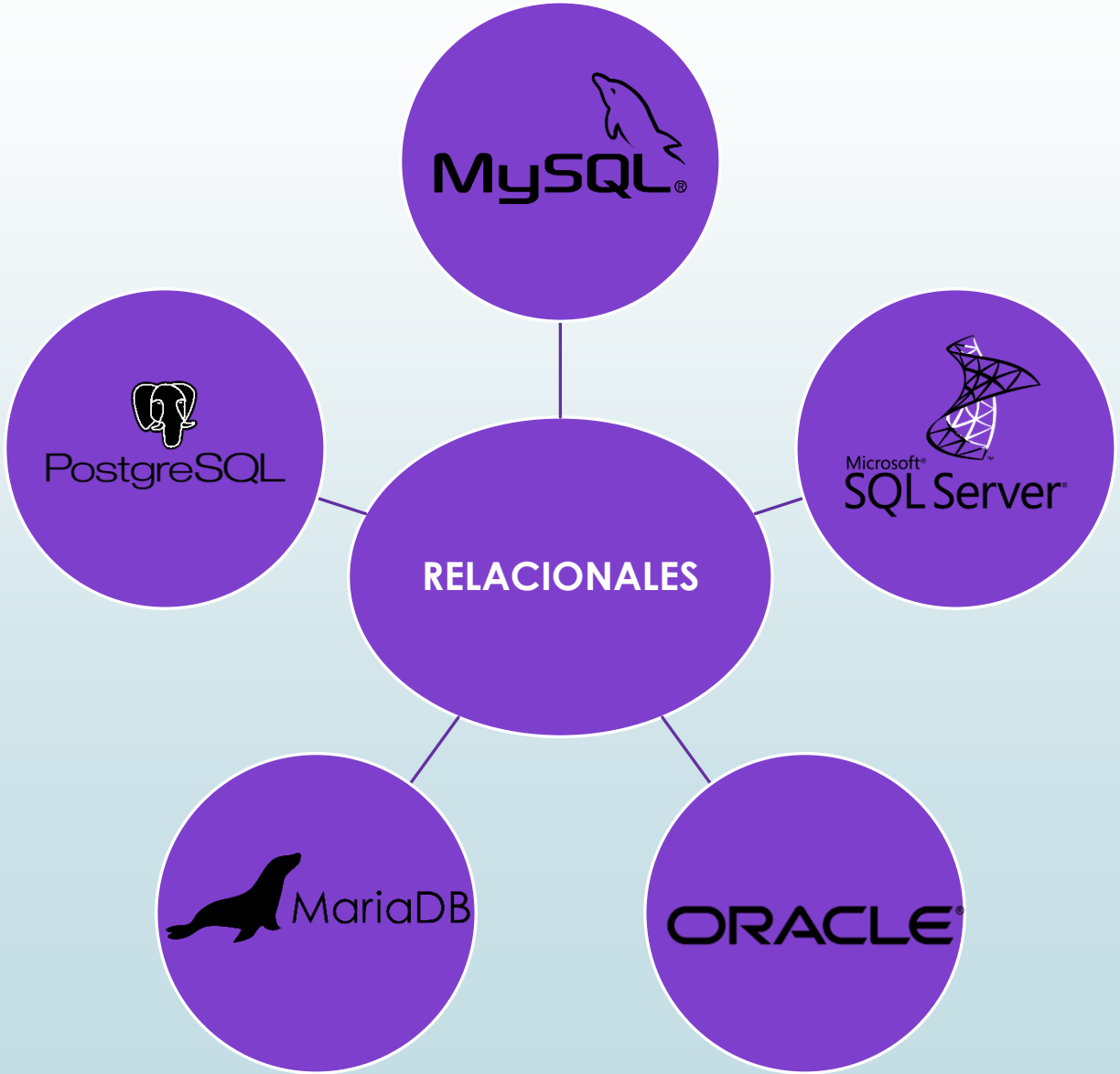
C R U D

A B M C

Control de Concurrencia

- Se refiere al hecho de que los DBMS (Sistemas de Administración de Bases de Datos) permiten que muchas transacciones accedan a una misma base de datos a la vez. Esto para asegurar que las transacciones concurrentes no interfieran entre sí

Motores de Base de Datos



Que necesitamos:

- https://dbeaver.io/download/

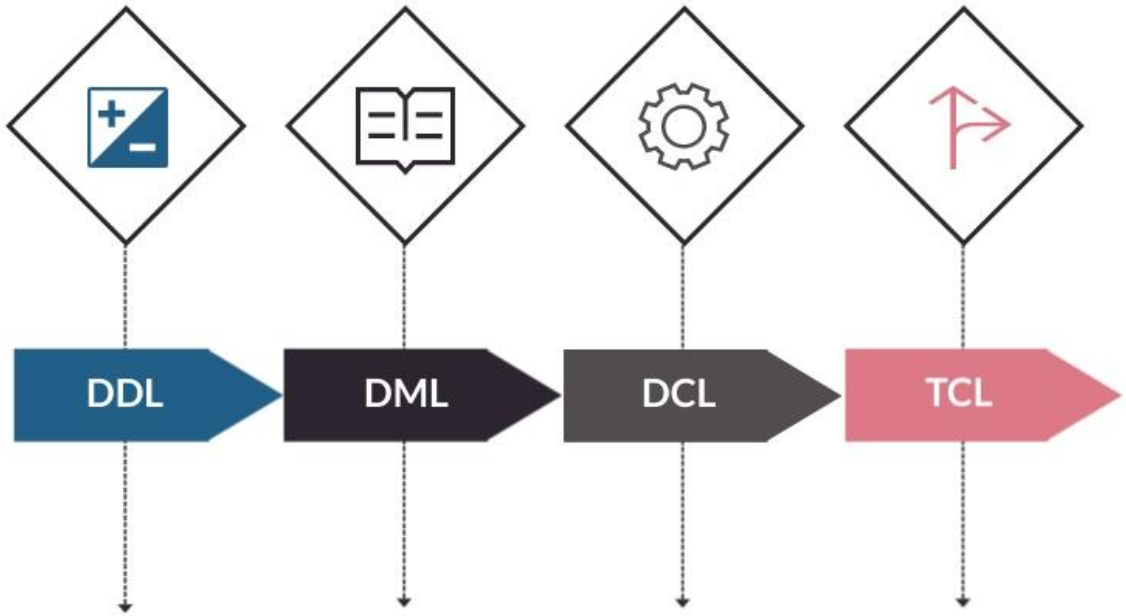
- https://www.apachefriends.org/es/download.html

- https://www.miblocdenotas.com/277061

SQL

- Es un Lenguaje Declarativo estándar de alto nivel con el cual se comunica con las bases de datos relacionales.
- Su nombre en ingles “Structured Query Language”, que traducido al español significa : Lenguaje de Consultas Estructurado.

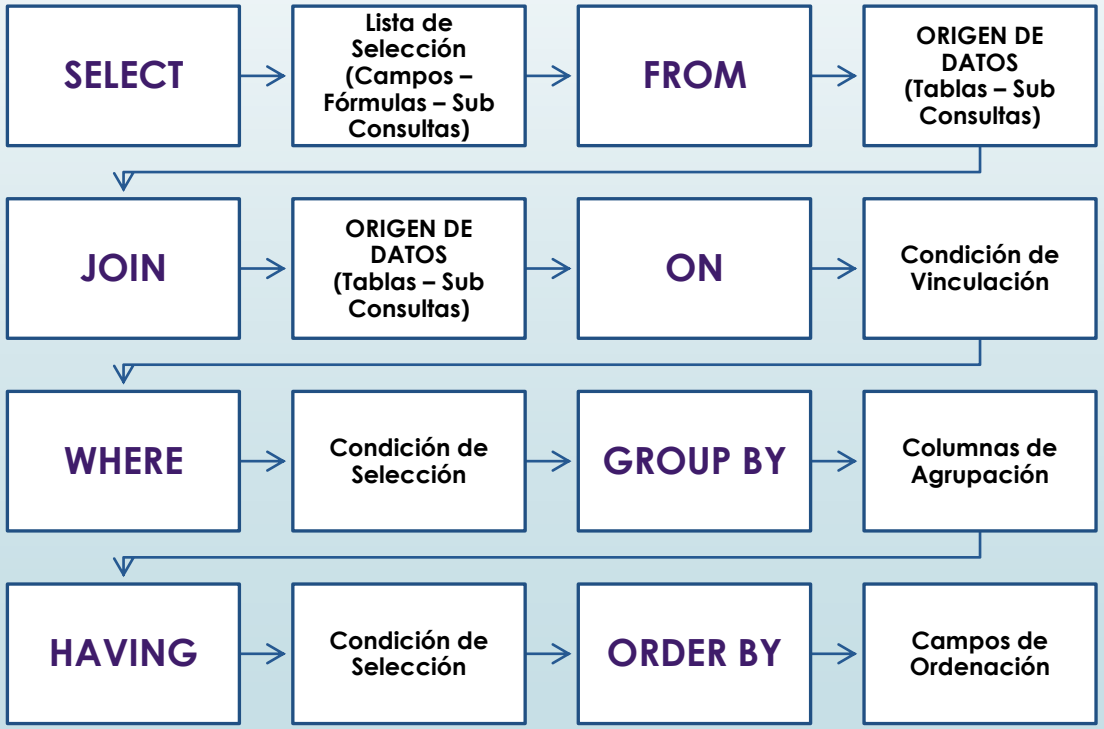
CLASIFICACIÓN DE SQL



CREADO POR
Octavio Robleto
https://orobleto.github.io/octaviorobleto.github.io

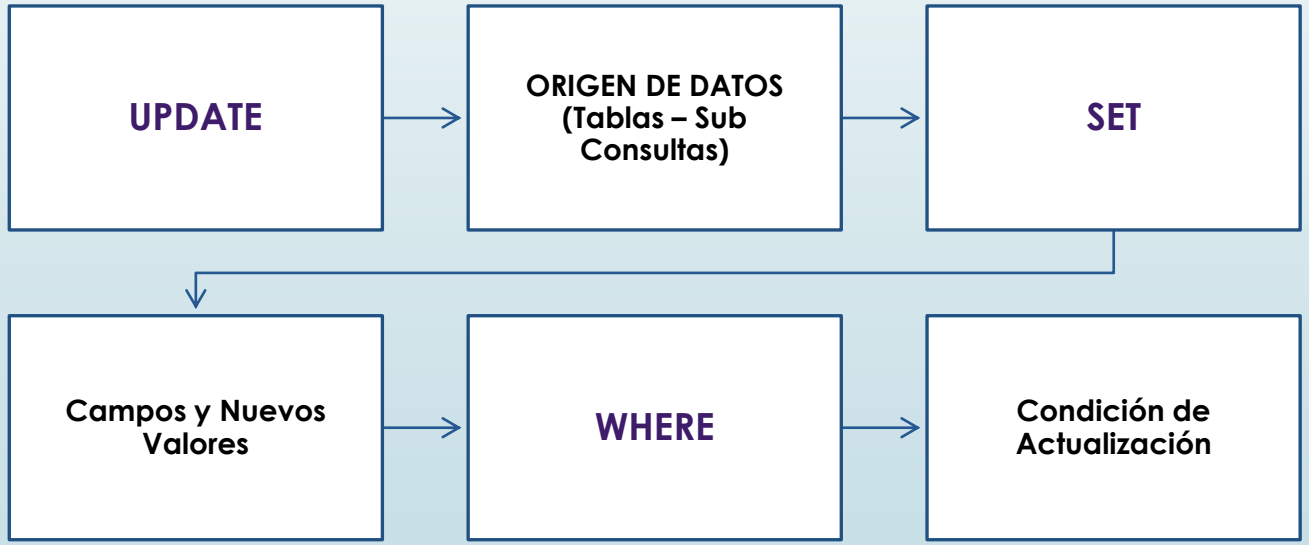
Select

- Recupera filas de la base de datos y permite la selección de una o varias filas o columnas de una o varias tabla.



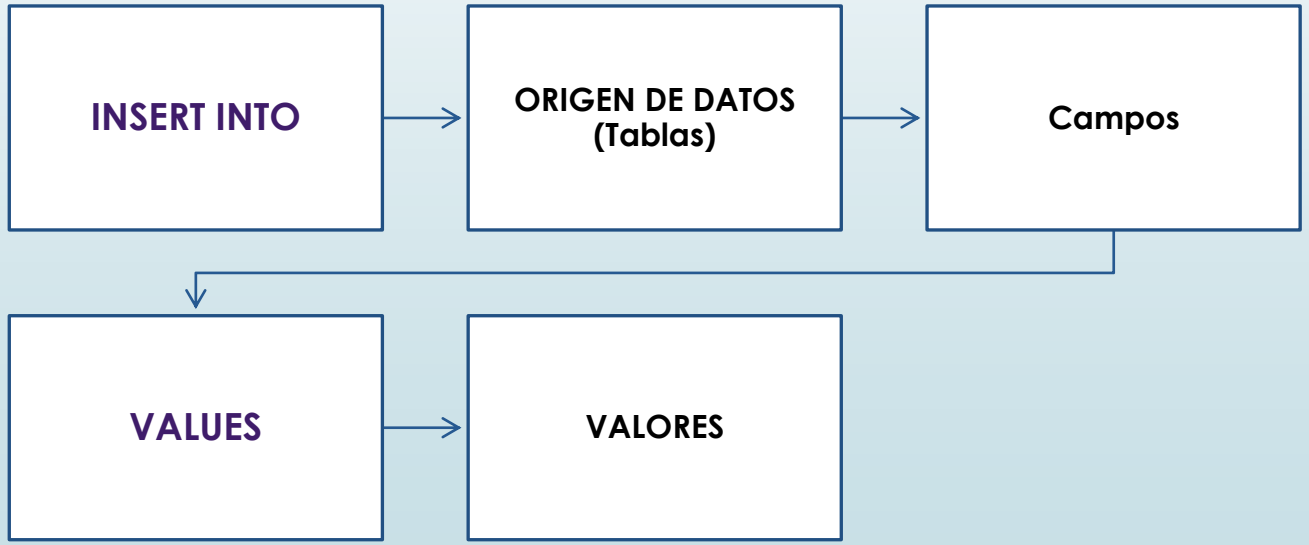
Update

- Modifica los datos de una tabla.



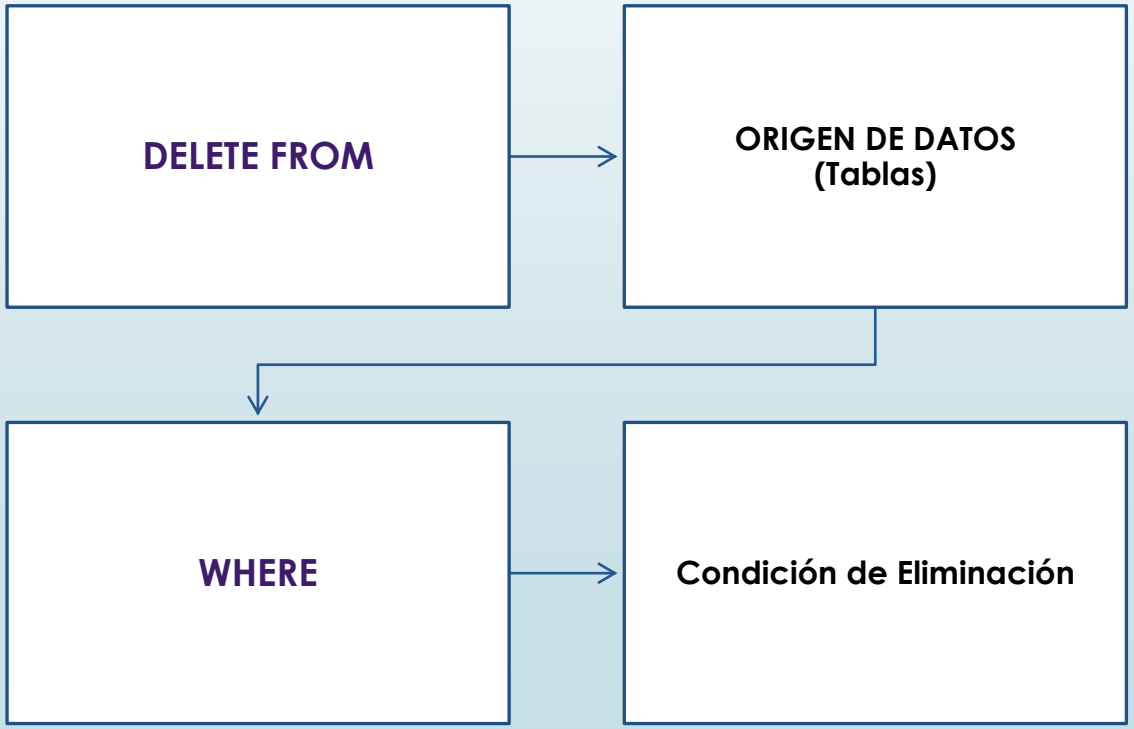
Insert

- se utiliza para insertar nuevas filas en una tabla.



Delete

- sirve para borrar filas de una tabla.

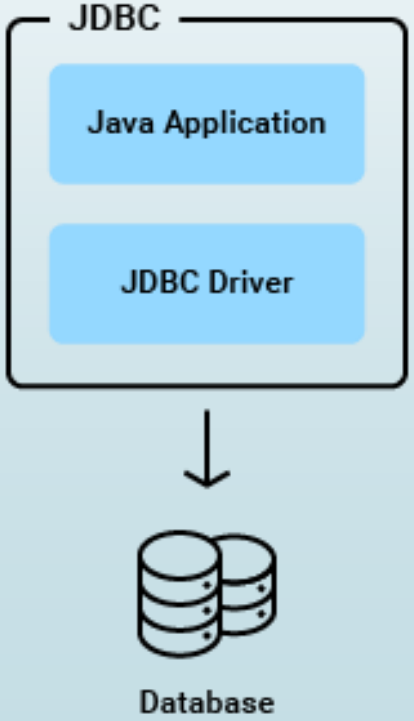


Java DataBase Connectivity (JDBC)

- Es una API de acceso a bases de datos estándar SQL que proporciona un acceso uniforme a una gran variedad de bases de datos relacionales.
- JDBC también proporciona una base común para la construcción de herramientas y utilidades de alto nivel.
- El paquete actual de JDK incluye JDBC.

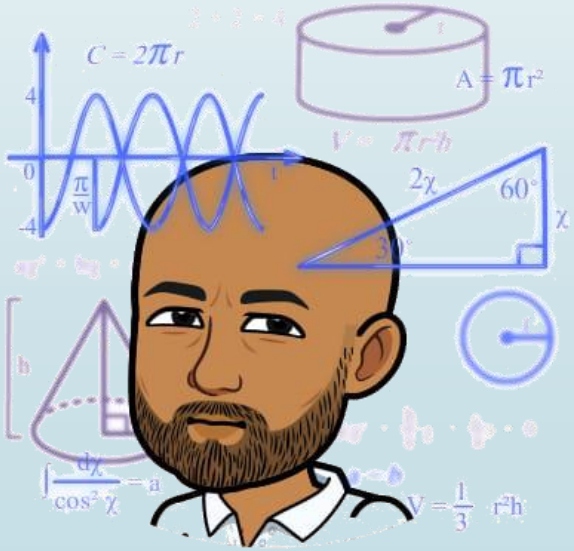
Pero, Que es?

- Consiste en un conjunto de clases e interfaces escritas en el lenguaje de programación Java. JDBC suministra un API estándar para los desarrolladores y hace posible escribir aplicaciones de base de datos usando un API puro Java.



Que hace?

- Simplemente JDBC hace posible estas tres cosas:
  - Establece una conexión con la base de datos.
  - Envía sentencias SQL.
  - Procesa los resultados.



Como nos Conectamos?

```
String url = "";
String user = "";
String password = "";
Connection conexion = null;

try {
    Class.forName("");
    conexion = DriverManager.getConnection(url, user, password);
} catch (ClassNotFoundException | SQLException e) {
    System.out.println(e.getMessage());
}
```

Como Buscamos?

- Un Objeto **ResultSet** contiene todas las filas que satisfacen las condiciones de una sentencia SQL y proporciona el acceso a los datos de estas filas mediante un conjunto de métodos get que permiten el acceso a las diferentes columnas de la filas.

```
ResultSet resultSet = null;

try {
    Statement statement = conexion.createStatement();

    // Create and execute a SELECT SQL statement.
    String selectSql = "SELECT Campos FROM tabla...";
    resultSet = statement.executeQuery(selectSql);

    // Print results from select statement
    while (resultSet.next()) {
        resultSet.getTipo("campo");
    }
} catch (SQLException e) {
    System.out.println(e.getMessage());
}
```

Como Ejecutamos?

- Un objeto **Statement** se usa para enviar sentencias SQL a la base de datos. Se usa para ejecutar una sentencia SQL simple sin parámetros.

```
String instruccionSQL = "INSERT,UPDATE OR DELETE";

try {
    Statement statement = conexion.createStatement();
    statement.execute(instruccionSQL);

} catch (SQLException e) {
    System.out.println(e.getMessage());
}
```