EJERCICIO3

```
public class  03 b2BorrarGrupoyUsuario {
	public static void main(String[] args) {
		ConexionJdbc conJdbc = null;

		try {
			conJdbc = new
ConexionJdbc("configuracion/PropiedadesInventario.txt");
			conJdbc.conectar();


			Scanner tec = new Scanner(System.in);

			System.out.println("id del grupo a borrar: ");
			String idgrupo = tec.nextLine();
			// Faltaría comprobar si el id del grupo introducido existe

			DaoGrupo daoGrupo = new DaoGrupo();
			boolean g = daoGrupo.BorrarGrupoUsuario(idgrupo);
			if(g){
				System.out.print("Usuario y grupo borrados");

			}
			else
				System.out.print("Usuario y grupo no borrados");

		} catch (BusinessException e) {
			System.out.println(e.getMessage());
		} finally{
			conJdbc.desconectar();
		}
	}
}
```

```
public  boolean BorrarGrupoUsuario(String idgrupo) throws BusinessException
	{	boolean borrado=true;
		boolean autoCommit = false;
		PreparedStatement stmBorrarUsuarios = null;
		PreparedStatement stmBuscarUsuarios = null;
		ResultSet rs=null;
		DaoUsuario daoUsuario=new DaoUsuario();
		Connection con=null;
```

```
            PreparedStatement stmBorrarGrupo = null;
            try{
                  con = ConexionJdbc.getConnection();
                autoCommit = con.getAutoCommit();
                con.setAutoCommit(false);

                stmBuscarUsuarios=con.prepareStatement("select idusuario
from usuario where grupo=?");
                stmBuscarUsuarios.setString(1, idgrupo);
                rs=stmBuscarUsuarios.executeQuery();
                while (rs.next())
                {
                      daoUsuario.borrar(rs.getInt("idusuario"));
                }
                borrar(idgrupo);
                borrado =true;


                /*Solución sin utilizar los métodos DAO
                 *stmBorrarUsuarios = con.prepareStatement("DELETE from
usuario WHERE grupo LIKE ?");
                stmBorrarGrupo = con.prepareStatement("DELETE from grupo
WHERE idgrupo LIKE ?");

                stmBorrarUsuarios.setString(1,idGrupo);
                stmBorrarGrupo.setString(1,idGrupo);

                stmBorrarUsuarios.executeUpdate();
                stmBorrarGrupo.executeUpdate();
                */
                con.commit();
            } catch (SQLException e) {
                try {
                      borrado=false;
                      e.printStackTrace();
                      con.rollback();
                } catch (SQLException e1) {
                      e1.printStackTrace();
                }
                throw new BusinessException("No se puede borrar el grupo");
            } finally {
                try {
                      con.setAutoCommit(autoCommit);
                } catch (SQLException e) {
                      e.printStackTrace();
                }
                ConexionJdbc.cerrar(stmBorrarUsuarios);
                ConexionJdbc.cerrar(stmBorrarGrupo);
            }
```

```
            return borrado;

      }
```

EJERCICIO 4

```
package interfaz;


import java.text.ParseException;

import java.text.SimpleDateFormat;

import java.time.LocalDateTime;

import java.time.format.DateTimeFormatter;

import java.util.Date;

import java.util.Scanner;


import jdbc.ConexionJdbc;

import pojos.Articulo;

import pojos.Salida;

import pojos.Usuario;

import dao.DaoArticulo;

import dao.DaoSalida;

import dao.DaoUsuario;

import excepciones.BusinessException;


public class _04_b2DevoluciónSalida {

      public static void main(String[] args) {

            ConexionJdbc conJdbc = null;

            Integer idart,idsalida;

            Salida s=new Salida();
```

```
Articulo a=new Articulo();
Usuario u=new Usuario();
boolean estado=false;
LocalDateTime dateTime=null;

try {
	conJdbc = new ConexionJdbc("configuracion/PropiedadesInventario.txt");
	conJdbc.conectar();
	Scanner tec = new Scanner(System.in);
	// OBTENER VALORES DE DEVOLUCION
	//y puesto que los usuarios no suelen conocer los id ponemos otros valores

	System.out.println("IdSalida a devolver: ");
	Integer id_salida = tec.nextInt();
	tec.nextLine();
	System.out.println("Fecha a devolver: ");
	String fecha_dev = tec.nextLine();
	System.out.println("Id a articulo: ");
	Integer id = tec.nextInt();
	tec.nextLine();
	 DaoSalida ds= new DaoSalida();
	 DaoArticulo dart= new DaoArticulo();
	String sfecha= "2011/03/01";
	Date date=null;

		DateTimeFormatter formatter = DateTimeFormatter.ofPattern("yyyy-MM-dd
HH:mm:ss");
		dateTime = LocalDateTime.parse(fecha_dev, formatter);
```

```
//COMPROBAR QUE LOS DATOS DE ENTRADA SON CORRECTOS
                                        a=dart.buscarPorId(id);
                                       if (a==null)
                                                System.out.println("El artículo no existe");
                                                else{
//REALIZAR devolucion
                                                        s.setIdSalida(id_salida);
                                                        s.setArticulo(id);
                                                        s.setFechaDevolucion(dateTime);

        System.out.println("devolucion"+s.getArticulo()+s.getIdSalida()+s.getFechaDevolucion());
                                                        estado=ds.devolver(s);
                        //INFORMAR AL USUARIO EL ESTADO DE LA OPERACION
                                                        if (estado)
                                                                System.out.println(" Devolucion realizado ");
                                                        else
                                                                System.out.println(" Devolucion no realizado ya tiene
mas de dos artículos prestados ");
                                        }




                        } catch (BusinessException e) {
                                System.out.println(e.getMessage());
                        } finally{
                                conJdbc.desconectar();
```

```
            }



}

}
```

```
public boolean devolver(Salida s) throws BusinessException{
            Connection con = null;
            PreparedStatement pstm = null;
            ResultSet rs=null;
            Integer cont=0;
            boolean estado=false;
            try {

                  con = ConexionJdbc.getConnection();
                  String sql="select * from salida where idsalida=? and
articulo=?";
                  pstm = con.prepareStatement(sql);
                  pstm.setInt(1,s.getIdSalida());
                  pstm.setInt(2,s.getArticulo());
                  rs=pstm.executeQuery();
                  if(rs.first())
                  {
                        LocalDateTime fechaSal =
(rs.getTimestamp("fechaSalida")).toLocalDateTime();
                        if ((s.getFechaDevolucion()).isBefore(fechaSal))
                        {
                              throw new BusinessException("Fecha devolucion
incorrecta");

                        }
                        else
                        {
                        sql="update Salida set fechaDevolucion=? where
idsalida=? and articulo=?";
                        pstm = con.prepareStatement(sql);

      pstm.setTimestamp(1,java.sql.Timestamp.valueOf(s.getFechaDevolucion())
);
                        pstm.setInt(2,s.getIdSalida());
                        pstm.setInt(3,s.getArticulo());
                        pstm.executeUpdate();
                        estado=true;
                        }
```

```
                }
                else
                {
                      throw new BusinessException("Error al devolver");
                }


          } catch (SQLException e){
                e.printStackTrace();
                throw new BusinessException("Error al prestar");
          } finally{
                ConexionJdbc.cerrar(pstm);
          }
          return estado;

    }
```