

Tarea Clase Acceso a Datos

1. Crear la base de datos con tuplas de ejemplo
2. Crear un procedimiento almacenado a traves de java en la base de datos

En este apartado sencillamente se ha usado la ejecucion de una sentencia SQL normal con un objeto de tipo "Statement"

Codigo:

```
/**
 *
 * @author Javier Gonzalez Rives
 */
public class Procedimiento {
    private static final String conexion =
        "jdbc:mysql://localhost:3306/ejercicio?
user=root&password=admin&useTimezone=true&serverTimezone=GMT&SSL=true";
    private static String procedure =
        "create procedure alumnosSinCombocatoria() begin select al.nombre,asig.nombre from
ejercicio.alumno al inner join ejercicio.asignatura asig inner join ejercicio.matricula mat on mat.dni
= al.dni and mat.AS = asig.AS where mat.nconvocatoria = 4 and mat.nota < 5 ; end";
    /**
     * @param args
     */
    public static void main(String[] args) {

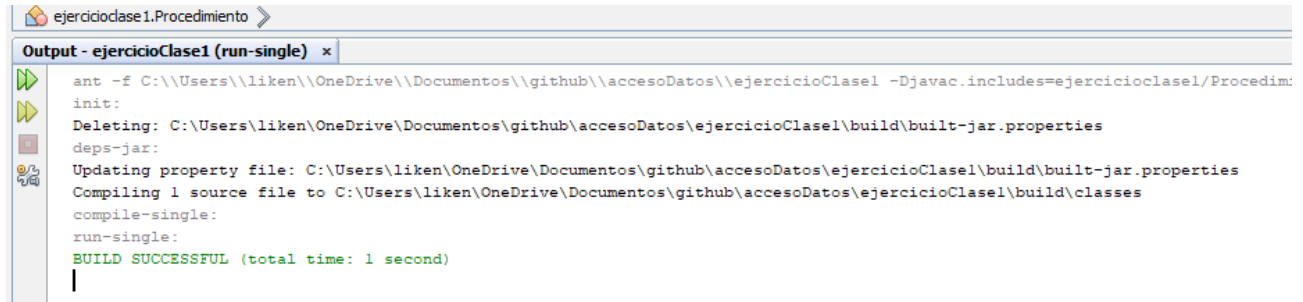
        try{
            Class.forName("com.mysql.cj.jdbc.Driver");
            // creacion de la conexion
            Connection conec = DriverManager.getConnection(conexion);
            Statement sentencia = conec.createStatement();
            sentencia.execute(procedure);

            // cerrado de la conexion
            conec.close();
        } catch(ClassNotFoundException cl){
            System.out.println("error");
        } catch(SQLException sq){
            System.out.println("error sql");
            sq.printStackTrace();
        }
    }
}
```

Javier González Rives

```
    }  
  }  
}
```

} Salida del programa:



```
ejercicioClase1.Procedimiento >>  
Output - ejercicioClase1 (run-single) x  
ant -f C:\Users\liken\OneDrive\Documents\github\accesoDatos\ejercicioClase1 -Djavac.includes=ejercicioClase1/Procedim:  
init:  
Deleting: C:\Users\liken\OneDrive\Documents\github\accesoDatos\ejercicioClase1\build\build-jar.properties  
deps-jar:  
Updating property file: C:\Users\liken\OneDrive\Documents\github\accesoDatos\ejercicioClase1\build\build-jar.properties  
Compiling 1 source file to C:\Users\liken\OneDrive\Documents\github\accesoDatos\ejercicioClase1\build\classes  
compile-single:  
run-single:  
BUILD SUCCESSFUL (total time: 1 second)  
|
```

Procedimiento:

```
create procedure alumnosSinCombocatoria()  
begin  
select al.nombre,asig.nombre from ejercicio.alumno al inner join  
ejercicio.asignatura asig inner join ejercicio.matricula mat on mat.dni = al.dni and mat.AS =  
asig.AS  
where mat.nconvocatoria = 4 and mat.nota < 5  
end
```

3. Ejecutar procedimiento desde Java.

Para la ejecución se ha realizado la conexión con el conector, una vez creado el objeto conexión se crea un “CallableStatement” con la llamada al procedimiento, debido a que este no tiene parametros directamente se introduce el “statement.execute()” y se captura su “resultSet” con “statement.ResultSet()”, una vez se tiene el “resultSet” lo podemos recorrer recuperando sus valores y tratarlos en este caso mostrarlos por pantalla.

Codigo:

```
/**  
 *  
 * @author Javier Gonzalez Rives  
 */  
public class EjecutarProcedimiento {  
    private static final String conexion =  
        "jdbc:mysql://localhost:3306/ejercicio?  
user=root&password=admin&useTimezone=true&serverTimezone=GMT&SSL=true";  
    private static String llamada = "{call alumnosSinCombocatoria()}";  
/**  
    opppppppppppppppppl  
    * @param args  
    */
```

Javier González Rives

```
public static void main(String[] args) {

    try{
        Class.forName("com.mysql.cj.jdbc.Driver");
        // creacion de la conexion
        Connection conec = DriverManager.getConnection(conexion);
        // creacion de la sentencia "llamable" para el procedimiento
        CallableStatement sentencia = conec.prepareCall(llamada);
        // preparacion de la ejecucion
        sentencia.execute();

        ResultSet res = sentencia.getResultSet();

        while(res.next()){
            System.out.println("nomnre del alumno: " + res.getString(1) + " asignatura: "
+res.getString(2) );
        }

        // cerrado de la conexion
        conec.close();
    }catch(ClassNotFoundException cl){
        System.out.println("error");
    }catch(SQLException sq){
        System.out.println("error sql");
        sq.printStackTrace();
    }
}
```

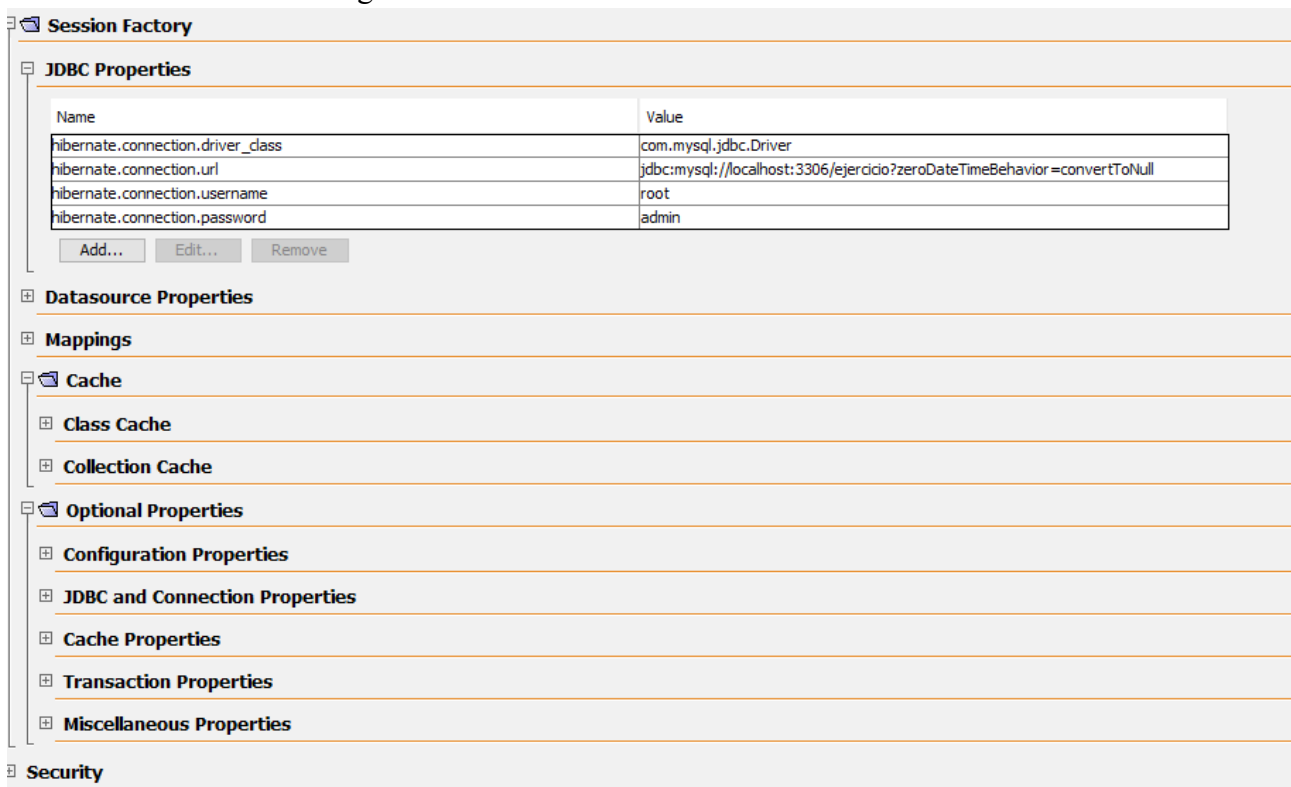
Resultado:

```
ant -f C:\\Users\\likem\\OneDrive\\Documentos\\github\\ac
init:
Deleting: C:\\Users\\likem\\OneDrive\\Documentos\\github\\acces
deps-jar:
Updating property file: C:\\Users\\likem\\OneDrive\\Documentos\\github\\acces
Compiling 1 source file to C:\\Users\\likem\\OneDrive\\Documentos\\github\\acces
compile-single:
run-single:
nomnre del alumno: Angel asignatura: Android Studio
BUILD SUCCESSFUL (total time: 1 second)
```

4. Mapeo u uso de la base de datos con Hibernate

Para usar la base de datos a traves de hibernate en primer lugar debemos importar sus librerías en Netbeans, una vez tenemos estas en el proyecto deberemos de seguir una serie de pasos de forma secuencial.

1. Conectar la base de datos con Netbeans
2. Crear el archivo de configuración



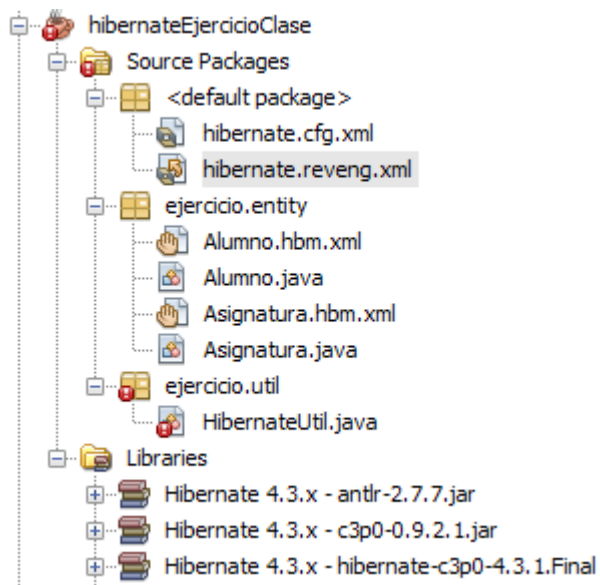
2. Crear un archivo HibernateUtil
3. Crear el archivo encargado de realizar la ingeniería inversa

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE hibernate-reverse-engineering PUBLIC "-//Hibernate/Hiber
<hibernate-reverse-engineering>
  <schema-selection match-catalog="ejercicio"/>
  <table-filter match-name="asignatura"/>
  <table-filter match-name="alumno"/>
</hibernate-reverse-engineering>
```

4. Generar los POJOs dicese las clases mapeadas

Javier González Rives

Tras estos pasos la estructura del proyecto quedaría así



Para realizar la consulta hacemos click derecho sen el archivo “hibernate.cfg.xml” y “run hql query”

Session: hibernate.cfg				
from Alumno				
Result SQL				
Dni	Beca	Edad	Nombre	
12345678	no	22	MiguelAngel	
23456789	no	20	Julian	
34567890	no	20	Javier	
45678901	no	22	Sergey	
56789012	no	33	Alicia	
67890123	no	20	Angel	
78901234	no	19	Sergio	
89012345	no	18	JuanRa	