2️⃣

CONEXIÓN A BD

1. Introducción: La Interfaz Connection

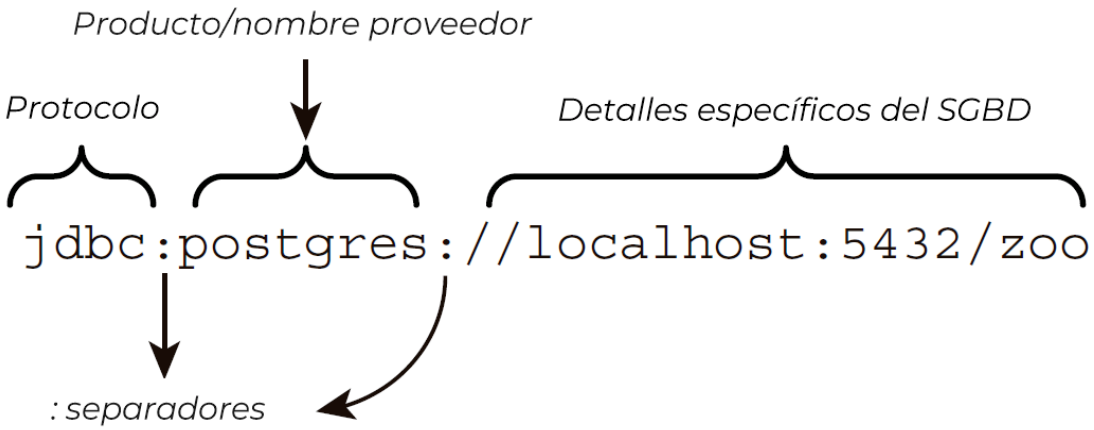
La interfaz `Connection` es el núcleo de las conexiones en JDBC, proporcionando acceso a una fuente de datos como un sistema de gestión de bases de datos (DBMS), un archivo heredado o cualquier otra fuente tabular.

- Permite enviar consultas, manejar transacciones y gestionar metadatos.
- **Clases principales para establecer conexiones:**
    - `DriverManager`: Maneja conexiones de forma directa, recomendada para aplicaciones independientes.
    - `DataSource`: Recomendado en aplicaciones Java EE, oculta detalles de la fuente de datos y soporta características avanzadas como *pooling* de

conexiones.

2. Establecimiento de Conexión con DriverManager

- Método principal: `DriverManager.getConnection`.
- Requiere una **URL de conexión** que devuelve un objeto de tipo Connection, que contiene información como:
  - Dirección del host o servidor.
  - Puerto y nombre de la base de datos.
  - Parámetros adicionales de configuración.



Ejemplo de Conexión Genérica

```
public Connection getConnection() throws SQLException {
    Properties props = new Properties();
    props.put("user", "root");
    props.put("password", "admin");
    return DriverManager.getConnection("jdbc:mysql://localhost:3306/miBa
se", props);
}
```

Componentes Clave de `DriverManager.getConnection`

1. **Propiedades de Conexión**:

- Se pueden proporcionar como un `Properties` con atributos como `user` y `password`.

2. **Objeto Connection**:
   - Representa la conexión activa con la base de datos.
   - Se utiliza para enviar consultas o administrar transacciones.

---

3. Formato de URL de Conexión por SGBD

Las URLs de conexión varían según el sistema gestor de bases de datos. Ejemplos comunes:

SGBD	Driver	URL
MySQL	`com.mysql.cj.jdbc.Driver`	`jdbc:mysql://localhost:3306/miBase`
MariaDB	`org.mariadb.jdbc.Driver`	`jdbc:mariadb://localhost:3306/miBase`
SQLite	`org.sqlite.JDBC`	`jdbc:sqlite:rutaArchivo`
H2 Database	`org.h2.Driver`	`jdbc:h2:~/test`
Oracle	`oracle.jdbc.driver.OracleDriver`	`jdbc:oracle:thin:@host:port:baseDatos`
HSQLDB	`org.hsqldb.jdbc.JDBCDriver`	`jdbc:hsqldb:file:rutaDirectorio`
Derby	`org.apache.derby.jdbc.ClientDriver`	`jdbc:derby:testdb;create=true`

---

4. Ejemplo de Uso Completo

Conexión a MySQL:

```
import java.sql.*;
import java.util.Properties;

public class ConexionBD {
    public static void main(String[] args) {
        String url = "jdbc:mysql://localhost:3306/miBase";
        Properties props = new Properties();
        props.put("user", "root");
        props.put("password", "admin");

        try (Connection conn = DriverManager.getConnection(url, props)) {
            System.out.println("Conexión establecida.");
```

```
        } catch (SQLException e) {
            e.printStackTrace();
        }
    }
}
```

---

5. Detalles Específicos de URLs de Conexión

1. **Java DB (Derby):**
   - Formato: `jdbc:derby:[subsubprotocol:][databaseName][;attribute=value]*`
   - Parámetros comunes:
     - `create=true`: Crea la base de datos si no existe.
     - `encryption`: Habilita la encriptación.
     - `logDevice`: Directorio de logs.
2. **MySQL Connector/J:**
   - Formato: `jdbc:mysql://[host][:port]/[database][?param1=val1&param2=val2]`
   - Soporta:
     - `failover`: Conexiones de respaldo.
     - Propiedades adicionales como:
       - `useSSL`: Para habilitar SSL.
       - `serverTimezone`: Configura la zona horaria del servidor.
3. **MariaDB:**
   - Compatible con MySQL desde la versión 5.5.3.
   - Ejemplo:

```
Connection connection = DriverManager.getConnection("jdbc:mariadb
```

6. Nota sobre Drivers JDBC

- Desde JDBC 4.0, los controladores se cargan automáticamente si están en el classpath.

- Para versiones previas, era necesario cargar manualmente:

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

Conclusión

El manejo de conexiones mediante `DriverManager` y las URLs específicas de cada SGBD proporcionan una forma flexible de conectar aplicaciones Java a diversas fuentes de datos. Si bien `DataSource` es preferido en entornos empresariales, `DriverManager` sigue siendo la opción más sencilla y común para proyectos pequeños y medianos.