Acceso a bases de datos desde Java

SGBD

JDBC

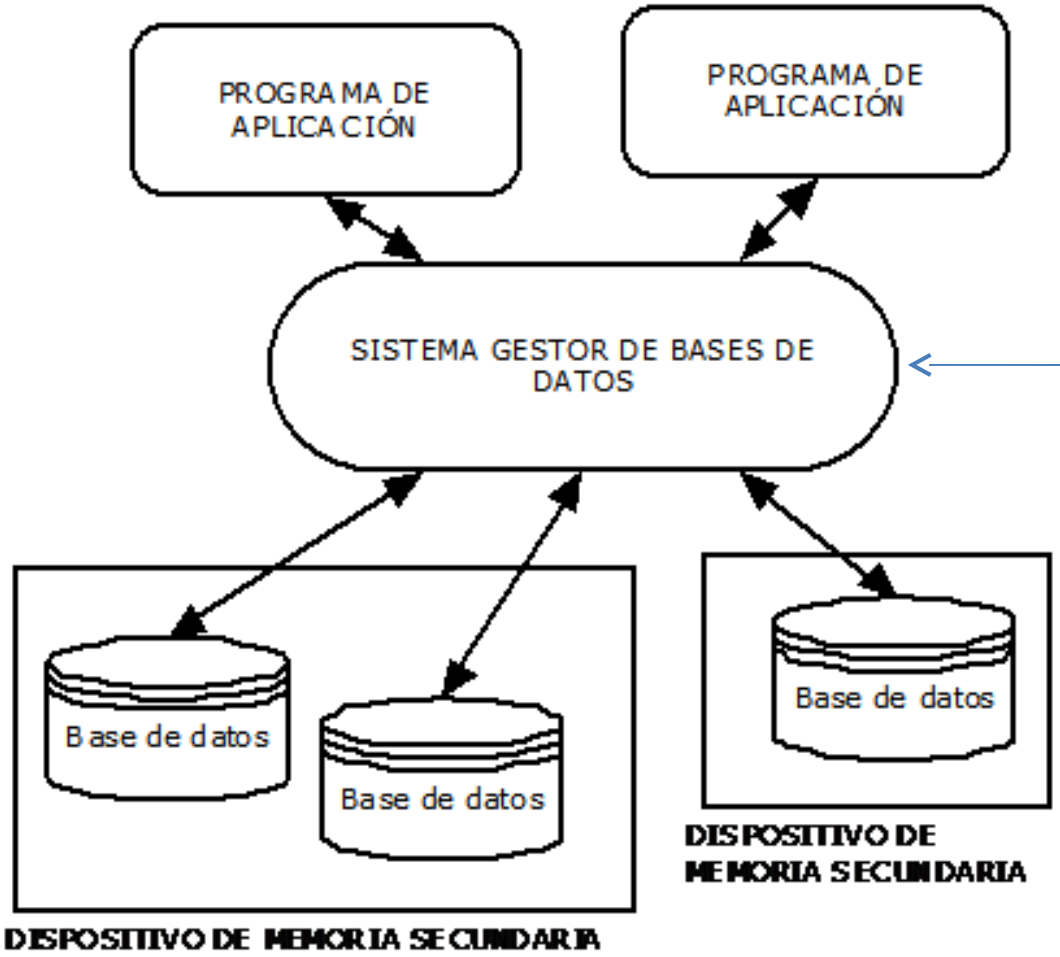
Drivers JDBC

Controlador puente JDBC-ODBC

Pasos para consultar una base de datos en Java

Transacciones

SGBD

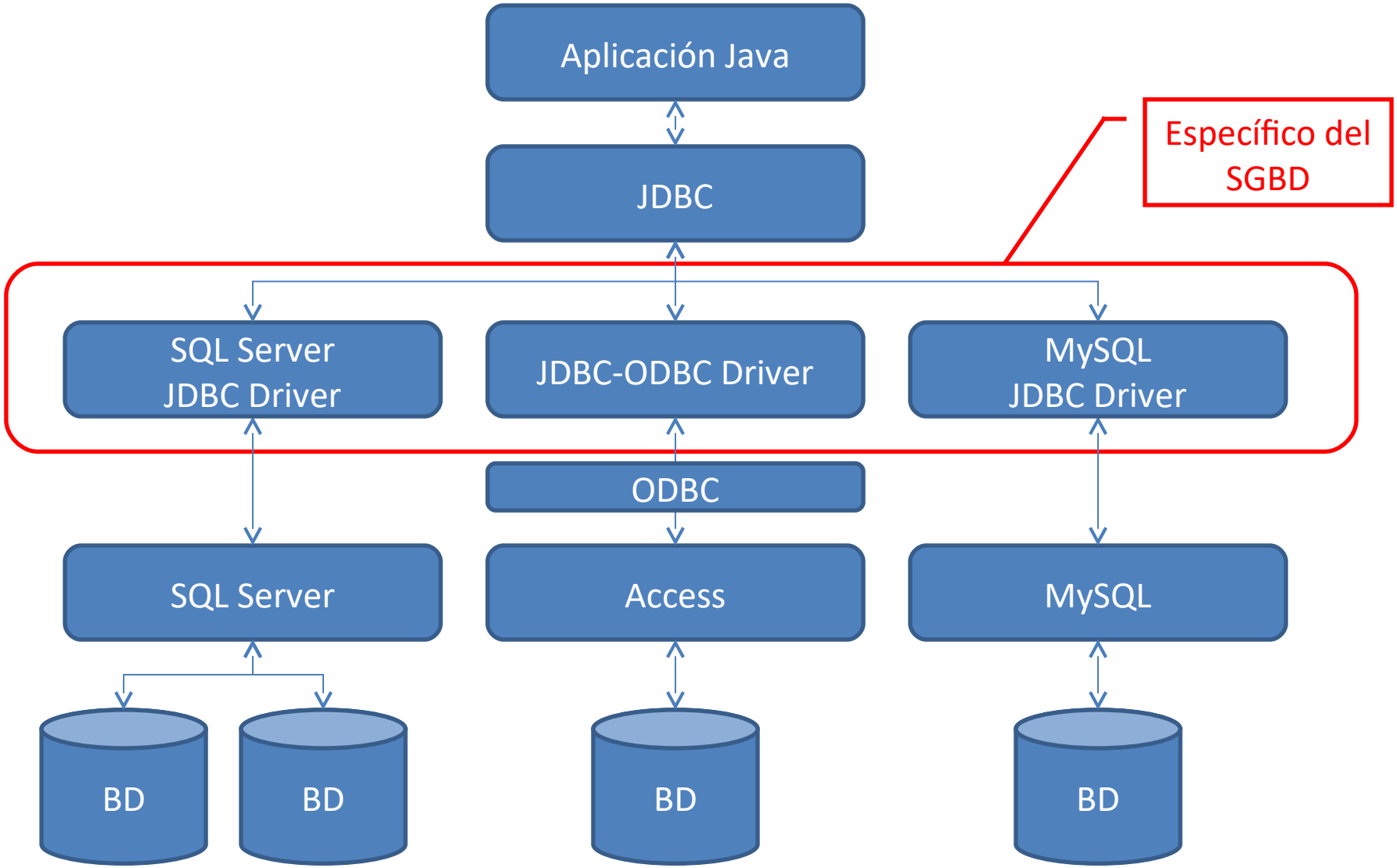


- MS Access
- SQL Server
- Oracle
- MySQL
- Etc.

JDBC

- JDBC = Java Data Base Connectivity
- API Java que proporciona un conjunto de clases que permiten interactuar con bases de datos relacionales
- Para cada SGBD hace falta un driver (controlador) específico
- Es necesario establecer una conexión mediante el driver, que se encarga de la comunicación (traducción) con el SGBD

JDBC


Aplicación Java
JDBC
Específico del SGBD
SQL Server JDBC Driver
JDBC-ODBC Driver
MySQL JDBC Driver
ODBC
SQL Server
Access
MySQL
BD
BD
BD
BD

API JDBC

- Paquetes: java.sql y javax.sql
- Clases fundamentales:
  - **Driver**: permite conectar con la base datos. Cada SGBD debe proporcionar el suyo.
  - **DriverManager:** permite controlar los drivers instalados.
  - **Connection**: representa la conexión con la BD.
  - **DatabaseMetaData**: proporciona información sobre una BD (nombre, tablas, nº máximo de conexiones,…).
  - **Statement**: Para ejecutar sentencias SQL sin parámetros.
  - **PreparedStatement**: Para ejecutar sentencias SQL con parámetros.
  - **CallableStatement**: Para ejecutar procedimientos almacenados.
  - **ResultSet**: Conjunto de resultados obtenidos tras realizar una consulta.
  - **ResultSetMetaData**: Información sobre un conjunto de resultados (número y tipo de columnas, número de registros,…).

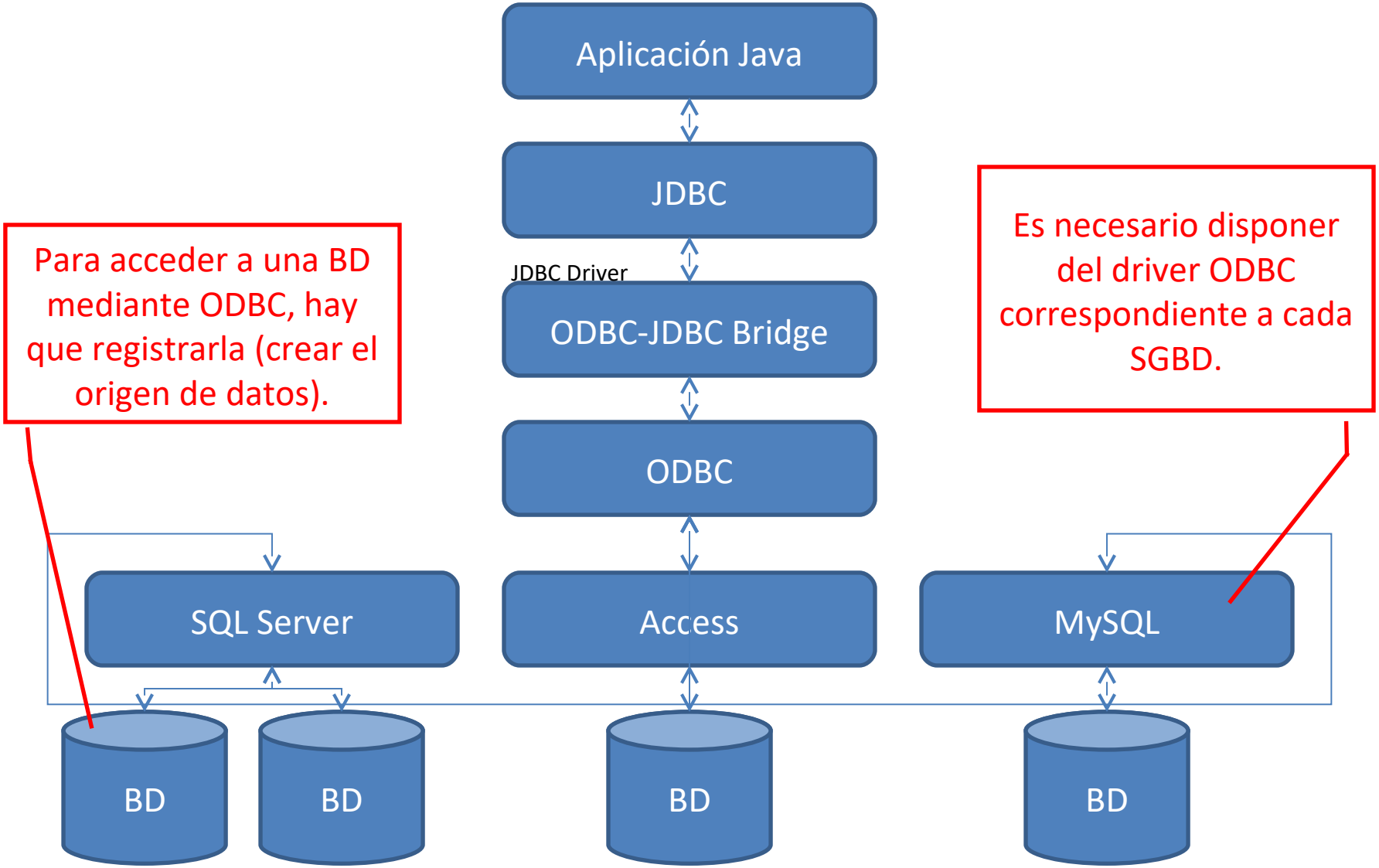
Drivers JDBC

- Cada fabricante de SGBD proporciona su “driver” específico, aunque también los hay de terceros
- El driver es una clase concreta que “sabe” como comunicarse con un SGBD.
- Algunos drivers:
  - MySql:
    - http://dev.mysql.com/downloads/connector/j/
  - MS SQL Server
    - http://www.microsoft.com/downloads/es-es/details.aspx?familyid=a737000d-68d0-4531-b65d-da0f2a735707&displaylang=es
  - Oracle
    - http://www.oracle.com/technetwork/database/enterprise-edition/jdbc-112010-090769.html

Controlador puente JDBC-ODBC

- Los drivers JDBC se proporcionan en forma de librerías Java (JAR = Java ARchive)
- El puente JDBC-ODBC viene por defecto con la JVM en Windows:
  - Paquete “sun.jdbc.odbc”
- ODBC = Object DataBase Connectivity
  - Estándar que pretende que el acceso a cualquier base de datos sea independiente del lenguaje de programación utilizado
  - Se encuentra implementado en los sistemas Windows

Puente JDBC-ODBC



Los pasos para consultar una BD en Java

1. **Cargar la clase** del controlador de la base de datos
2. **Establecer la conexión** a la base de datos. asignamos el tipo y nombre del controlador (driver), la ruta de la BD, el usuario y contraseña.
3. **Crear un objeto Statement** o **PreparedStatment** para consultar la base de datos.
4. **Ejecutar y Crear un objeto ResultSet**, que guardará el conjunto de resultados provenientes de la consulta
5. **Recorrer el objeto ResulSet**, que contiene el resultado de la consulta:
6. **Cerrar la conexión**.

Los pasos para consultar una BD en Java

1. **Cargar la clase**

```
Class.forName("com.mysql.cj.jdbc.Driver");
```

2. **Establecer la conexión**

```
Connection con =
    DriverManager.getConnection( url,usr,pswd);
```

URL: jdbc:mysql://<host>:<puerto>/<basededatos>

```
Connection con = DriverManager.getConnection(
"jdbc:mysql://localhost:3306/prueba","root","pass");
```

Los pasos para consultar una BD en Java

3. **Crear un objeto Statement** o **PreparedStatment**

El objeto **PreparedStatement** contiene una sentencia SQL que ha sido precompilada. Por lo que se puede ejecutar sin tener que compilarla. Se pondrá un **?** Por cada parámetro.

```
PreparedStatement actualiza =
    conecta.prepareStatement( "UPDATE dept SET
    dname =? WHERE deptno LIKE ?");
```

Los pasos para consultar una BD en Java

3. Asignación de parámetros a la sentencia SQL.

Utilizaremos los métodos definidos en la clase PreparedStatement

setXXX (numero de parametro, valor)

```
actualiza.setString(1, "VENTAS");
actualiza.setInt(2, 10);
```

Los pasos para consultar una BD en Java

4. **Ejecutar y Crear un objeto ResultSet**,

Utilizaremos los métodos

- **executeQuery( )** se usa con sentencias de recuperación
- **execute()** para consultas de recuperación o modificación. Devuelve verdadero o falso dependiendo si devuelve un conjunto de registros o el recuento de la operacion
- **executeUpdate( )** consultas de modificación.

ResultSet resultado = consulta.executeQuery();

actualiza.executeUpdate() //devuelve un entero con el número de registros afectados

Los pasos para consultar una BD en Java

5. **Recorrer el objeto ResulSet** (en el caso de consultas que devuelven registros)

```
while (resultado.next())  {
   int cod = resultado.getInt("deptno");
   String nom = resultado.getString("dname");
   System.out.println(cod + " " + nom);
}
// siendo deptno y dname los nombres de los campos en la tabla. También podemos acceder por las posiciones de los campos en el select
```

Los pasos para consultar una BD en Java

6. **Cerrar la conexión**.

```
coneccion.close( );
```

Transacciones

- Una **transacción** es un conjunto de una o más sentencias que se ejecutan como una unidad.
- Cuando se crea una conexión, está en modo **auto-commit.** Cada sentencia SQL individual es tratada como una transacción
- Para permitir que dos o más sentencias sean agrupadas en una transacción hay que desactivar el modo auto-commit.

conecta.setAutoCommit(false);

Situaremos aquí todas las instrucciones de la transacción, no se entregará ninguna sentencia SQL hasta que llamemos explícitamente al método **commit.**

conecta.commit();

conecta.setAutoCommit(true); // retornamos el auto-commit a su valor

- Si se produce algun error, si obtenemos una **SQLException,** deberíamos llamar al método rollback para abortar la transacción

conecta.rollback();