Ejercicio No. 2. Carrera-Alumno (Consola)

Barrera Pérez Carlos Tonatihu
Profesor: José Asunción Enríquez Zárate
Web Application Development
Grupo: 3CM9

1 de junio de 2019

Índice

1. Introducción

Este ejercicio tuvo como objetivo desarrollar un pequeño CRUD para la base de datos que se muestra en la figura 1.

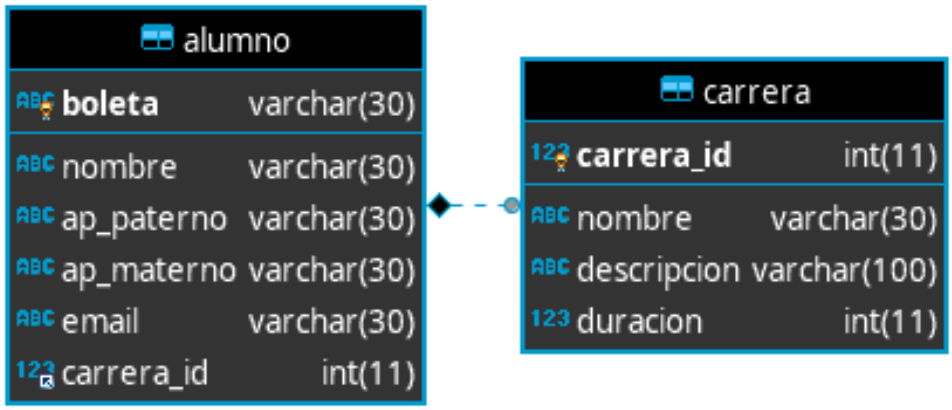

Figura 1: Base de datos que se trabajo en MySQL

Se utilizo el patrón DAO para realizar una implementación más limpia y que pudiera ser utilizada en un futuro.

2. Desarrollo

2.1. Código

```
1  package instituto;
2
3  import java.io.Serializable;
4
5  /**
6   *
7   * @author tonatihu
8   */
9  public class Alumno implements Serializable {
10     private String noBoleta;
11     private String nombre;
```

```
12      private String apPaterno;
13      private String apMaterno;
14      private String email;
15      private Carrera carrera;
16
17      public Alumno() {
18      }
19
20      public String getNoBoleta() {
21          return noBoleta;
22      }
23
24      public void setNoBoleta(String noBoleta) {
25          this.noBoleta = noBoleta;
26      }
27
28      public String getNombre() {
29          return nombre;
30      }
31
32      public void setNombre(String nombre) {
33          this.nombre = nombre;
34      }
35
36      public String getApPaterno() {
37          return apPaterno;
38      }
39
40      public void setApPaterno(String apPaterno) {
41          this.apPaterno = apPaterno;
42      }
43
44      public String getApMaterno() {
45          return apMaterno;
46      }
47
48      public void setApMaterno(String apMaterno) {
49          this.apMaterno = apMaterno;
50      }
51
52      public String getEmail() {
53          return email;
54      }
```

```
55
56      public void setEmail(String email) {
57          this.email = email;
58      }
59
60      public Carrera getCarrera() {
61          return carrera;
62      }
63
64      public void setCarrera(Carrera carrera) {
65          this.carrera = carrera;
66      }
67
68      @Override
69      public String toString() {
70          return "dto.Alumno{" + "noBoleta=" + noBoleta + ",
    nombre='" + nombre
71                  + '\'' + ", apPaterno" + "='" + apPaterno +
     '\''
72                  + ", apMaterno='" + apMaterno + '\'' + ",
    email='"
73                  + email + '\'' + ", carrera=" + carrera + '
    }';
74      }
75  }
```

Archivo 1: Alumno.java

```
1  package instituto;
2
3  import java.io.Serializable;
4
5  /**
6   *
7   * @author tonatihu
8   */
9  public class Carrera implements Serializable {
10      private int id;
11      private String nombre;
12      private String descripcion;
13      private int duracion;
14
15      public Carrera() {
```

```
16      }
17
18      public int getId() {
19          return id;
20      }
21
22      public void setId(int id) {
23          this.id = id;
24      }
25
26      public String getNombre() {
27          return nombre;
28      }
29
30      public void setNombre(String nombre) {
31          this.nombre = nombre;
32      }
33
34      public String getDescripcion() {
35          return descripcion;
36      }
37
38      public void setDescripcion(String descripcion) {
39          this.descripcion = descripcion;
40      }
41
42      public int getDuracion() {
43          return duracion;
44      }
45
46      public void setDuracion(int duracion) {
47          this.duracion = duracion;
48      }
49
50      @Override
51      public String toString() {
52          return "dto.Carrera{id=" + id + ", nombre='" +
    nombre + '\''
53                  + ", descripcion='" + descripcion + '\'' +
    ", duracion="
54                  + duracion + '}';
55      }
56  }
```

Archivo 2: Carrera.java

```java
package instituto;

import java.sql.SQLException;
import java.util.List;

/**
 *
 * @author tonatihu
 * Created on 01-Jun-2019
 */
public abstract class GenericDAO <T>{
    public abstract void create(T entity) throws SQLException;
    public abstract void update(T entity) throws SQLException;
    public abstract T read(T entity) throws SQLException;
    public abstract void delete(T entity) throws SQLException;
    public abstract List<T> readAll() throws SQLException;

}
```

Archivo 3: GenericDAO.java

```java
package instituto;

import java.sql.PreparedStatement;
import java.sql.ResultSet;
import java.sql.SQLException;
import java.util.ArrayList;
import java.util.List;

/**
 * @author tonatihu
 * Created on 1/30/19
 */
public class CarreraDAO extends GenericDAO<Carrera>{
    private static final String SQL_INSERT = "insert into carrera(nombre, "
```

```
15              + "descripcion, duracion) values (?, ?, ?)";
16      private static final String SQL_UPDATE = "update
    carrera set nombre=?, "
17              + "descripcion=?, duracion=? where carrera_id=?
    ";
18      private static final String SQL_SELECT = "select * from
     carrera "
19              + "where carrera_id=?";
20      private static final String SQL_SELECT_ALL = "select *
    from carrera";
21      private static final String SQL_DELETE = "delete from
    carrera "
22              + "where carrera_id=?";
23
24      private final Conexion conexion;
25
26      public CarreraDAO() {
27          conexion = new Conexion();
28      }
29
30      @Override
31      public void create(Carrera c) throws SQLException  {
32          PreparedStatement ps = null;
33          conexion.conectar();
34          try {
35              ps = conexion.createPreparedStatement(
    SQL_INSERT);
36              ps.setString(1, c.getNombre());
37              ps.setString(2, c.getDescripcion());
38              ps.setInt(3, c.getDuracion());
39              ps.executeUpdate();
40          } finally {
41              conexion.cerrar(ps);
42              conexion.cerrar();
43          }
44      }
45
46      @Override
47      public Carrera read(Carrera c) throws SQLException  {
48          PreparedStatement ps = null;
49          ResultSet rs = null;
50          conexion.conectar();
51          try {
```

```
52              ps = conexion.createPreparedStatement(
    SQL_SELECT);
53              ps.setInt(1, c.getId());
54              rs = ps.executeQuery();
55              List<Carrera> resultados = obtenerResultados(rs
    );
56              if (resultados.size() > 0) {
57                  return resultados.get(0);
58              } else {
59                  return null;
60              }
61          } finally {
62              conexion.cerrar(rs);
63              conexion.cerrar(ps);
64              conexion.cerrar();
65          }
66      }
67
68      @Override
69      public List<Carrera> readAll() throws SQLException  {
70          PreparedStatement ps = null;
71          ResultSet rs = null;
72          conexion.conectar();
73          try {
74              ps = conexion.createPreparedStatement(
    SQL_SELECT_ALL);
75              rs = ps.executeQuery();
76              List<Carrera> resultados = obtenerResultados(rs
    );
77              if (resultados.size() > 0) {
78                  return resultados;
79              } else {
80                  return new ArrayList<>();
81              }
82          } finally {
83              conexion.cerrar(rs);
84              conexion.cerrar(ps);
85              conexion.cerrar();
86          }
87      }
88
89      @Override
90      public void update(Carrera c) throws SQLException  {
```

```
91          PreparedStatement ps = null;
92          conexion.conectar();
93          try {
94              ps = conexion.createPreparedStatement(
    SQL_UPDATE);
95              ps.setString(1, c.getNombre());
96              ps.setString(2, c.getDescripcion());
97              ps.setInt(3, c.getDuracion());
98              ps.setInt(4, c.getId());
99              ps.executeUpdate();
100         } finally {
101             conexion.cerrar(ps);
102             conexion.cerrar();
103         }
104     }
105
106     @Override
107     public void delete(Carrera c) throws SQLException  {
108         PreparedStatement ps = null;
109         conexion.conectar();
110         try {
111             ps = conexion.createPreparedStatement(
    SQL_DELETE);
112             ps.setInt(1, c.getId());
113             ps.executeUpdate();
114         } finally {
115             conexion.cerrar(ps);
116             conexion.cerrar();
117         }
118     }
119
120     private List<Carrera> obtenerResultados(ResultSet rs)
    throws SQLException {
121         List<Carrera> resultados = new ArrayList<>();
122         while (rs.next()) {
123             Carrera c = new Carrera();
124             c.setId(rs.getInt("carrera_id"));
125             c.setNombre(rs.getString("nombre"));
126             c.setDescripcion(rs.getString("descripcion"));
127             c.setDuracion(rs.getInt("duracion"));
128             resultados.add(c);
129         }
130         return resultados;
```

```
131     }
132 }
```

Archivo 4: CarreraDAO.java

```
 1 package instituto;
 2
 3 import java.sql.PreparedStatement;
 4 import java.sql.ResultSet;
 5 import java.sql.SQLException;
 6 import java.util.ArrayList;
 7 import java.util.List;
 8
 9 /**
10  *
11  * @author tonatihu
12  */
13 public class AlumnoDAO extends GenericDAO<Alumno>{
14     private static final String SQL_INSERT = "insert into
    alumno(boleta, "
15             + "nombre, ap_paterno, ap_materno, email,
    carrera_id) "
16             + "values (?, ?, ?, ?, ?, ?)";
17     private static final String SQL_UPDATE = "update alumno
     set nombre=?, "
18             + "ap_paterno=?, ap_materno=?, email=?,
    carrera_id=? "
19             + "where boleta=?";
20     private static final String SQL_SELECT = "select * from
     alumno "
21             + "where boleta=?";
22     private static final String SQL_SELECT_ALL = "select *
    from alumno";
23     private static final String SQL_DELETE = "delete from
    alumno "
24             + "where boleta=?";
25
26     private final Conexion conexion;
27
28     public AlumnoDAO() {
29         conexion = new Conexion();
30     }
31
```

```
32      @Override
33      public void create(Alumno a) throws SQLException {
34          PreparedStatement ps = null;
35          conexion.conectar();
36          try {
37              ps = conexion.createPreparedStatement(
    SQL_INSERT);
38              ps.setString(1, a.getNoBoleta());
39              ps.setString(2, a.getNombre());
40              ps.setString(3, a.getApPaterno());
41              ps.setString(4, a.getApMaterno());
42              ps.setString(5, a.getEmail());
43              ps.setInt(6, a.getCarrera().getId());
44              ps.executeUpdate();
45          } finally {
46              conexion.cerrar(ps);
47              conexion.cerrar();
48          }
49      }
50
51      @Override
52      public Alumno read(Alumno a) throws SQLException {
53          PreparedStatement ps = null;
54          ResultSet rs = null;
55          conexion.conectar();
56          try {
57              ps = conexion.createPreparedStatement(
    SQL_SELECT);
58              ps.setString(1, a.getNoBoleta());
59              rs = ps.executeQuery();
60              List<Alumno> resultados = obtenerResultados(rs)
    ;
61              if (resultados.size() > 0) {
62                  return resultados.get(0);
63              } else {
64                  return null;
65              }
66          } finally {
67              conexion.cerrar(rs);
68              conexion.cerrar(ps);
69              conexion.cerrar();
70          }
71      }
```

```
72
73      private List<Alumno> obtenerResultados(ResultSet rs)
    throws SQLException {
74          List<Alumno> resultados = new ArrayList<>();
75          while (rs.next()) {
76              Alumno a = new Alumno();
77              a.setNoBoleta(rs.getString("boleta"));
78              a.setNombre(rs.getString("nombre"));
79              a.setApPaterno(rs.getString("ap_paterno"));
80              a.setApMaterno(rs.getString("ap_materno"));
81              a.setEmail(rs.getString("email"));
82              resultados.add(a);
83          }
84          return resultados;
85      }
86
87      @Override
88      public List readAll() throws SQLException {
89          PreparedStatement ps = null;
90          ResultSet rs = null;
91          conexion.conectar();
92          try {
93              ps = conexion.createPreparedStatement(
    SQL_SELECT_ALL);
94              rs = ps.executeQuery();
95              List<Alumno> resultados = obtenerResultados(rs)
    ;
96              if (resultados.size() > 0) {
97                  return resultados;
98              } else {
99                  return null;
100             }
101         } finally {
102             conexion.cerrar(rs);
103             conexion.cerrar(ps);
104             conexion.cerrar();
105         }
106     }
107
108     @Override
109     public void update(Alumno a) throws SQLException {
110         PreparedStatement ps = null;
111         conexion.conectar();
```

```
112         try {
113             ps = conexion.createPreparedStatement(
    SQL_UPDATE);
114             ps.setString(1, a.getNombre());
115             ps.setString(2, a.getApPaterno());
116             ps.setString(3, a.getApMaterno());
117             ps.setString(4, a.getEmail());
118             ps.setInt(5, a.getCarrera().getId());
119             ps.setString(6, a.getNoBoleta());
120             ps.executeUpdate();
121         } finally {
122             conexion.cerrar(ps);
123             conexion.cerrar();
124         }
125     }
126 
127     @Override
128     public void delete(Alumno a) throws SQLException {
129         PreparedStatement ps = null;
130         conexion.conectar();
131         try {
132             ps = conexion.createPreparedStatement(
    SQL_DELETE);
133             ps.setString(1, a.getNoBoleta());
134             ps.executeUpdate();
135         } finally {
136             conexion.cerrar(ps);
137             conexion.cerrar();
138         }
139     }
140 }
```

Archivo 5: AlumnoDAO.java

```
1 package instituto;
2 
3 import java.sql.Connection;
4 import java.sql.DriverManager;
5 import java.sql.PreparedStatement;
6 import java.sql.ResultSet;
7 import java.sql.SQLException;
8 
9 /**
```

```
10  * @author tonatihu
11  * Created on 1/30/19
12  */
13 public class Conexion {
14     private Connection connection = null;
15 
16     public void conectar() {
17         String user = "root";
18         String pwd = "respuesta42";
19         String url = "jdbc:mysql://localhost:3306/escuelita
    ";
20         String mySqlDriver = "com.mysql.jdbc.Driver";
21         try {
22             Class.forName(mySqlDriver);
23             connection = DriverManager.getConnection(url,
    user, pwd);
24         } catch (Exception e) {
25             e.printStackTrace();
26         }
27     }
28 
29     public PreparedStatement createPreparedStatement(String
    sqlQuery) throws
30 SQLException {
31         return connection.prepareStatement(sqlQuery);
32     }
33 
34     public void cerrar(PreparedStatement ps) {
35         if (ps != null) {
36             try {
37                 ps.close();
38             } catch (SQLException ignored) {}
39         }
40     }
41 
42     public void cerrar() {
43         if (connection != null) {
44             try {
45                 connection.close();
46             } catch (SQLException ignored) {}
47         }
48     }
49 
```

```
50      public void cerrar(ResultSet rs) {
51          if (rs != null) {
52              try {
53                  rs.close();
54              } catch (SQLException ignored) {}
55          }
56      }
57  }
```

Archivo 6: Conexion.java

```
1  package instituto;
2
3  import java.io.BufferedReader;
4  import java.io.IOException;
5  import java.io.InputStreamReader;
6  import java.sql.SQLException;
7  import java.util.List;
8
9  /**
10  *
11  * @author tonatihu
12  */
13 public class Instituto {
14
15      /**
16       * @param args the command line arguments
17       * @throws java.sql.SQLException
18       * @throws java.io.IOException
19       */
20      public static void main(String[] args) throws
    SQLException, IOException {
21          boolean continuar = true;
22          int opcion = 0;
23          BufferedReader br = new BufferedReader(new
24 InputStreamReader(System.in));
25          CarreraDAO dao = new CarreraDAO();
26          Carrera carrera;
27
28          while (continuar) {
29              System.out.println("1) Alta carrera");
30              System.out.println("2) Baja carrera");
31              System.out.println("3) Cambio carrera");
```

```
32              System.out.println("4) Consultar carrera");
33              System.out.println("5) Consultar todas las
    carrera");
34              System.out.println("6) Salir");
35              System.out.println("Selecciona una opcion:");
36              opcion = Integer.valueOf(br.readLine());
37              switch(opcion) {
38                  case 1: // alta
39                      System.out.println("ALTA");
40                      carrera = new Carrera();
41                      System.out.println("Ingrese la
    descripcion:");
42                      carrera.setDescripcion(br.readLine());
43                      System.out.println("Ingrese la duracion
    :");
44                      carrera.setDuracion(Integer.valueOf(br.
    readLine()));
45                      System.out.println("Ingrese el nombre:"
    );
46                      carrera.setNombre(br.readLine());
47                      dao.create(carrera);
48                      System.out.println("Listo...");
49                      break;
50                  case 2: // baja
51                      System.out.println("BAJA");
52                      carrera = new Carrera();
53                      System.out.println("Ingrese el id:");
54                      carrera.setId(Integer.valueOf(br.
    readLine()));
55                      dao.delete(carrera);
56                      System.out.println("Listo...");
57                      break;
58                  case 3: // cambio
59                      System.out.println("CAMBIO");
60                      carrera = new Carrera();
61                      System.out.println("Ingrese el id:");
62                      carrera.setId(Integer.valueOf(br.
    readLine()));
63                      System.out.println("Ingrese la
    descripcion:");
64                      carrera.setDescripcion(br.readLine());
65                      System.out.println("Ingrese la duracion
    :");
```

```
66                      carrera.setDuracion(Integer.valueOf(br.
    readLine()));
67                      System.out.println("Ingrese el nombre:"
    );
68                      carrera.setNombre(br.readLine());
69                      System.out.println("Listo...");
70                      dao.update(carrera);
71                      break;
72                  case 4: // consulta
73                      System.out.println("CONSULTA");
74                      carrera = new Carrera();
75                      System.out.println("Ingrese el id:");
76                      carrera.setId(Integer.valueOf(br.
    readLine()));
77                      carrera = dao.read(carrera);
78                      if (carrera != null)
79                          System.out.println(carrera.toString
    ());
80                      System.out.println("Listo...");
81                      break;
82                  case 5: // consulta todos
83                      System.out.println("CONSULTAR TODOS");
84                      List<Carrera> carreras = dao.readAll();
85                      for (Carrera c : carreras) {
86                          System.out.println(c.toString());
87                      }
88                      System.out.println("Listo");
89                      break;
90                  default:
91                      continuar = false;
92                      break;
93              }
94          }
95          System.out.println("See you space cowboy...");
96      }
97  }
```

Archivo 7: Instituto.java

2.2. Alta

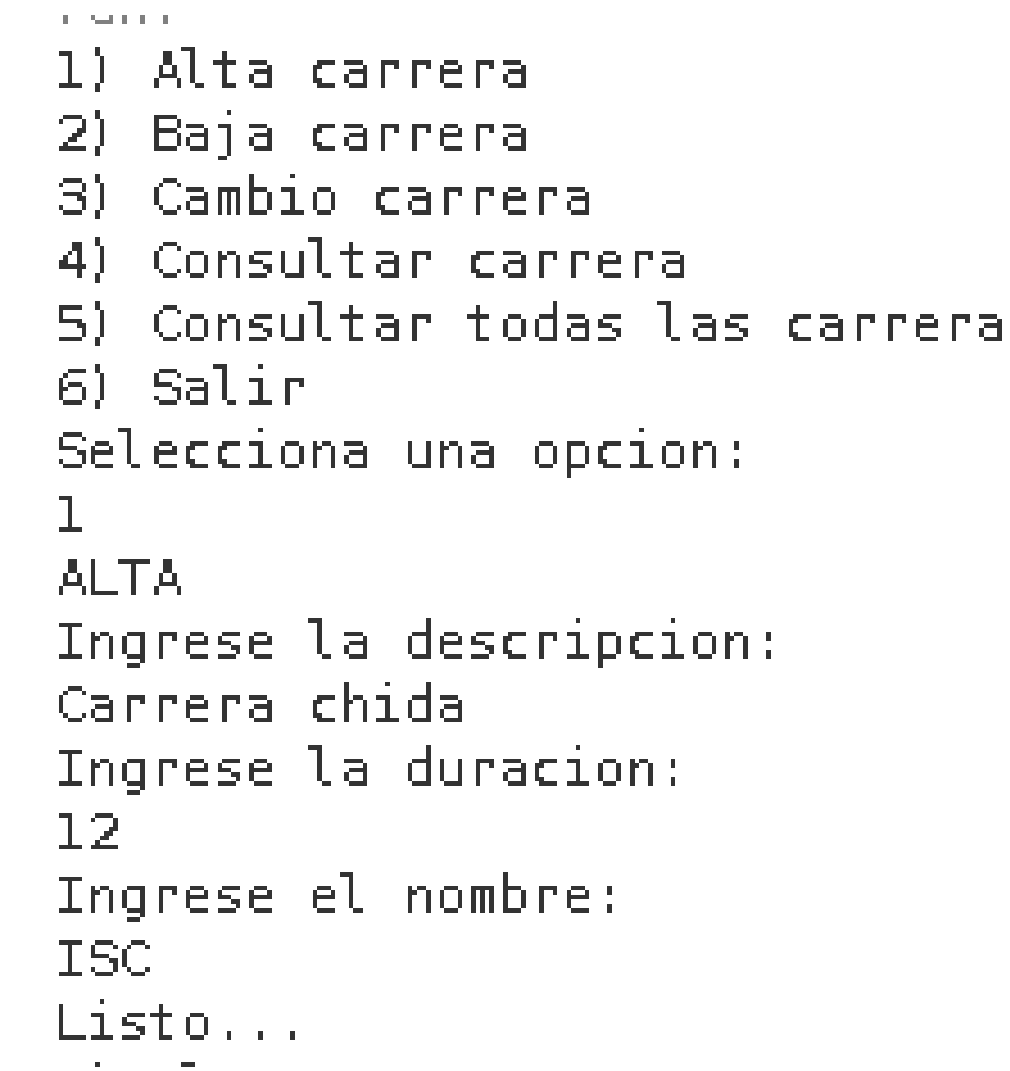

Figura 2: Creamos un registro

2.3. Consulta

```
1) Alta carrera
2) Baja carrera
3) Cambio carrera
4) Consultar carrera
5) Consultar todas las carrera
6) Salir
Selecciona una opcion:
4
CONSULTA
Ingrese el id:
2
dto.Carrera{id=2, nombre='ISC', descripcion='Carrera chida', duracion=12}
Listo...
```

Figura 3: Consulta con base en un id

2.4. Consultar todos

```
1) Alta carrera
2) Baja carrera
3) Cambio carrera
4) Consultar carrera
5) Consultar todas las carrera
6) Salir
Selecciona una opcion:
5
CONSULTAR TODOS
dto.Carrera{id=2, nombre='ISC', descripcion='Carrera chida', duracion=12}
dto.Carrera{id=3, nombre='Fisica', descripcion='Dificil', duracion=3}
Listo
```

Figura 4: Se consultan todos

2.5. Baja

```
1) Alta carrera
2) Baja carrera
3) Cambio carrera
4) Consultar carrera
5) Consultar todas las carrera
6) Salir
Selecciona una opcion:
2
BAJA
Ingrese el id:
3
Listo...
```

Figura 5: Borramos un registro

2.6. Cambio

```
1) Alta carrera
2) Baja carrera
3) Cambio carrera
4) Consultar carrera
5) Consultar todas las carrera
6) Salir
Selecciona una opcion:
3
CAMBIO
Ingrese el id:
2
Ingrese la descripcion:
Es muy chida
Ingrese la duracion:
3
Ingrese el nombre:
ISC
Listo...
```

Figura 6: Realizamos cambios con base en un id

Figura 7: Registros de la base de datos después del cambio y la baja

3. Conclusiones

Este fue un pequeño ejemplo de como implementar el patrón DAO, con ello podemos utilizar el código de este ejercicio en futuras practicas y lo importante es que se entendió el como trabajar siguiendo dicho patrón y lo que este nos permite realizar.