JDBC API (Java Database Connectivity API)

La API de JDBC provee acceso a datos desde Java. Usando esta API podemos acceder a variadas fuentes de datos: bases de datos relacionales, hojas de cálculo (spreadsheets) y archivos planos. Demos una mirada a qué son las bases de datos antes de revisar cómo las manejamos desde Java.

Introducción a Bases de Datos

Una base de datos es una colección integrada de registros o archivos lógicamente relacionados consolidados en una unidad que provee datos para usos múltiples.
Para conseguir este objetivo se han propuesto varias estructuras o formatos de base de datos. A esto se le llama el modelo de la bases de datos. Entre ellos están el modelo plano, modelo jerárquico, modelo de red, modelo relacional, ver Figura 1. En adelante nos concentraremos sólo en las bases de datos con estructura relacional.

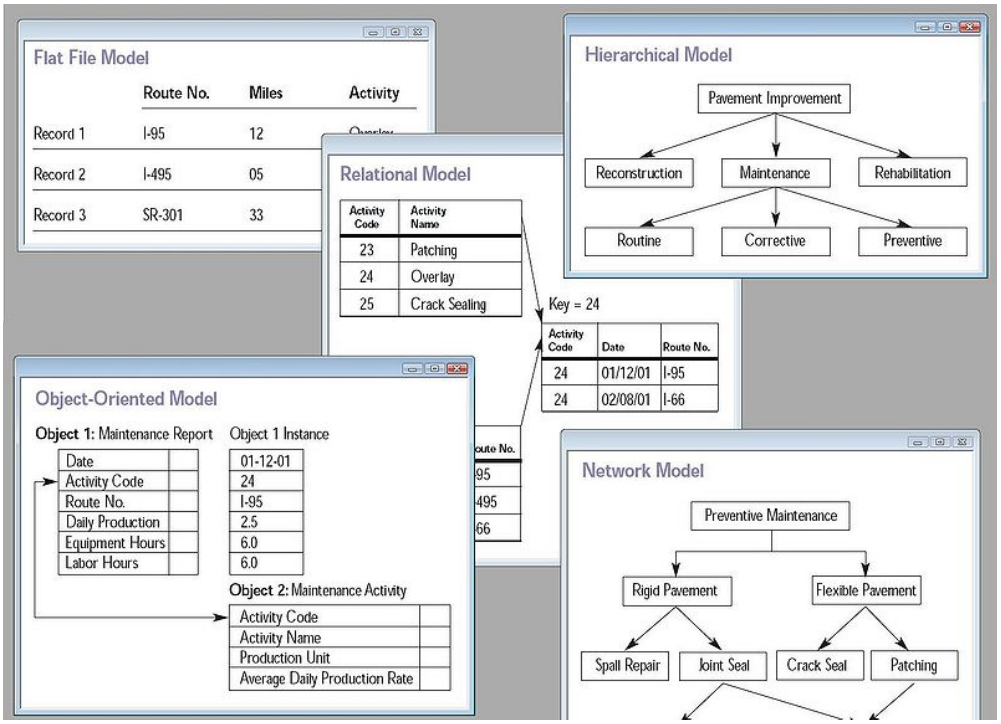

Figura 1: Modelos de Bases de Datos (tomado de wikipedia)

Las bases de datos relacionales están compuestas por **relaciones** (en el sentido del álgebra o del cálculo relacional). Cada relación es representada y almacenada por una **tablas**. Los **datos** son representados como **n-tupas**, que pertenecen al producto Carteciano de n dominios. La tabla almacena estas **n-tuplas,** también llamados **registros**, las cuales corresponden a las **filas de la tabla**. Las **columnas de la tabla** son referidas como los **atributos**. Todos los valores de una columna son del mismo tipo, por ejemplo entero, string, fecha. Cada tupla, o fila, tiene así un valor de un tipo definido para cada atributo, o columna.

La base de datos es el conjunto de datos organizados en algún medio, otra cosa son los programas que nos permiten su creación, control y acceso. Este conjunto de programas se conoce como Sistema Administrador de la Base de Datos (DBMS, Database Management System). Cuando este sistema maneja una base de datos relacional se habla de un RDBMS por Sistema Administrador de Base de Datos Relacional. Uno de tales sistemas es MySQL (licencia GPL, General Public License). Otro RDBMS es PostgreSQL (licencia BSD).

Con licencia GPL podemos usar y/o alterar el original con la condición que el resultado sea hecho disponible en los mismos términos. Esto se resumen bajo el término copyleft (dejar copiar), en contraste con copyright. Con licencia BSD es código abierto y libre.

La manipulación de la base de datos relacional, a través del Sistema Administrador de la Base de Datos, es hecha por medio de consultas o peticiones formuladas en un lenguaje de consultas. Un lenguajes común para manejar bases de datos relacionales es **SQL** (**Structured Query Language**).

Figura 2: Relación entre SQL, RDBMS y DB

Propiedades de las Relaciones

1. Cada relación (o tabla) tiene un nombre distinto.
2. Los valores de los atributos son atómicos.
3. No hay dos atributos con el mismo nombre en una misma relación.
4. El orden de los atributos no importa.
5. Cada tupla es distinta de las demás.
6. El orden de las tuplas no importa.

Para manipular relaciones, es necesario identificar el conjunto de atributos que consiguen identificar en forma unívoca una tupla.
Definiciones:
Clave Candidata es el conjunto de atributos de una tabla tales que: permiten identificar en forma unívoca a una tupla y no podemos eliminar un atributo y seguir cumpliendo unicidad.
Clave Primaria es una clave candidata que ha sido seleccionada para identificar unívocamente las los registros de una tabla.
Clave Foránea: es un atributo o conjunto de éstos que es clave primaria en otra tabla.
Estos conceptos son ilustrados en la figura 3.

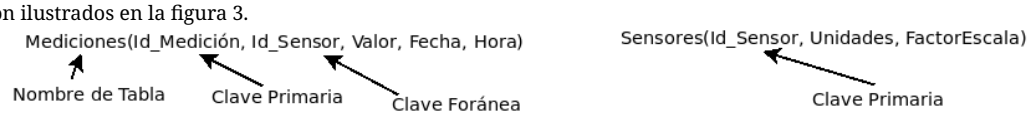

Figura 3: Ejemplo de términos usados en Bases de datos Relacionales

De la Figura 2 se desprende la necesidad de conocer el lenguaje SQL para generar soluciones que ocupen bases de datos. Ciertamente no es lo único, para decidir qué relaciones (tablas) crear y con qué atributos, es necesario hacer un análisis de los datos que participan en una aplicación y cómo estos datos están relacionados. Para esto existen técnicas de modelado de datos que conducen al **modelo de datos**. Uno de tales modelos es el **modelo de entidad y relación** en cual es usado para generar la base de datos relacional.

Para focalizarnos en el acceso a las bases de datos desde Java, este material no cubre SQL ni las técnicas que conducen a **bases de datos normalizadas**. La normalización de una base de datos es la manera sistemática par asegurar que la estructura de una base de datos permita consultas de propósito general y sea libre de características que puedan conducir a pérdida de integridad de los datos.

JDBC API

Este paquete permite conectarse a una base de datos, consultarla o actualizarla usando SQL. Su manejo es de importancia debido a la frecuencia con que las bases de datos son usadas hoy.
Así como con Java se logra independencia de la plataforma, al trabajar con JDBC se logra además independencia del proveedor de la base de datos.
Una dificultad enfrentada por los desarrolladores de JDBC fue que existen muchos proveedores de bases de datos cada uno usando su protocolo propio. Es así como se acordó el desarrollo de una API Java para SQL, la cual accede la base de datos vía un administrador de drivers de terceros los cuales se conectan a bases de datos específicas. Cada proveedor de bases de datos debía generar su propio driver conectable al administrador de drivers.
JDBC sigue un modelo similar al de ODBC (Open Database Connectivity) de Microsoft. ODBC es una API estándar para conectarse a bases de datos. Por esto una opción común es conectar JDBC a través de un driver que actúa como puente entre JDBC y ODBC, ver Figura 4.

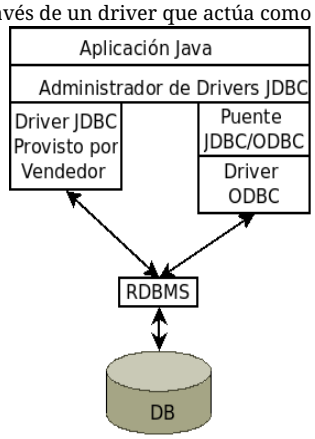

Figura 4: Administrador de Drivers

Instalación de Base de Datos de Prueba

Usaremos MySQL para probar la conectividad desde Java. Cambiando el driver y el punto de conexión con la base de datos basta para acceder a otras bases de datos compatibles con SQL.
Un sistema de desarrollo simple es XAMPP. Luego de bajarlo y seguir sus instrucciones, es simple crear bases de datos para probar nuestros desarrollos. Con XAMPP tenemos una base de datos relacional MySQL en nuestro computador y su DBMS.
Una vez instalada podemos iniciarla con:

```
agustin@agustin-DELL:/opt/lampp$ sudo ./lampp start
Starting XAMPP for Linux 1.7.2...
XAMPP: Starting Apache with SSL (and PHP5)...
XAMPP: Starting MySQL...
XAMPP: Starting ProFTPD...
XAMPP for Linux started.
```

Con su navegador dé una mirada a http/localhost/ y use XAMPP para crear una base de datos. Luego cree en ella alguna tabla con algunos registros en ella.

Instalación del Driver JDBC

MySQL ha desarrollado un driver compatible con el administrador de drivers de Java. Este driver permite la conexión de Java con bases de datos MySQL. Para su instalación, puede ver las instrucciones aquí.
Para que Java tenga acceso al driver podemos incluir su directorio en la variable CLASSPATH, podemos usar la opción -cp al ejecutar el programa, o podemos instalarlo en un directorio de Java que ya sea revisado por la máquina virtual; por ejemplo, bajo directorio ...../jre/lib/ext/

Con MySQL instalado y el driver JDBC instalado podemos ejercitar la conexión de Java con su base de datos.

Aplicaciones Java con Base de Datos MySQL

Entendido lo anterior, todo programa Java que desea conectar a una base de datos necesita cargar el driver específico para la base de datos a usar.
Para esto se usa:

```
try {  // Carga el driver JDBC
  Class.forName("com.mysql.jdbc.Driver");
} catch( Exception e ) {
  System.out.println( "No se pudo cargar el Drive." );
  return;
}
```

Luego su programa puede conectarse con la base de datos específica, la cual debe estar previamente creada. Para esto se usa:

```
Connection conexion; // Aloja una conexión o sesión con la base de datos para interactuar con ella.
Statement sentencia; // A partir de la conexión podemos crear objetos sentencias que permiten hacer consultas SQL
try {
   conexion = DriverManager.getConnection( "jdbc:mysql://localhost/ELO330","root","");  // Aquí ELO330 es la Base de Datos que debe existir previamente
   sentencia = conexion.createStatement();
 :
} catch( SQLException e ) {
  System.out.println("Error en la operacion" + e.getMessage());
}
```

En general los pasos para hacer consultas SQL en JDBC de debe:

1. Establecer la conexión con la base de datos.
2. Crear un objeto sentencia.
3. Ejecutar consultas (query) usando el objeto sentencia.
4. Para cada consulta procesarel objeto `ResultSet` retornado por la consulta.
5. Cerrar la conexión.

Dar una mirada a las interfaces Connection y Statement en la documentación Java.

Ejemplos:
Creación y llenado de una tabla, para su compilación y ejecución revise el archivo Makefile.
Consultas de Clientes de Banco, para su compilación y ejecución revise el archivo Makefile.
Pacientes, para su compilación y ejecución revise el archivo Makefile.