

Tema 2. Conectores.

Contenido

Tema 2. Conectores.....	1
Introducción.....	1
Conceptos claves:.....	2
Desfase objeto-relacional	2
Arquitectura JDBC.	3
Conectores o drivers	3
Operaciones básicas JDBC	4
Conectar con la base de datos.....	5
Acceder a los resultados de una consulta.	5

Introducción

Actualmente, las bases de datos relacionales siguen siendo uno de los sistemas de almacenamiento más extendidos, aunque otros sistemas se estén abriendo paso poco a poco.

Una base de datos relacional se puede definir, de una manera simple, como aquella que presenta la información en tablas con filas y columnas.

Una tabla o relación es una colección de objetos del mismo tipo (filas o tuplas).

En cada tabla de una base de datos se elige una clave primaria para identificar de manera unívoca a cada fila.

El sistema gestor de bases de datos Database Management System (DBMS) gestiona el modo en que los datos se almacenan, mantienen y recuperan.

JDBC (Java Database Connectivity), permite simplificar el acceso a bases de datos relacionales, proporcionando un lenguaje mediante el cual las aplicaciones pueden comunicarse con motores de bases de datos. Sun desarrolló este API para el acceso a bases de datos, con tres objetivos principales en mente:

- Ser un API con soporte de SQL: poder construir sentencias SQL e insertarlas dentro de llamadas al API de Java.
- Aprovechar la experiencia de los API's de bases de datos existentes.
- Ser lo más sencillo posible.

Conceptos claves:

Conector: API que permite a los programas de aplicación trabajar con bases de datos.

Desfase objeto-relacional: Los problemas que supone almacenar objetos en bases de datos relacionales, con estructuras basadas en tablas.

Driver JDBC: Biblioteca de software que proporciona una implementación para una base de datos particular de las interfaces definidas en la especificación JDBC, de forma que la permite interactuar con la base de datos.

Iterador: Mecanismo para acceder a los resultados de una consulta, con operaciones para navegar por los resultados de la consulta y recuperar los datos uno a uno.

Pool de conexiones: Conjunto de conexiones de una base de datos que se mantienen abiertas y a disposición de los procesos que puedan necesitarlas, lo que evita el retraso y la sobrecarga que supone la apertura de una nueva conexión cada vez que un proceso la necesita.

Desfase objeto-relacional

Las bd relacionales no están preparadas para almacenar objetos.

Existe un desfase entre las construcciones usadas en las bd relacionales con los proporcionados en programación. Al guardar Objetos en una bd relacional se incrementa la complejidad del código para salvar la diferencia entre objetos y tablas.

El desfase objeto-relacional consiste en la diferencia de aspectos que existen entre la programación orientada a objetos y la base de datos. Estos aspectos se pueden presentar en cuestiones como:

- Lenguaje de programación: el programador debe conocer:
 - El lenguaje de programación orientado a objetos (POO).
 - El lenguaje de acceso a datos.
- Tipos de datos:
 - En las bases de datos relacionales siempre hay restricciones en el uso de tipos.
 - Mientras que la programación orientada a objetos utiliza tipos de datos más complejos.
- Paradigma de programación: en el proceso de diseño y construcción del software se tiene que hacer una traducción del modelo orientado a objetos de clases al modelo Entidad-Relación (E/R) puesto que el primero maneja objetos y el segundo maneja tablas y tuplas (o filas), lo que implica que se tengan que diseñar dos diagramas diferentes para el diseño de la aplicación.

Definición:

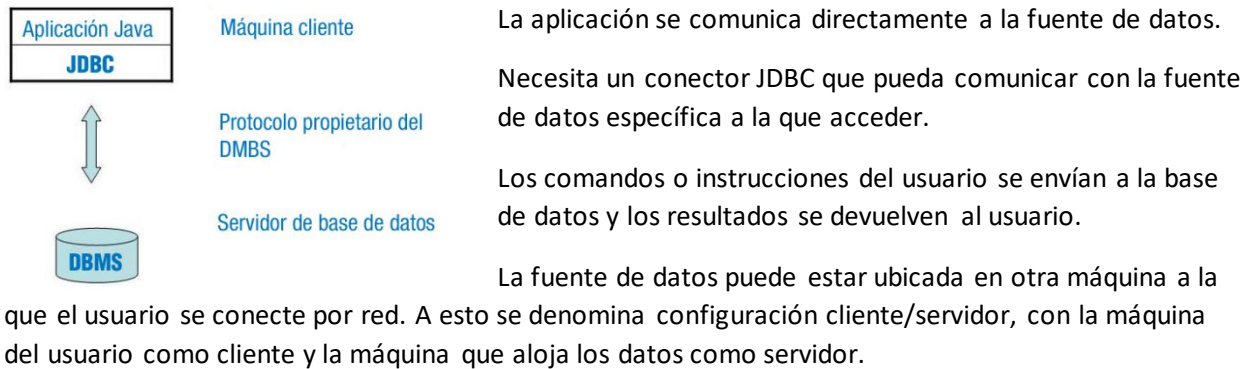
Definición: problemas que ocurren debido a las diferencias entre el modelo de datos en la BD y el lenguaje de POO.

A pesar de esto, ambos paradigmas pueden ser compatibles. Cada vez que se deben extraer o almacenar en una bd relacional algún objeto, se necesita realizar un mapeo. Esto lo veremos más adelante.

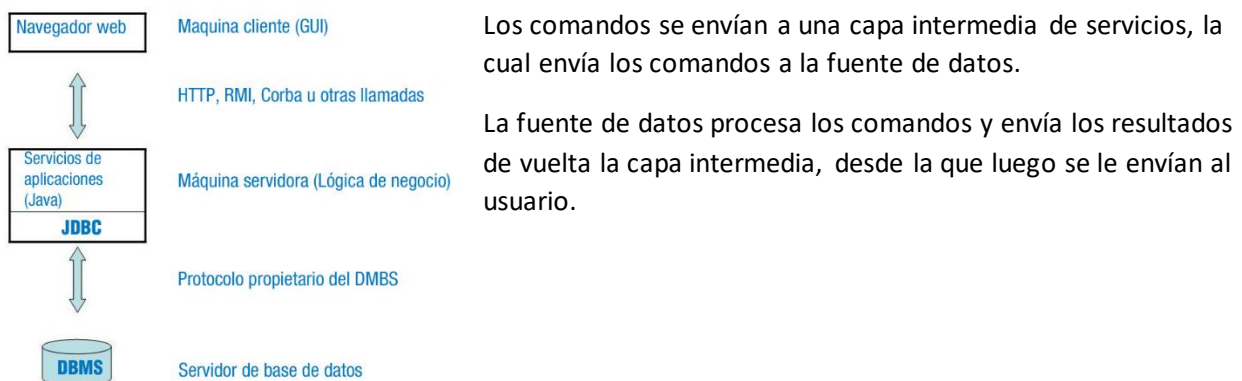
Arquitectura JDBC.

El API JDBC soporta dos modelos de procesamiento para acceso a bases de datos: de dos y tres capas.

Modelo de dos capas:



Modelo de tres capas:



El API JDBC viene distribuido en dos paquetes:

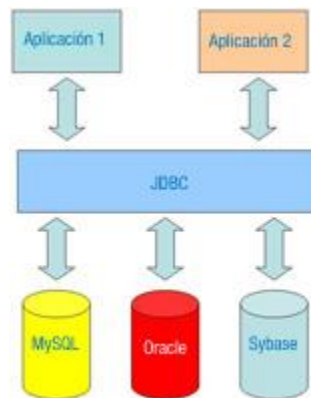
- java.sql, dentro de J2SE.
- javax.sql, extensión dentro de J2EE.

Conectores o drivers

Un conector o driver es un conjunto de clases encargadas de implementar los interfaces del API y acceder a la base de datos.

Un conector suele ser un fichero .jar que contiene una implementación de todas las interfaces del API JDBC.

El conector lo proporciona el fabricante de la base de datos o bien un tercero.



El código de nuestra aplicación no depende del driver, puesto que trabajamos mediante los paquetes `java.sql` y `javax.sql`.

JDBC ofrece las clases e interfaces para:

- Establecer una conexión a una base de datos.
- Ejecutar una consulta.
- Procesar los resultados.

Hay cuatro tipos de drivers JDBC: Tipo 1, Tipo 2, Tipo 3 y Tipo 4

ACTIVIDAD DE BÚSQUEDA: Busca en Internet información sobre los 4 tipos de conectores JDBC.

Operaciones básicas JDBC

Los sublenguajes SQL son:

- DML: para manipular los datos
- DDL: Para la definición de los datos.

Si pensamos en DML, sabemos que podemos diferenciar entre operaciones de:

- Consulta: `SELECT`, que se ejecutan con `executeQuery()` y nos devuelve una lista de filas en un `ResultSet`, sobre el que *iteraremos* para obtener los resultados de uno en uno.
- Modificación de datos: `UPDATE`, `INSERT`, `DELETE`, se ejecutan sobre `executeUpdate`, que devuelve el número de filas afectadas por la operación.

Consultas **DDL**: se ejecutan con `execute()`

Crear una conexión	<code>Connection con = DriverManager.getConnection()</code>
Crear una sentencia	<code>Statement sentencia = con.createStatement()</code>
Ejecutar sentencia <code>SELECT</code>	<pre> ResultSet rs = sentencia.executeQuery(sql) while (rs.next()){ } </pre>

	rs.close
Ejecutar sentencia UPDATE, INSERT, DELETE	ResultSet rs = s.execute()
sentencia. Close con.close	Cerrar sentencia Cerrar conexión

Actividad práctica 1: Crea una conexión jdbc con la base de datos.

Paso 1: Busca en el sitio oficial el conector JDBC adecuado para la conexión con MySQL.

Paso 2: Crea una conexión con la base de datos ies.sql

Conectar con la base de datos.

Actividad de búsqueda y reflexión:

1. Parámetros necesarios
2. Clases
3. Métodos
4. Sintaxis

Acceder a los resultados de una consulta.

No podemos perder de vista que la consulta devuelve registros y el lenguaje maneja objetos.

Los conectores nos permiten acceder a una base de datos relacional para hacer operaciones sobre ella.

Sabemos que una consulta SQL devuelve un conjunto de filas o registros. Los conectores nos permiten recuperar esos resultados fila a fila mediante un objeto que actúa como iterador o cursos.