#### Universidad Central de Venezuela

#### Facultad de Ciencias

#### Escuela de Computación

#### Administración de Bases de datos

#### Ejemplo de conexión con Java

El objetivo es conectarse a una base de datos utilizando el lenguaje de programación Java, para lo cual se deben realizar dos pasos:

1. Registrar (cargar en memoria) el controlador. Esto se puede hacer de dos formas:
   * De forma dinámica, por medio del método **Class.forName( String de driver )**. Esta es la forma más usual.
   * Ejemplos de carga dinámica:
     + En MySQL local: Class.forName("com.mysql.jdbc.Driver");
     + En Oracle: Class.forName("oracle.jdbc.Driver.OracleDriver");
   * De manera estática, usando **System.setProperties("jdbc.drivers", String de driver)**.
2. Establecer la conexión por medio del método **DriverManager.getConnection( String de la URL de base de datos, String de login, String de password )**.

La conexión usa una sintaxis semejante a la de una URL. La sintaxis de la URL (los corchetes indican elementos opcionales) es la siguiente:

jdbc:Controlador://[host][,failoverhost...][:port]/[database][?propertyName1][=propertyValue1][&propertyName2][=propertyValue2]...

Notas a la sintaxis de la URL:

* + Controlador: controlador de base de datos, en nuestro ejemplo es **Oracle**.
  + Host: servidor de base de datos (por defecto es localhost).
  + Failoverhost: servidor de respaldo, en caso de que falle el primero
  + Port: si no se indica puerto, se toma el valor por defecto. En el caso de Oracle es 1521.
  + Database:nombre de la base de datos. Si no se indica base de datos, asume la base de datos actual. Por medio del método **setCatalog** de la clase **Connection** se puede determinar la base de datos actual.

Ejemplos de conexión:

* + En MySQL local:

Connection con = DriverManager.getConnection("jdbc:mysql://localhost/BD", "usuario", "contraseña" );

* + En Oracle:

Connection con = DriverManager.getConnection( "jdbc:oracle:thin:@127.0.0.1:1521:XE", "usuario", "contraseña" );

#### Un sencillo ejemplo con registro dinámico

A continuación el primer ejemplo con registro dinámico del controlador (driver) y la conexión mediante **DriverManager.getConnection**; en la llamada a este método, el segundo y tercer argumento son el login y el password. Contiene comentarios detallados:

package paquete;  
  
 import java.sql.DriverManager;  
 import java.sql.Connection;  
 import java.lang.ClassNotFoundException;  
 import java.sql.\*;  
  
 public class Main {  
 public static void main(String[] args) {  
 try {  
  
 /\*\*\*\* Cargamos el driver \*\*\*\*/  
 Class.forName("oracle.jdbc.driver.OracleDriver");  
  
 /\*\*\*\* Realizamos la conexión \*\*\*\*/  
 Connection con = DriverManager.getConnection

("jdbc:oracle:thin:@127.0.0.1:1521:xe", "**abd01**", "**abd01**" ); /\*\*Colocar tu usuario\*\*/

/\*\*\*\* avisamos \*\*\*\*/  
 System.out.println( "Si he llegado hasta aquí es que se ha producido la conexión");  
 System.out.println( "Si no se hubiera producido, se habría disparado SQLException");

/\*Para realizar una consulta\*/

Statement stmt = con.createStatement();

ResultSet rset = stmt.executeQuery("select \* from producto order by precio ASC");

while (rset.next()){

System.out.println(rset.getString(1)+ " " + rset.getString(2));

//Aquí se imprime por pantalla cada producto

}

/\*Para actualizar un campo. \*/

String nombre\_elegido = "Mesa"; //Este es el nuevo nombre del producto.

String cod= "C-03";

Integer precio = 150;

String desc = "Bella mesa para comer";

stmt.executeUpdate("update producto set nombre = '"+nombre\_elegido+"', precio = "+precio+", descripcion = '"+desc+"' where codigo = '"+cod+"'");

stmt.close();

/\*\*\*\* Una buena costumbre: cerramos la conexión \*\*\*\*/  
 con.**close**();  
 }  
 /\*\*\*\* Excepción que se dispara si falla la carga del driver \*\*\*\*/  
 catch( **ClassNotFoundException** e ) { e.printStackTrace(); }  
  
 /\*\*\*\* Excepción que se dispara si falla la conexión \*\*\*\*\*/  
 catch ( **SQLException** e) { e.printStackTrace(); }  
  
 }  
 }

Donde:

* **Class.forName**: si esta llamada falla (normalmente porque no se encuentra el archivo del controlador), entonces se dispara una excepción **ClassNotFoundException**.
* De manera semejante, si el intento de conexión mediante **DriverManager.getConnection** ha fallado se dispara la excepción **SQLException**. La gestión de esta excepción no sólo es necesaria en la conexión a la base de datos, sino en otras circunstancias: definición de datos, manipulación de datos, consulta, etc.
* Por último: cerramos la conexión.

El resultado final por pantalla:

**Al ejecutarlo una vez:**

Si he llegado hasta aquí es que se ha producido la conexión  
Si no se hubiera producido, se habría disparado SQLException

A-01 Vela

C-03 Sofa

B-02 Estante

**Al ejecutarlo de nuevo:**

Si he llegado hasta aquí es que se ha producido la conexión  
Si no se hubiera producido, se habría disparado SQLException

A-01 Vela

C-03 Mesa

B-02 Estante