

# PROGRAMACION ORIENTADA A OBJETOS II

## PRACTICAS DE LABORATORIO

ESTAS PRACTICAS DE LABORATORIO DEBEN SER REALIZADA DE MANERA INDIVIDUAL. LEA DETENIDAMENTE LAS INSTRUCCIONES, PUES EL NO SEGUIRLAS TAL COMO SE INDICA PUEDE RESULTAR EN UNA PRACTICA INCORRECTA. CUANDO LA PRÁCTICA SEA TERMINADA, ASEGURESE DE QUE CUMPLE CON TODOS LOS REQUERIMIENTOS SOLICITADOS Y DE QUE FUNCIONE CORRECTAMENTE ANTES DE ENTREGARLA.

PRACTICA 6. USO DE LA CLASE DriverManager EN JDBC

PRACTICA 7. USO DE LA INTERFACE DataSource EN JDBC

Estas prácticas tienen que ver con la API de JDBC en cuanto a la creación de tablas a través de la clase DriverManager y de la conexión a través de la interface DataSource. Se asume que la práctica 4 fue realizada exitosamente, por lo cual ya tiene instalado MySQL y configurada la variable de entorno CLASSPATH (necesaria solo en caso de que vaya a desarrollar la práctica sin usar IntelliJ IDEA). El desarrollo de estas prácticas puede hacerse usando IntelliJ IDEA, o desde la terminal de comandos si es que no tiene suficientes recursos su computadora para cargar IntelliJ.

Paso 1: Deberá crear la base de datos `controlconcursos_ad2021` y darle permisos a un usuario de nombre `IngSW`, para ello, primero entre a la consola de MySQL tecleando el siguiente comando desde la ventana de comandos del sistema operativo: **`mysql -u root -p`**

y una vez en la consola de MySQL teclee los siguientes comandos:

```
CREATE DATABASE IF NOT EXISTS controlconcursos_ad2021;
DROP USER IF EXISTS 'IngSW'@'localhost';
CREATE USER 'IngSW'@'localhost' IDENTIFIED BY 'UAZsw2021';
GRANT ALL ON controlconcursos_ad2021.* TO 'IngSW'@'localhost';
```

Paso 2. Clone el repositorio usando la técnica que le convenga (ya sea usando IntelliJ IDEA o desde la terminal de comandos usando **`git clone`**). Los detalles están en el archivo README.md

NOTA: Para los siguientes pasos, edite los archivos mencionados usando ya sea IntelliJ IDEA o el editor de su preferencia. Los lugares donde debe hacer cambios están marcados en el código base con un comentario **TODO**, no modifique ninguna otra parte del código.

Paso 3. Edite el archivo ***MainPractica06\_07.java***, modifique la primera línea dentro de la clase, la cual es como la siguiente:

```
public static final String matricula="xxxx";
```

para que el atributo matricula tenga como valor su matrícula propia. **En este archivo es lo único que debe cambiar**. La clase incluida en este archivo la puede ejecutar, **DE MANERA OPCIONAL**, para que el código que implemente en las otras cosas vaya mostrando mensajes de lo que va realizando.

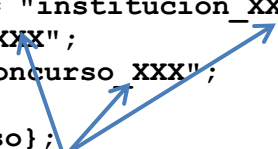
Paso 4. Edite el archivo `CreaTablasConDriverManager.java`, para hacer lo siguiente (**vaya haciendo commits conforme vaya haciendo cada cambio**):

a) Coloca en los comentarios siguientes tu nombre en vez de las NNNNNNNN y tu matricula en vez de las XXXXXXXX:

```
/**
    @author:  NNNNNNNN
    @author:  XXXXXXXX
 */
```

b) Agregue los siguientes atributos dentro de la clase *CreaTablasConDriverManager*:

```
private final String nomInstitucion = "institucion_XXX";
private final String nomSede = "sede_XXX";
private final String nomConcurso = "concurso_XXX";
private final String[] nomtablas =
    { nomInstitucion, nomSede, nomConcurso};
```



NOTA: En los valores de estas últimas constantes deben reemplazar XXX por su matrícula.

c) Coloque dentro el primer constructor de *CreaTablasConDriverManager* (el que recibe 4 strings), el siguiente código, el cual crea un URL a partir del primer y cuarto argumento y usa tal URL en conjunto con el segundo y tercer argumento para crear una conexión a través del método *getConnection* de la clase *DriverManager*:

```
String url = String.format("jdbc:mysql://%s/%s",
    direccionServidor, nomBD);
conn = DriverManager.getConnection(url, usuario, clave);
```

d) Coloque dentro el segundo constructor de *CreaTablasConDriverManager* (el que recibe 1 solo argumento), el siguiente código, el cual asocia el objeto de tipo *Connection* recibido como argumento al atributo *conn* (que representa la conexión a usar en los métodos de esta clase), asegurándose que no sea null:

```
if (conn==null) {
    throw new Exception("Conector recibido es nulo");
}
this.conn = conn;
```

e) Coloque el siguiente código dentro del método *creaTablas()*. Este método creará las tres tablas necesarias (note que en algunas sentencias SQL involucradas en este paso y pasos posteriores hay espacios entre las comillas, estos espacios son necesarios, de otra manera su programa no funcionará)

```
Statement sentencia;
String prefijo="DROP TABLE IF EXISTS ";
String stringEliminacionInstitucion= prefijo + nomInstitucion;
String stringEliminacionSede = prefijo + nomSede;
String stringEliminacionConcurso = prefijo + nomConcurso;

if (debug) {
    System.out.println("Iniciando creacion de tablas ...");
}

sentencia = conn.createStatement();
sentencia.executeUpdate(stringEliminacionInstitucion);
sentencia.executeUpdate(stringEliminacionSede);
sentencia.executeUpdate(stringEliminacionConcurso);
```

```

Scanner lector =
    new Scanner(new File("tablas_ad2021.txt"));
int i=0;
while (lector.hasNext()) {
    String sql = lector.nextLine();
    if (debug) {
        System.out.println("Creando tabla " +
            nomtablas[i]+"..");
    }
    sentencia.executeUpdate(String.format(sql,
        nomtablas[i]));
    i++;
}
sentencia.close();

```

f) Coloque el siguiente código dentro del método `llenaTablaInstitucion()`. Este método insertará datos a la tabla cuyo nombre está en la variable `nomInstitucion` a través del uso de una sentencia preparada que permite obtener el valor de un campo autoincrementable. **NO SE LIMITE A SIMPLEMENTE COPIAR EL CÓDIGO, ANALICELO CUIDADOSAMENTE PUES INCLUYE UNA TÉCNICA QUE PUEDE NECESITAR PARA POSTERIORES PROGRAMAS Y/O PRACTICAS, COPIAR LOS COMENTARIOS ES OPCIONAL, SOLO ESTAN PARA EXPLICAR LO QUE SE ESTA HACIENDO**

```

if (debug) {
    System.out.println("\nColocando datos en tabla " +
        nomInstitucion+":");
}
Statement sentencia=conn.createStatement();
sentencia.executeUpdate("DELETE FROM "+nomInstitucion);

// Se crea una sentencia preparada que permite obtener el valor
// del campo autoincrementable despues de hacer un INSERT
PreparedStatement stmtInsertInst = conn.prepareStatement(
    "INSERT INTO "+nomInstitucion+"(nombre_institucion, "+
    "nombre_corto_institucion, "+
    "url_institucion, calle_num_institucion, " +
    "id_entidad_institucion, "+
    "id_municipio_institucion) "+
    "VALUES(?,?,?,?,?,?,?)",
    Statement.RETURN_GENERATED_KEYS);

BufferedReader lector = new BufferedReader(
    new InputStreamReader(
        new FileInputStream("datosInstitucion.txt")));
String linea=lector.readLine();
while (linea!=null) {
    String[] elems=linea.split(",");
    if (debug) {
        System.out.printf(
            "Agregando a la %s, ",elems[1]);
    }
}

```

```

        stmtInsertInst.setString(1,elems[0]);
        stmtInsertInst.setString(2,elems[1]);
        stmtInsertInst.setString(3,elems[2]);
        stmtInsertInst.setString(4,elems[3]);
        stmtInsertInst.setInt(5,
            Integer.parseInt(elems[4]));
        stmtInsertInst.setInt(6,
            Integer.parseInt(elems[5]));
        stmtInsertInst.executeUpdate();
        int valorLLaveAutoIncremento = -1;
        // Se obtiene el valor del campo autoincrementable
        ResultSet rs = stmtInsertInst.getGeneratedKeys();
        if(rs.next()){
            valorLLaveAutoIncremento = rs.getInt(1);
        }
        if (debug) {
            System.out.println(
                "el id_institucion que le toco fue : "
                + valorLLaveAutoIncremento);
        }
        linea=lector.readLine();
    }

    lector.close();
    stmtInsertInst.close();

```

g) Coloque el siguiente código dentro del método `close()`. Este método cierra la conexión, si es que existe:

```

    if (isConnActive()) {
        conn.close();
    }

```

Paso 5. Ejecute las prueba de esta práctica dando click en *CreaTablasConDriverManagerTest* con el botón derecho y seleccionado Run si está en IntelliJ o emitiendo el comando **gradle test --tests poo2.prac06.CreaTablasConDriverManagerTest** desde la terminal de comando

Paso 6. Edite el archivo `ColocaDatosUsandoDataSource.java`, para hacer lo siguiente (**vaya haciendo commit conforme vaya haciendo cada cambio**):

a) Coloque en los comentarios siguientes tu nombre en vez de las NNNNNNNN y tu matricula en vez de las XXXXXXXX:

```

/**
    @author:  NNNNNNNN
    @author:  XXXXXXXX
*/

```

b) Agregue los siguientes atributos dentro de la clase `ColocaDatosUsandoDataSource`:

```

    private final String nomSede = "sede_XXX";
    private final String nomConcurso = "concurso_XXX";

```

**NOTA:** En el valor de las dos últimas constantes debe reemplazar **XXX** por su matrícula.

c) Coloque dentro del constructor de `ColocaDatosUsandoDataSource` (el que recibe 4 strings), el siguiente código, el cual crea un `DataSource` específico a MySQL, lo configura con los argumentos recibidos e inicializa el objeto `Connection` que representa a la conexión a usarse en los otros métodos de esta clase:

```
MysqlDataSource fuente = new MysqlDataSource();
fuente.setServerName(direccionServidor);
fuente.setUser(usuario);
fuente.setPassword(clave);
fuente.setDatabaseName(nomBD);
conn = fuente.getConnection();
```

d) Coloque el siguiente código dentro del método `llenaTablaSede()`. Este método meterá datos a la tabla cuyo nombre está en la variable `nomSede` y demuestra una de las formas para acceder al valor de un campo que se incrementa de manera automática (`id_sede`). **NO SE LIMITE A SIMPLEMENTE COPIAR EL CÓDIGO, ANALICELO CUIDADOSAMENTE PUES INCLUYE UNA TÉCNICA QUE PUEDE NECESITAR PARA POSTERIORES PROGRAMAS Y/O PRACTICAS, COPIAR LOS COMENTARIOS ES OPCIONAL, SOLO ESTAN PARA EXPLICAR LO QUE SE ESTA HACIENDO:**

```
if (debug) {
    System.out.println("\nColocando datos en tabla "
        + nomSede+":");
}
Statement sentencia=conn.createStatement();
sentencia.executeUpdate("DELETE FROM "+nomSede);
sentencia.executeUpdate("ALTER TABLE "+nomSede+
    " AUTO_INCREMENT=1");
Scanner lector = new Scanner(new File("datosSede.txt"));
lector.useLocale(Locale.US);
//Delimitador de los datos en archivo
lector.useDelimiter("[,\\n\\r]+");
// Creamos una sentencia que permita actualizaciones directas
// en el ResultSet obtenido por la ejecucion de un SELECT
sentencia = conn.createStatement(ResultSet.TYPE_SCROLL_SENSITIVE,
    ResultSet.CONCUR_UPDATABLE);
ResultSet resultado = sentencia.executeQuery("SELECT * FROM "
    + nomSede);

String valor;
while (lector.hasNext()) {
    // Indicamos que queremos usar un registro nuevo
    resultado.moveToInsertRow();
    valor = lector.next();
    if (debug){
        System.out.print("Agregando la sede "+valor);
    }
    resultado.updateString("nombre_sede",valor);
    resultado.updateInt(3,lector.nextInt());
    valor = lector.next();
    resultado.updateString(4,valor);
    // Agregamos el registro a la tabla
    resultado.insertRow();
    // Nos movemos al ultimo registro y obtenemos el valor id_sede
```

```

        resultado.last();
        int idSede=resultado.getInt("id_sede");
        if (debug) {
            System.out.println(", el id_sede que le toco fue : "
                + idSede);
        }
    }
    lector.close();
    sentencia.close();

```

e) Coloque el siguiente código dentro del método `llenaTablaConcurso()`. Este método meterá datos a la tabla cuyo nombre está en la variable `nomConcurso` y demuestra una de las formas para acceder al valor de un campo que se incrementa de manera automática (`id_concurso`). **NO SE LIMITE A SIMPLEMENTE COPIAR EL CÓDIGO, ANALICELO CUIDADOSAMENTE PUES INCLUYE UNA TÉCNICA QUE PUEDE NECESITAR PARA POSTERIORES PROGRAMAS Y/O PRACTICAS, COPIAR LOS COMENTARIOS ES OPCIONAL, SOLO ESTAN PARA EXPLICAR LO QUE SE ESTA HACIENDO:**

```

        if (debug) {
            System.out.println("\nColocando datos en tabla "
                + nomConcurso+":");
        }
        Statement sentencia=conn.createStatement();
        sentencia.executeUpdate("DELETE FROM "+nomConcurso);
        sentencia.executeUpdate("ALTER TABLE "+nomConcurso+
            " AUTO_INCREMENT=1");
        String insertaDatos = "INSERT INTO "+nomConcurso +
            "(nombre_concurso, fecha_concurso, fecha_inicio_registro," +
            " fecha_fin_registro) VALUES(";
        if (debug) {
            System.out.print("Agregando ' Gran Premio ICPC 2021'");
        }
        String sql = insertaDatos+"'Gran Premio ICPC 2021','2021-10-30','+
            "'2021-09-27', '2021-10-27')";
        sentencia.executeUpdate(sql);
        int valorLlaveAutoIncremento = -1;

        // La funcion LAST_INSERT_ID() de MySQL devuelve el valor del
        // campo autoincrementable en el ultimo registro insertado
        ResultSet rs = sentencia.executeQuery("SELECT LAST_INSERT_ID()");
        if (rs.next()) {
            valorLlaveAutoIncremento = rs.getInt(1);
        }

        if (debug) {
            System.out.println(", el id_concurso que le toco fue : "
                + valorLlaveAutoIncremento);
        }
        sentencia.close();

```

f) Coloque el siguiente código dentro del método `close()`. Este método cierra la conexión, si es que existe:

```
if (isConnActive()) {  
    conn.close();  
}
```

Paso 7. Ejecute las pruebas de esta práctica dando click en **ColocaDatosUsandoDataSourceTest** con el botón derecho y seleccionado Run si está en IntelliJ o emitiendo el comando **gradle test --tests poo2.prac07.ColocaDatosUsandoDataSourceTest** desde la terminal de comando

Paso 8 (**OPCIONAL Y NO NECESARIO**). Si desea ver mensajes de lo que va haciendo su código puede ejecutar la clase **MainPractica06\_07** dándole click con el botón derecho y seleccionado Run si está en IntelliJ o emitiendo el comando **gradle run** desde la terminal de comando.

La salida generada por **MainPractica06\_07** debe ser algo como lo siguiente (las XXX representan su matrícula):

```
Iniciando creacion de tablas ...  
Creando tabla institucion_XXX..  
Creando tabla sede_XXX..  
Creando tabla concurso_XXX..
```

```
Colocando datos en tabla institucion_XXX:  
Agregando a la UAZ, el id_institucion que le toco fue : 1  
Agregando a la UPIIZ, el id_institucion que le toco fue : 2  
Agregando a la UTZAC, el id_institucion que le toco fue : 3
```

```
Colocando datos en tabla sede_XXX:  
Agregando la sede UAIE, el id_sede que le toco fue : 1  
Agregando la sede TIC-UTZAC, el id_sede que le toco fue : 2
```

```
Colocando datos en tabla concurso_XXX:  
Agregando 'Gran Premio ICPC 2021', el id_concurso que le toco fue : 1
```

Paso 9. Verifique entrando a la consola de MySQL que la base de datos creada en el Paso 1 contenga las tablas `carrera_XXX` y `periodoescolar_XXX` y que tengan los registros insertados por las sentencias SQL ejecutadas en el método `llenaTablaInstitucion`. (Nuevamente, recuerde que XXX es sustituido por su matrícula)

Paso 10. Haga un `git push` para subir los cambios a su repositorio privado

Recuerda que cada vez que hagas **git push** se realizarán automáticamente pruebas sobre tu programa para verificar si funciona correctamente. Verifica que las pruebas que se hacen no marquen errores. Recuerda que en la página de tu repositorio en la sección **Pull Requests**, se encuentra una subsección de nombre **Feedback**, donde podrás encontrar los resultados de las pruebas en la pestaña denominada **Check** (expandiendo la parte que dice **Run education/autograding@v1**), y cualquier comentario general que el profesor tenga sobre tu código en la pestaña **Conversation**.