



Argentina  
programa  
4.0

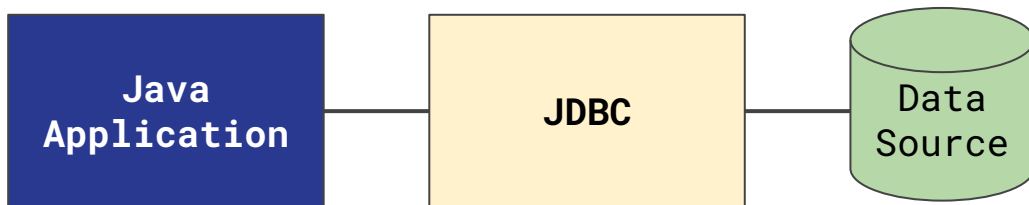
# JDBC

---

“Desarrollador Java Inicial”

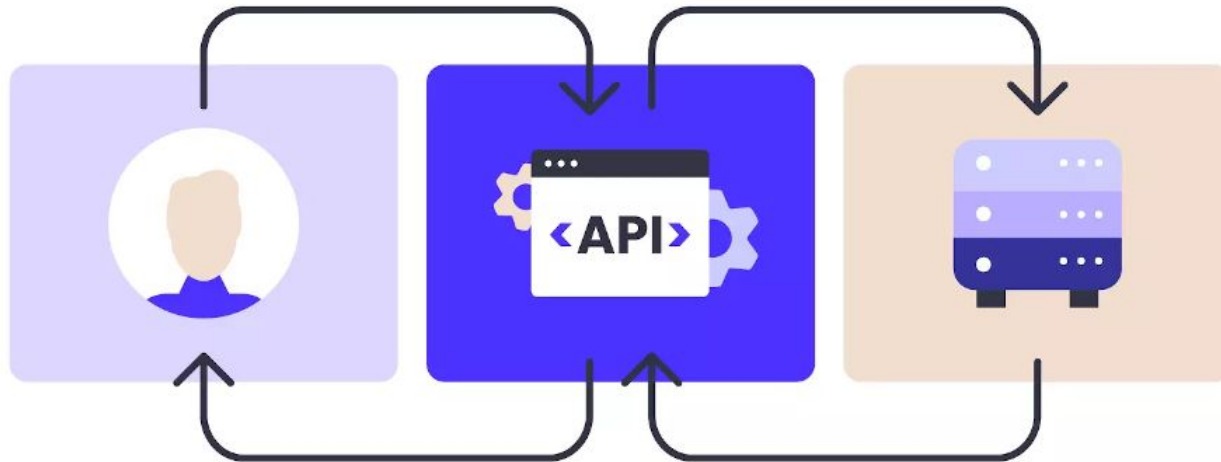
# JDBC

- **JDBC** (Java **D**atabase **C**onnectivity) conecta una aplicación Java a una fuente de datos como puede ser una DB.
- Es una API que es usada para:
  - Conectar a una fuente de datos
  - Enviar consultas y actualizaciones
  - Recupera y procesa el resultado
- JDBC utiliza drivers para conectarse a la DB.

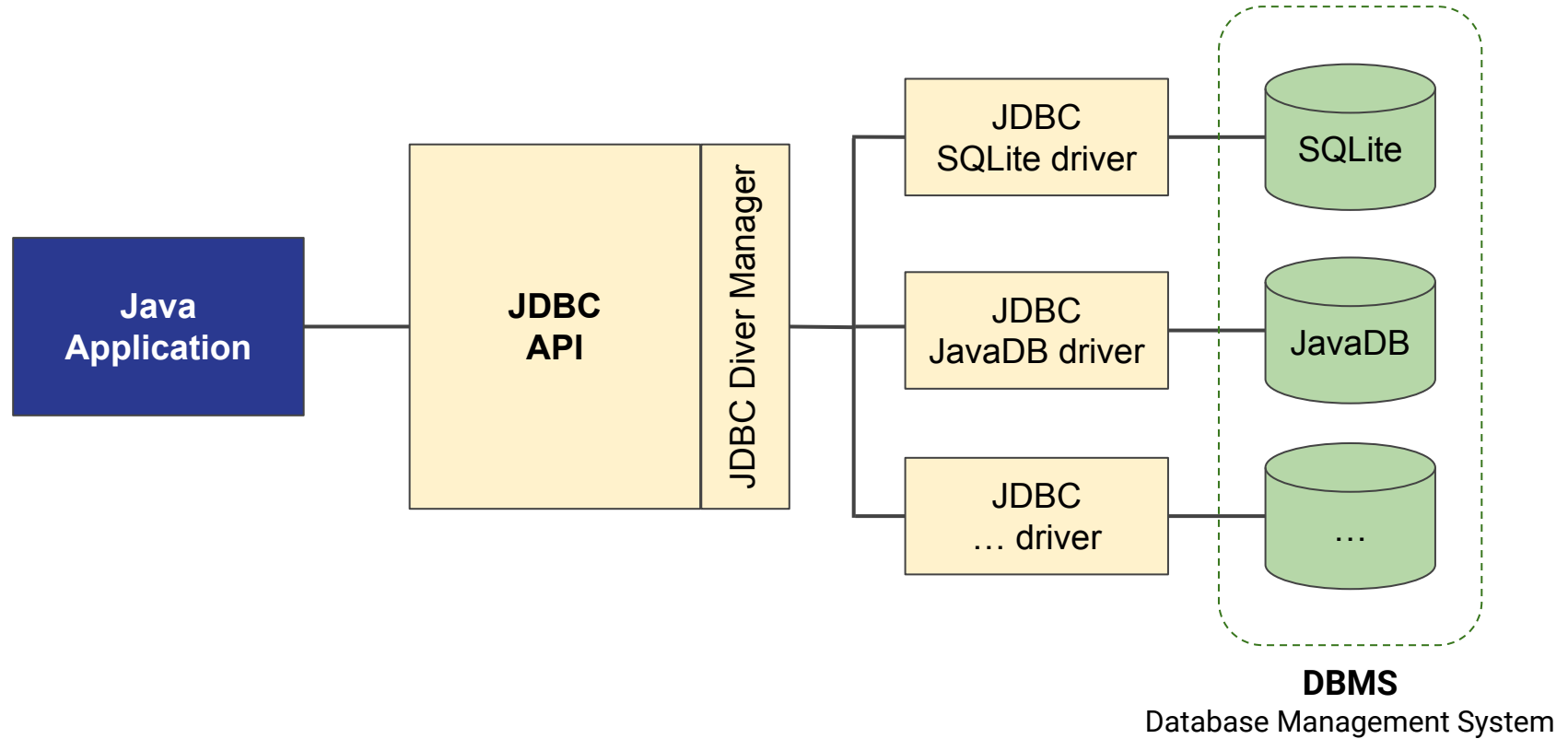


# JDBC - API

- **A**pplication **P**rogramming **I**nterfaces
- Es un conjunto de definiciones y protocolos que se utiliza para desarrollar e integrar el software de las aplicaciones, permitiendo la comunicación a través de un conjunto de reglas.

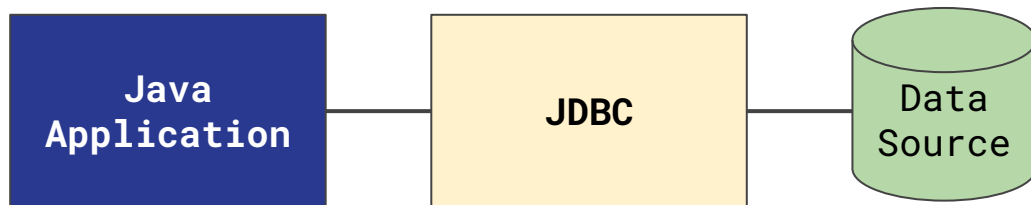


# JDBC - API



# JDBC - API

- **JDBC** (Java **D**atabase **C**onnectivity) conecta una aplicación Java a una fuente de datos como puede ser una DB.
- Es una API que es usada para:
  - Conectar a una fuente de datos
  - Enviar consultas y actualizaciones
  - Recupera y procesa el resultado
- JDBC utiliza drivers para conectarse a la DB.

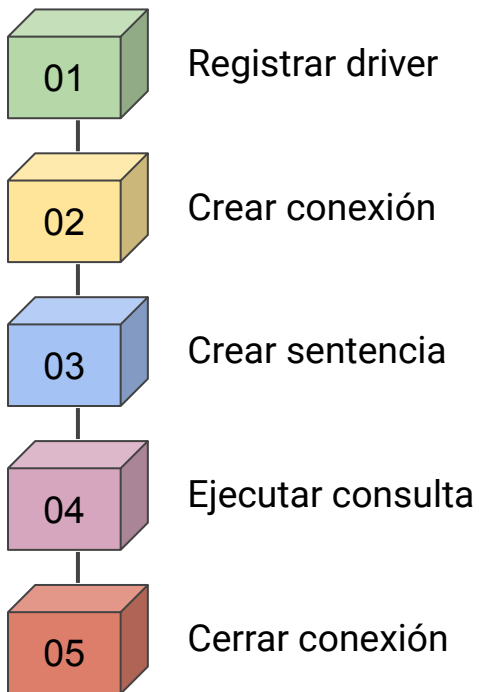


# JDBC - Implementación

- JDBC Driver es un componente de software que habilita a la aplicación java a interactuar con la DB.
- Existen 4 tipos de Drivers JDBC
  - JDBC-ODBC bridge driver
  - Native-API driver
  - Network Protocol driver
  - Thin driver

# JDBC - Implementación

Para interactuar con una DB debemos considerar 5 pasos:



# JDBC - Implementación

## 1. Registrar la 'clase' del driver - *Este método se utiliza para registrar la clase que se utilizará como driver.*

Sintaxis del método `forName()` method

```
public static void forName(String className) throws ClassNotFoundException
```

## 2. Crear el objeto de conexión - *Este método se utiliza para establecer conexión con la DB.*

Sintaxis del método `getConnection()` method

```
public static Connection getConnection(String url, String username, String password) throws SQLException
```

## 3. Crear la sentencia - *Este método es usado para crear la sentencia.*

*Esta sentencia es la responsable de ejecutar las consultas a la DB.*

Sintaxis del método `createStatement()` method

```
public Statement createStatement() throws SQLException
```



# JDBC - Implementación

## 4. Ejecutar consulta - *Este método devuelve el resultado de una consulta (filas).*

Sintaxis del método `executeQuery()` method

```
public ResultSet executeQuery(String sql) throws ClassNotFoundException
```

## 5. Cerrar la conexión - *Este método finaliza la conexión con la DB.*

Sintaxis del método `close()` method

```
public void close() throws ClassNotFoundException
```

# JDBC - Conexión

Para conectarnos a una DB (por ej. mysql) debemos conocer la siguiente información:

1. 'Clase' del driver: La clase para el driver de mysql es **`com.mysql.jdbc.Driver`**
2. URL de conexión: **`jdbc:mysql://localhost:3306/dbname`**
  - “**`jdbc`**” es la **API**
  - “**`mysql`**” es la **DB**
  - “**`localhost`**” es el nombre del **servidor** (se puede utilizar la dirección IP)
  - “**`3306`**” es el número de **puerto**
  - “**`dbname`**” es el nombre de la **DB**
3. Nombre de usuario: el usuario por defecto en mysql es **root**
4. Password: la contraseña asociada al usuario

# JDBC - Conexión

## Ejemplo de conexión con *mysql*

```
import java.sql.*;
class MysqlCon{
    public static void main(String args[]){
        try{
            Class.forName("com.mysql.jdbc.Driver");
            Connection
con=DriverManager.getConnection("jdbc:mysql://localhost:3306/dbname", "root", "password");
            Statement stmt=con.createStatement();
            //USO DE LA DB
            con.close();
        } catch(Exception e){ System.out.println(e);}
    }
}
```

Para realizar consultas a una DB utilizaremos los métodos de la interfaz **Statement**.

Los métodos más comunes son:

1. **public ResultSet executeQuery(String sql):** utilizado para ejecutar sentencias SELECT. Devuelve un ResultSet.
2. **public int executeUpdate(String sql):** utilizado para ejecutar sentencias CREATE, DROP, INSERT, UPDATE, DELETE, etc.

# JDBC - Consultas

## Ejemplo de Statements con mysql

```
import java.sql.*;
class FetchRecord{
    public static void main(String args[])throws Exception{
        Class.forName("com.mysql.jdbc.Driver");
        Connection
        con=DriverManager.getConnection("jdbc:mysql://localhost:3306/dbname","root","password");
        Statement stmt=con.createStatement();

        ResultSet rs=stmt.executeQuery("select * from emp");
        while(rs.next())
            System.out.println(rs.getInt(1)+" "+rs.getString(2)+" "+rs.getString(3));

        int result=stmt.executeUpdate("delete from emp where id=33");
        System.out.println(result + " records affected");
        con.close();
    }
}
```