**Universidad de Guadalajara**

**Departamento de ciencias computacionales**

**Ingeniería de Software II**

Código Fuente

****

*Piceno Roque Adán Oswaldo.*

***Código:*** *213405611.*

***Carrera:*** *Ing. Informática.*

***Sección:*** *D04*

*Mtra. Bolaños Gutiérrez, Sergio Manuel.*

Contenido

[Paquete ENTITY 3](#_Toc10035141)

[Clase Empleado 3](#_Toc10035142)

[Clase Nómina 5](#_Toc10035143)

[Clase nominaEmpleado 6](#_Toc10035144)

[Clase Usuario 9](#_Toc10035145)

[Paquete dao 11](#_Toc10035146)

[empleadoDao 11](#_Toc10035147)

[nominaDao 12](#_Toc10035148)

[nominaEmpleadoDao 12](#_Toc10035149)

[usuarioDao 13](#_Toc10035150)

[Paquete daoMysql 13](#_Toc10035151)

[Conexión 13](#_Toc10035152)

[empleadoDaoMsyql 14](#_Toc10035153)

[nominaDaoMysql 20](#_Toc10035154)

[nominaEmpleadoDaoMysql 24](#_Toc10035155)

[usuarioDaoMysql 32](#_Toc10035156)

# Paquete ENTITY

## Clase Empleado

**package** Entity;

**import** **java.util.Date**;

*/\*\**

*\**

*\* @author Adan*

*\*/*

**public** **class** **Empleado** {

**private** int codigo;

**private** String nombre;

**private** String apellidoP;

**private** String apellidoM;

**private** Date fehcaIngreso;

**private** short antiguedad;

**private** double sueldo;

**private** double sdi;

*/\*\**

*\* @return the nombre*

*\*/*

**public** String getNombre() {

**return** nombre;

}

*/\*\**

*\* @param nombre the nombre to set*

*\*/*

**public** void setNombre(String nombre) {

**this**.nombre = nombre;

}

*/\*\**

*\* @return the fehcaIngreso*

*\*/*

**public** Date getFehcaIngreso() {

**return** fehcaIngreso;

}

*/\*\**

*\* @param fehcaIngreso the fehcaIngreso to set*

*\*/*

**public** void setFehcaIngreso(Date fehcaIngreso) {

**this**.fehcaIngreso = fehcaIngreso;

}

*/\*\**

*\* @return the antiguedad*

*\*/*

**public** short getAntiguedad() {

**return** antiguedad;

}

*/\*\**

*\* @param antiguedad the antiguedad to set*

*\*/*

**public** void setAntiguedad(short antiguedad) {

**this**.antiguedad = antiguedad;

}

*/\*\**

*\* @return the sueldo*

*\*/*

**public** double getSueldo() {

**return** sueldo;

}

*/\*\**

*\* @param sueldo the sueldo to set*

*\*/*

**public** void setSueldo(double sueldo) {

**this**.sueldo = sueldo;

}

*/\*\**

*\* @return the sdi*

*\*/*

**public** double getSdi() {

**return** sdi;

}

*/\*\**

*\* @param sdi the sdi to set*

*\*/*

**public** void setSdi(double sdi) {

**this**.sdi = sdi;

}

*/\*\**

*\* @return the apellidoM*

*\*/*

**public** String getApellidoM() {

**return** apellidoM;

}

*/\*\**

*\* @param apellidoM the apellidoM to set*

*\*/*

**public** void setApellidoM(String apellidoM) {

**this**.apellidoM = apellidoM;

}

*/\*\**

*\* @return the apellidoP*

*\*/*

**public** String getApellidoP() {

**return** apellidoP;

}

*/\*\**

*\* @param apellidoP the apellidoP to set*

*\*/*

**public** void setApellidoP(String apellidoP) {

**this**.apellidoP = apellidoP;

}

*/\*\**

*\* @return the codigo*

*\*/*

**public** int getCodigo() {

**return** codigo;

}

*/\*\**

*\* @param codigo the codigo to set*

*\*/*

**public** void setCodigo(int codigo) {

**this**.codigo = codigo;

}

**public** String toString(){

**return** getApellidoP() + " " + getApellidoM()+ " " + getNombre();

}

}

## Clase Nómina

**package** Entity;

**import** **java.text.SimpleDateFormat**;

**import** **java.util.Calendar**;

**import** **java.util.Date**;

*/\*\**

*\**

*\* @author Adan*

*\*/*

**public** **class** **Nomina** {

*/\*\**

*\* @return the id*

*\*/*

**public** int getId() {

**return** id;

}

*/\*\**

*\* @param id the id to set*

*\*/*

**public** void setId(int id) {

**this**.id = id;

}

*/\*\**

*\* @return the tipo*

*\*/*

**public** int getTipo() {

**return** tipo;

}

*/\*\**

*\* @param tipo the tipo to set*

*\*/*

**public** void setTipo(int tipo) {

**this**.tipo = tipo;

}

*/\*\**

*\* @return the fechaInicio*

*\*/*

**public** Date getFechaInicio() {

**return** fechaInicio;

}

*/\*\**

*\* @param fechaInicio the fechaInicio to set*

*\*/*

**public** void setFechaInicio(Date fechaInicio) {

**this**.fechaInicio = fechaInicio;

}

*/\*\**

*\* @return the fechaFin*

*\*/*

**public** Date getFechaFin() {

**return** fechaFin;

}

*/\*\**

*\* @param fechaFin the fechaFin to set*

*\*/*

**public** void setFechaFin(Date fechaFin) {

**this**.fechaFin = fechaFin;

}

**private** int id;

**private** int tipo;

**private** Date fechaInicio;

**private** Date fechaFin;

*/\*\**

*\* @return the idNomina*

*\*/*

**public** String toString(){

**return** getFechaInicio() + " hasta " + getFechaFin();

}

}

## Clase nominaEmpleado

**package** Entity;

*/\*\**

*\**

*\* @author Adan*

*\*/*

**public** **class** **nominaEmpleado** {

*/\*\**

*\* @return the id*

*\*/*

**public** int getId() {

**return** id;

}

*/\*\**

*\* @param id the id to set*

*\*/*

**public** void setId(int id) {

**this**.id = id;

}

*/\*\**

*\* @return the nomina*

*\*/*

**public** Nomina getNomina() {

**return** nomina;

}

*/\*\**

*\* @param nomina the nomina to set*

*\*/*

**public** void setNomina(Nomina nomina) {

**this**.nomina = nomina;

}

*/\*\**

*\* @return the empleado*

*\*/*

**public** Empleado getEmpleado() {

**return** empleado;

}

*/\*\**

*\* @param empleado the empleado to set*

*\*/*

**public** void setEmpleado(Empleado empleado) {

**this**.empleado = empleado;

}

*/\*\**

*\* @return the diasTrabajados*

*\*/*

**public** int getDiasTrabajados() {

**return** diasTrabajados;

}

*/\*\**

*\* @param diasTrabajados the diasTrabajados to set*

*\*/*

**public** void setDiasTrabajados(int diasTrabajados) {

**this**.diasTrabajados = diasTrabajados;

}

*/\*\**

*\* @return the sueldoTrabajo*

*\*/*

**public** double getSueldoTrabajo() {

**return** sueldoTrabajo;

}

*/\*\**

*\* @param sueldoTrabajo the sueldoTrabajo to set*

*\*/*

**public** void setSueldoTrabajo(double sueldoTrabajo) {

**this**.sueldoTrabajo = sueldoTrabajo;

}

*/\*\**

*\* @return the spe\_isr*

*\*/*

**public** double getSpe\_isr() {

**return** spe\_isr;

}

*/\*\**

*\* @param spe\_isr the spe\_isr to set*

*\*/*

**public** void setSpe\_isr(double spe\_isr) {

**this**.spe\_isr = spe\_isr;

}

*/\*\**

*\* @return the infonavit*

*\*/*

**public** double getInfonavit() {

**return** infonavit;

}

*/\*\**

*\* @param infonavit the infonavit to set*

*\*/*

**public** void setInfonavit(double infonavit) {

**this**.infonavit = infonavit;

}

*/\*\**

*\* @return the cuotaImss*

*\*/*

**public** double getCuotaImss() {

**return** cuotaImss;

}

*/\*\**

*\* @param cuotaImss the cuotaImss to set*

*\*/*

**public** void setCuotaImss(double cuotaImss) {

**this**.cuotaImss = cuotaImss;

}

*/\*\**

*\* @return the censantiaVejez*

*\*/*

**public** double getCensantiaVejez() {

**return** censantiaVejez;

}

*/\*\**

*\* @param censantiaVejez the censantiaVejez to set*

*\*/*

**public** void setCensantiaVejez(double censantiaVejez) {

**this**.censantiaVejez = censantiaVejez;

}

*/\*\**

*\* @return the sueldoNeto*

*\*/*

**public** double getSueldoNeto() {

**return** sueldoNeto;

}

*/\*\**

*\* @param sueldoNeto the sueldoNeto to set*

*\*/*

**public** void setSueldoNeto(double sueldoNeto) {

**this**.sueldoNeto = sueldoNeto;

}

**private** int id;

**private** Nomina nomina;

**private** Empleado empleado;

**private** int diasTrabajados;

**private** double sueldoTrabajo;

**private** double spe\_isr;

**private** double infonavit;

**private** double cuotaImss;

**private** double censantiaVejez;

**private** double sueldoNeto;

}

## Clase Usuario

**package** Entity;

*/\*\**

*\**

*\* @author Adan*

*\*/*

**public** **class** **Usuario** {

**private** int Codigo;

**private** String nombre;

**private** String apellidoP;

**private** String apellidoM;

**private** String contrasena;

**private** String role;

**public** String nomPriv [] = **new** String [4];

**public** String empPriv [] = **new** String [4];

**public** String cuoPriv [] = **new** String [4];

**public** Usuario(){

**for** (int i = 0; i < 4; i++) {

nomPriv[i] = "";

empPriv[i] = "";

cuoPriv[i] = "";

}

**this**.nombre = "";

**this**.apellidoM = "";

**this**.apellidoP = "";

**this**.contrasena = "";

**this**.role = "";

}

*/\*\**

*\* @return the Codigo*

*\*/*

**public** int getCodigo() {

**return** Codigo;

}

*/\*\**

*\* @param Codigo the Codigo to set*

*\*/*

**public** void setCodigo(int Codigo) {

**this**.Codigo = Codigo;

}

*/\*\**

*\* @return the nombre*

*\*/*

**public** String getNombre() {

**return** nombre;

}

*/\*\**

*\* @param nombre the nombre to set*

*\*/*

**public** void setNombre(String nombre) {

**this**.nombre = nombre;

}

*/\*\**

*\* @return the apellidoP*

*\*/*

**public** String getApellidoP() {

**return** apellidoP;

}

*/\*\**

*\* @param apellidoP the apellidoP to set*

*\*/*

**public** void setApellidoP(String apellidoP) {

**this**.apellidoP = apellidoP;

}

*/\*\**

*\* @return the apellidoM*

*\*/*

**public** String getApellidoM() {

**return** apellidoM;

}

*/\*\**

*\* @param apellidoM the apellidoM to set*

*\*/*

**public** void setApellidoM(String apellidoM) {

**this**.apellidoM = apellidoM;

}

*/\*\**

*\* @return the contrasena*

*\*/*

**public** String getContrasena() {

**return** contrasena;

}

*/\*\**

*\* @param contrasena the contrasena to set*

*\*/*

**public** void setContrasena(String contrasena) {

**this**.contrasena = contrasena;

}

*/\*\**

*\* @return the role*

*\*/*

**public** String getRole() {

**return** role;

}

*/\*\**

*\* @param role the role to set*

*\*/*

**public** void setRole(String role) {

**this**.role = role;

}

**public** String toString (){

**return** getApellidoP() + " " + getApellidoM()+ " " + getNombre();

}

}

# Paquete dao

## empleadoDao

**package** dao;

**import** **Entity.Empleado**;

**import** **Entity.Usuario**;

**import** **java.util.List**;

*/\*\**

*\**

*\* @author Adan*

*\*/*

**public** **interface** **empleadoDao** {

**public** Usuario getUsuario();

**public** void setUsuario(Usuario usuario);

**public** void agrega(Empleado empleado);

**public** void Actualiza (Empleado empleado);

**public** void Elimina (Empleado empleado);

**public** List <Empleado> obtenTodos();

**public** List <Empleado> obtenEmpleado(int codigo);

**public** List <Empleado> obtenMayorSueldo(Double sueldo);

**public** List <Empleado> obtenMenorSueldo(Double sueldo);

**public** List <Empleado> ontenMayorAntiguedad();

**public** List <Empleado> obtenMenotAntiguedad();

}

## nominaDao

**package** dao;

**import** **Entity.Nomina**;

**import** **Entity.Empleado**;

**import** **Entity.Usuario**;

**import** **java.util.List**;

*/\*\**

*\**

*\* @author Adan*

*\*/*

**public** **interface** **nominaDao** {

**public** Usuario getUsuario();

**public** void setUsuario(Usuario usuario);

**public** void Agrega(Nomina nomina);

**public** void Actualiza (Nomina nomina);

**public** void Elimina (Nomina nomina);

**public** List <Nomina> obtenTodos();

**public** List <Nomina> obtenTipo(int tipo);

**public** Nomina obtenNomina(int id);

}

## nominaEmpleadoDao

**package** dao;

**import** **Entity.Usuario**;

**import** **Entity.nominaEmpleado**;

**import** **java.util.List**;

*/\*\**

*\**

*\* @author Adan*

*\*/*

**public** **interface** **nominaEmpleadoDao** {

**public** Usuario getUsuario();

**public** void setUsuario(Usuario usuario);

**public** void Agrega(nominaEmpleado nominaEmp);

**public** void Actualiza (nominaEmpleado nominaEmp);

**public** void Elimina (int id);

**public** List <nominaEmpleado> obtenTodos(int tipo);

**public** List <nominaEmpleado> registroNominaEmpleado(int tipo);

**public** List <nominaEmpleado> obtenNominaEmpleado(int id);

**public** List <nominaEmpleado> obtenNominaFecha(int idNomina);

**public** List <nominaEmpleado> obtenNominaEmpleadoFecha(int idNomina, int idEmpleado);

}

## usuarioDao

**package** dao;

**import** **Entity.Usuario**;

**import** **Entity.Empleado**;

**import** **java.util.List**;

*/\*\**

*\**

*\* @author Adan*

*\*/*

**public** **interface** **usuarioDao** {

**public** Usuario iniciaSesion(int codigo, String contrasena);

**public** void agrega(Usuario usuario);

**public** void agregaTodos(Usuario usuario);

**public** void Actualiza (Usuario usuario);

**public** void otorgaTodosLosDerechos(Usuario usuario);

**public** void Elimina (Usuario usuario);

**public** void quitarPrivilegios(Usuario Usuario);

**public** List <Usuario> obtenTodos();

**public** List <Usuario> obtenUsuarioAdmin();

**public** List <Empleado> obteneEmpleados();

}

# Paquete daoMysql

## Conexión

**package** daoMysql;

**import** **com.mysql.jdbc.Connection**;

**import** **java.sql.\***;

**import** **java.util.logging.Level**;

**import** **java.util.logging.Logger**;

**import** **Entity.Usuario**;

**import** **javax.swing.JOptionPane**;

*/\*\**

*\**

*\* @author Adan*

*\*/*

**public** **class** **Conexion** {

**private** Usuario usuario;

**private** Connection conn;

**private** **static** **final** String url = "jdbc:mysql://localhost:3306/nomina";

**private** **static** **final** String driver = "com.mysql.jdbc.Driver";

**private** String user;

**private** String password;

*//Iniciar Conexion (Constructor)*

Conexion (){

}

**private** void conectar(){

**try**{

Class.forName(driver);

conn = (Connection) DriverManager.getConnection(**this**.url, **this**.user, **this**.password);

*//conn = (Connection) DriverManager.getConnection("jdbc:mysql://localhost:3306/nomina","root","");*

}**catch**(ClassNotFoundException | SQLException e){

e.printStackTrace();

JOptionPane.showMessageDialog(**null**, "Cuenta de usuario o contraseña invalida");

}

}

**public** void newConnetionCont(String user, String password) **throws** SQLException{

**this**.user=user;

**this**.password=password;

conectar();

}

**public** void newConnection(){

user = "root";

password="";

conectar();

}

**public** Connection getConnection() **throws** SQLException{

**return** conn;

}

**public** void Salir(){

**try** {

conn.close();

} **catch** (SQLException ex) {

System.out.println("Error al salir");

}

}

}

## empleadoDaoMsyql

**package** daoMysql;

**import** **Entity.Empleado**;

**import** **Entity.Usuario**;

**import** **dao.empleadoDao**;

**import** **java.sql.\***;

**import** **java.util.ArrayList**;

**import** **java.util.List**;

**import** **java.util.logging.Level**;

**import** **java.util.logging.Logger**;

**import** **javax.swing.JOptionPane**;

*/\*\**

*\**

*\* @author Adan*

*\*/*

**public** **class** **empleadoDaoMysql** **implements** empleadoDao {

**private** Usuario usuario;

@Override

**public** void agrega(Empleado empleado) {

*//throw new UnsupportedOperationException("Not supported yet."); //To change body of generated methods, choose Tools | Templates.*

String clave = Integer.toString(getUsuario().getCodigo());

String password = getUsuario().getContrasena();

Conexion conexion = **new** Conexion();

String query = "INSERT INTO empleado (nombre, apellidoP , apellidoM, sueldoDiario, fechaIngreso, activo) VALUES (?,?,?,?,?,?)";

PreparedStatement stmt;

**try** {

conexion.newConnetionCont(clave, password);

stmt = conexion.getConnection().prepareStatement(query);

java.sql.Date fechaActual= **new** java.sql.Date( empleado.getFehcaIngreso().getTime());

stmt.setString(1, empleado.getNombre());

stmt.setString(2, empleado.getApellidoP());

stmt.setString(3, empleado.getApellidoM());

stmt.setDouble(4, empleado.getSueldo());

stmt.setDate(5, fechaActual);

stmt.setInt(6, 1);

stmt.executeUpdate();

JOptionPane.showMessageDialog(**null**, "Se ha registrado el nuevo empleado");

stmt.close();

} **catch** (SQLException ex) {

JOptionPane.showMessageDialog(**null**, "Verifique sus derechos con el administrador", "¡¡¡ERROR!!!",JOptionPane.WARNING\_MESSAGE);

Logger.getLogger(empleadoDaoMysql.class.getName()).log(Level.SEVERE, **null**, ex);

}**finally**{

conexion.Salir();

}

}

@Override

**public** void Actualiza(Empleado empleado) {

String clave = Integer.toString(getUsuario().getCodigo());

String password = getUsuario().getContrasena();

Conexion conexion = **new** Conexion();

**try** {

String query;

conexion.newConnetionCont(clave, password);

PreparedStatement stmt;

*//Valida los campos que no fueron nulos concatenando la sintaxis del query*

*//Nombre*

**if** (!"".equals(empleado.getNombre())){

query = "UPDATE empleado SET nombre=? WHERE codigo=?";

stmt = conexion.getConnection().prepareStatement(query);

stmt.setString(1, empleado.getNombre());

stmt.setInt(2, empleado.getCodigo());

stmt.executeUpdate();

}

*//Apellido Paterno*

**if** (!"".equals(empleado.getApellidoP())){

query = "UPDATE empleado SET apellidoP=? WHERE codigo=?";

stmt = conexion.getConnection().prepareStatement(query);

stmt.setString(1, empleado.getApellidoP());

stmt.setInt(2, empleado.getCodigo());

stmt.executeUpdate();

}

*//ApellidoMaterno*

**if** (!"".equals(empleado.getApellidoM())){

query = "UPDATE empleado SET apellidoM=? WHERE codigo=?";

stmt = conexion.getConnection().prepareStatement(query);

stmt.setString(1, empleado.getApellidoM());

stmt.setInt(2, empleado.getCodigo());

stmt.executeUpdate();

}

*//Fecha de ingreso*

**if**(empleado.getFehcaIngreso() != **null**){

java.sql.Date fechaActual= **new** java.sql.Date( empleado.getFehcaIngreso().getTime());

query = "UPDATE empleado SET fechaIngreso=? WHERE codigo=?";

stmt = conexion.getConnection().prepareStatement(query);

stmt.setDate(1, fechaActual);

stmt.setInt(2, empleado.getCodigo());

stmt.executeUpdate();

}

*//Sueldo Diario*

**if** (empleado.getSueldo() != 0.0){

query = "UPDATE empleado SET sueldoDiario=? WHERE codigo=?";

stmt = conexion.getConnection().prepareStatement(query);

stmt.setDouble(1, empleado.getSueldo());

stmt.setInt(2, empleado.getCodigo());

stmt.executeUpdate();

}

} **catch** (SQLException ex) {

JOptionPane.showMessageDialog(**null**, "Verifique sus derechos con el administrador", "¡¡¡ERROR!!!",JOptionPane.WARNING\_MESSAGE);

}**finally**{

conexion.Salir();

}

}

@Override

**public** void Elimina(Empleado empleado) {

String clave = Integer.toString(getUsuario().getCodigo());

String password = getUsuario().getContrasena();

Conexion conexion = **new** Conexion();

String query= "DELETE FROM empleado WHERE codigo=?";

PreparedStatement stmt;

**try**{

conexion.newConnetionCont(clave, password);

stmt = conexion.getConnection().prepareStatement(query);

*//stmt.setInt(1, 0);*

stmt.setInt(1, empleado.getCodigo());

stmt.executeUpdate();

}**catch**(SQLException ex){

JOptionPane.showMessageDialog(**null**, "Verifique sus derechos con el administrador", "¡¡¡ERROR!!!",JOptionPane.WARNING\_MESSAGE);

}**finally**{

conexion.Salir();

}

}

@Override

**public** List<Empleado> obtenTodos() {

List <Empleado> listaEmp = **new** ArrayList<Empleado>();

Conexion conexion = **new** Conexion();

conexion.newConnection();

*/\*\**

*\*El QUERYEMPLEADO SIRVE PARA PODER SACAR LA ANTIGUEDAD DEL EMPLEADO YA QUE ES UN METODO CALCULADO*

*\* JUNTO CON TODOS LOS DEMAS CAMPOS DEL EMPLEADO*

*\* Y DESPUES CONFORMME A LA ANTIGUEDAD SACAR EL SUELDO DIARIO INTEGRADO (SDI) HACIENDO*

*\* INNER JOIN ANTIGUEDAD->AÑO CON IMSS. LA RAZON ES POR QUE NO LEE ANTIGUEDAD COMO CAMPO*

*\* SINO ALIAS Y LOS ALIAS NO SE PUEDEN USAR PARA LAS RESTRICCIONES "WHERE"*

*\*/*

String queryEmpleado="SELECT \*, 1+TIMESTAMPDIFF(YEAR,fechaIngreso,CURDATE()) AS ant FROM empleado";

String query =" SELECT e.\*, i.sdi\*e.sueldoDiario FROM ("+queryEmpleado+") e "

+ "INNER JOIN imss i ON e.ant = i.anyo WHERE e.activo=? AND e.codigo!=? ORDER BY e.apellidoP, e.apellidoM ";

PreparedStatement stmt;

**try** {

stmt = conexion.getConnection().prepareStatement(query);

stmt.setInt(1, 1);

stmt.setInt(2, getUsuario().getCodigo());

ResultSet rs = stmt.executeQuery();

**while** (rs.next()){

Empleado empleado = **new** Empleado();

empleado.setCodigo(rs.getInt(1));

empleado.setNombre(rs.getString(2));

empleado.setApellidoP(rs.getString(3));

empleado.setApellidoM(rs.getString(4));

empleado.setFehcaIngreso(rs.getDate(5));

empleado.setSueldo(rs.getDouble(6));

*//Omite el 7 ya que representa si esta activo a no en mysql*

empleado.setAntiguedad(rs.getShort(8));

empleado.setSdi(rs.getDouble(9));

listaEmp.add(empleado);

}

} **catch** (SQLException ex) {

JOptionPane.showMessageDialog(**null**, "Verifique sus derechos con el administrador", "¡¡¡ERROR!!!",JOptionPane.WARNING\_MESSAGE);

Logger.getLogger(empleadoDaoMysql.class.getName()).log(Level.SEVERE, **null**, ex);

}**finally**{

conexion.Salir();

}

**return** listaEmp;

}

@Override

**public** List<Empleado> obtenEmpleado(int codigo) {

List <Empleado> listaEmp = **new** ArrayList<Empleado>();

Conexion conexion = **new** Conexion();

conexion.newConnection();

String queryEmpleado="SELECT \*, 1+TIMESTAMPDIFF(YEAR,fechaIngreso,CURDATE()) AS ant FROM empleado";

String query =" SELECT e.\*, i.sdi\*e.sueldoDiario FROM ("+queryEmpleado+") e "

+ "INNER JOIN imss i ON e.ant = i.anyo WHERE e.activo=? AND codigo =? ORDER BY e.apellidoP, e.apellidoM ";

*/\*String query = "SELECT \*, 1+TIMESTAMPDIFF(YEAR,fechaIngreso,CURDATE()) AS ant, i.sdi "*

*+ "FROM empleado WHERE activo=? AND codigo =? ORDER BY apellidoP, apellidoM ";\*/*

PreparedStatement stmt;

**try** {

stmt = conexion.getConnection().prepareStatement(query);

stmt.setInt(1, 1);

stmt.setInt(2, codigo);

ResultSet rs = stmt.executeQuery();

**while** (rs.next()){

Empleado empleado = **new** Empleado();

empleado.setCodigo(rs.getInt(1));

empleado.setNombre(rs.getString(2));

empleado.setApellidoP(rs.getString(3));

empleado.setApellidoM(rs.getString(4));

empleado.setFehcaIngreso(rs.getDate(5));

empleado.setSueldo(rs.getDouble(6));

*//Omite el 7 ya que es el acmpo activo de mysql*

empleado.setAntiguedad(rs.getShort(8));

empleado.setSdi(rs.getDouble(9));

listaEmp.add(empleado);

}

} **catch** (SQLException ex) {

JOptionPane.showMessageDialog(**null**, "Verifique sus derechos con el administrador", "¡¡¡ERROR!!!",JOptionPane.WARNING\_MESSAGE);

Logger.getLogger(empleadoDaoMysql.class.getName()).log(Level.SEVERE, **null**, ex);

}**finally**{

conexion.Salir();

}

**return** listaEmp;

}

@Override

**public** List<Empleado> obtenMayorSueldo(Double sueldo) {

List <Empleado> listaEmp = **new** ArrayList<Empleado>();

Conexion conexion = **new** Conexion();

conexion.newConnection();

String queryEmpleado="SELECT \*, 1+TIMESTAMPDIFF(YEAR,fechaIngreso,CURDATE()) AS ant FROM empleado";

String query =" SELECT e.\*, i.sdi\*e.sueldoDiario FROM ("+queryEmpleado+") e "

+ "INNER JOIN imss i ON e.ant = i.anyo WHERE e.activo=? AND e.sueldoDiario\*i.sdi >=? ORDER BY e.apellidoP, e.apellidoM ";

*/\*String query = "SELECT \*, 1+TIMESTAMPDIFF(YEAR,fechaIngreso,CURDATE()) AS ant, i.sdi "*

*+ "FROM empleado WHERE activo=? AND sueldoDiario >=? ORDER BY apellidoP, apellidoM ";\*/*

PreparedStatement stmt;

**try** {

stmt = conexion.getConnection().prepareStatement(query);

stmt.setInt(1, 1);

stmt.setDouble(2, sueldo);

ResultSet rs = stmt.executeQuery();

**while** (rs.next()){

Empleado empleado = **new** Empleado();

empleado.setCodigo(rs.getInt(1));

empleado.setNombre(rs.getString(2));

empleado.setApellidoP(rs.getString(3));

empleado.setApellidoM(rs.getString(4));

empleado.setFehcaIngreso(rs.getDate(5));

empleado.setSueldo(rs.getDouble(6));

*//omite el 7 ya que es el campo activo de myqsl*

empleado.setAntiguedad(rs.getShort(8));

empleado.setSdi(rs.getDouble(9));

listaEmp.add(empleado);

}

} **catch** (SQLException ex) {

JOptionPane.showMessageDialog(**null**, "Verifique sus derechos con el administrador", "¡¡¡ERROR!!!",JOptionPane.WARNING\_MESSAGE);

Logger.getLogger(empleadoDaoMysql.class.getName()).log(Level.SEVERE, **null**, ex);

}**finally**{

conexion.Salir();

}

**return** listaEmp;

}

@Override

**public** List<Empleado> obtenMenorSueldo(Double sueldo) {

List <Empleado> listaEmp = **new** ArrayList<Empleado>();

Conexion conexion = **new** Conexion();

conexion.newConnection();

String queryEmpleado="SELECT \*, 1+TIMESTAMPDIFF(YEAR,fechaIngreso,CURDATE()) AS ant FROM empleado";

String query =" SELECT e.\*, i.sdi\*e.sueldoDiario FROM ("+queryEmpleado+") e "

+ "INNER JOIN imss i ON e.ant = i.anyo WHERE e.activo=? AND e.sueldoDiario\*i.sdi <=? ORDER BY e.apellidoP, e.apellidoM ";

*/\*String query = "SELECT \*, 1+TIMESTAMPDIFF(YEAR,fechaIngreso,CURDATE()) AS ant FROM empleado WHERE activo=? AND sueldoDiario <=? ORDER BY apellidoP, apellidoM ";\*/*

PreparedStatement stmt;

**try** {

stmt = conexion.getConnection().prepareStatement(query);

stmt.setInt(1, 1);

stmt.setDouble(2, sueldo);

ResultSet rs = stmt.executeQuery();

**while** (rs.next()){

Empleado empleado = **new** Empleado();

empleado.setCodigo(rs.getInt(1));

empleado.setNombre(rs.getString(2));

empleado.setApellidoP(rs.getString(3));

empleado.setApellidoM(rs.getString(4));

empleado.setFehcaIngreso(rs.getDate(5));

empleado.setSueldo(rs.getDouble(6));

*//Omite el 7 ya que es el campo activo de mysql*

empleado.setAntiguedad(rs.getShort(8));

empleado.setSdi(rs.getDouble(9));

listaEmp.add(empleado);

}

} **catch** (SQLException ex) {

JOptionPane.showMessageDialog(**null**, "Verifique sus derechos con el administrador", "¡¡¡ERROR!!!",JOptionPane.WARNING\_MESSAGE);

Logger.getLogger(empleadoDaoMysql.class.getName()).log(Level.SEVERE, **null**, ex);

}**finally**{

conexion.Salir();

}

**return** listaEmp;

}

@Override

**public** List<Empleado> ontenMayorAntiguedad() {

**throw** **new** UnsupportedOperationException("Not supported yet."); *//To change body of generated methods, choose Tools | Templates.*

}

## nominaDaoMysql

**package** daoMysql;

**import** **Entity.Empleado**;

**import** **Entity.Nomina**;

**import** **Entity.Usuario**;

**import** **dao.empleadoDao**;

**import** **dao.nominaDao**;

**import** **java.sql.PreparedStatement**;

**import** **java.sql.ResultSet**;

**import** **java.sql.SQLException**;

**import** **java.util.ArrayList**;

**import** **java.util.List**;

**import** **java.util.logging.Level**;

**import** **java.util.logging.Logger**;

**import** **javax.swing.JOptionPane**;

*/\*\**

*\**

*\* @author Adan*

*\*/*

**public** **class** **nominaDaoMysql** **implements** nominaDao{

**private** Usuario usuario;

@Override

**public** void Agrega(Nomina nomina) {

Conexion conexion = **new** Conexion();

*//conexion.newConnection();*

String clave = Integer.toString(getUsuario().getCodigo());

String password = getUsuario().getContrasena();

String query = "INSERT INTO nomina (fecha\_inicio, fecha\_fin, tipo) VALUES (?,?,?)";

PreparedStatement stmt;

java.sql.Date fechaInicio= **new** java.sql.Date(nomina.getFechaInicio().getTime());

java.sql.Date fechaFin= **new** java.sql.Date(nomina.getFechaFin().getTime());

**try** {

conexion.newConnetionCont(clave, password);

stmt = conexion.getConnection().prepareStatement(query);

stmt.setDate(1, fechaInicio);

stmt.setDate(2, fechaFin);

stmt.setInt(3, nomina.getTipo());

stmt.executeUpdate();

JOptionPane.showMessageDialog(**null**, "Se agrego la nómina");

stmt.close();

} **catch** (SQLException ex) {

JOptionPane.showMessageDialog(**null**, "Verifique sus derechos con el administrador", "¡¡¡ERROR!!!",JOptionPane.WARNING\_MESSAGE);

Logger.getLogger(nominaDaoMysql.class.getName()).log(Level.SEVERE, **null**, ex);

}**finally**{

conexion.Salir();

}

}

@Override

**public** void Actualiza(Nomina nomina) {

**throw** **new** UnsupportedOperationException("Not supported yet."); *//To change body of generated methods, choose Tools | Templates.*

}

@Override

**public** void Elimina(Nomina nomina) {

**throw** **new** UnsupportedOperationException("Not supported yet."); *//To change body of generated methods, choose Tools | Templates.*

}

@Override

**public** List<Nomina> obtenTodos() {

Conexion conexion = **new** Conexion();

conexion.newConnection();

List <Nomina> listaNomina =**new** ArrayList<Nomina>();

String query = "SELECT \* FROM nomina";

PreparedStatement stmt;

**try** {

stmt = conexion.getConnection().prepareStatement(query);

*//stmt.setInt(1,tipo);*

ResultSet rs = stmt.executeQuery();

**while** (rs.next()){

Nomina nomina = **new** Nomina();

nomina.setId(rs.getInt(1));

nomina.setFechaInicio(rs.getDate(2));

nomina.setFechaFin(rs.getDate(3));

nomina.setTipo(rs.getInt(4));

listaNomina.add(nomina);

}

stmt.close();

} **catch** (SQLException ex) {

JOptionPane.showMessageDialog(**null**, "Verifique sus derechos con el administrador", "¡¡¡ERROR!!!",JOptionPane.WARNING\_MESSAGE);

Logger.getLogger(nominaDaoMysql.class.getName()).log(Level.SEVERE, **null**, ex);

}**finally**{

conexion.Salir();

}

**return** listaNomina;

}

@Override

**public** List<Nomina> obtenTipo(int tipo) {

Conexion conexion = **new** Conexion();

conexion.newConnection();

List <Nomina> listaNomina =**new** ArrayList<Nomina>();

String query = "SELECT \* FROM nomina WHERE tipo =?";

PreparedStatement stmt;

**try** {

stmt = conexion.getConnection().prepareStatement(query);

stmt.setInt(1,tipo);

ResultSet rs = stmt.executeQuery();

**while** (rs.next()){

Nomina nomina = **new** Nomina();

nomina.setId(rs.getInt(1));

nomina.setFechaInicio(rs.getDate(2));

nomina.setFechaFin(rs.getDate(3));

nomina.setTipo(rs.getInt(4));

listaNomina.add(nomina);

}

stmt.close();

} **catch** (SQLException ex) {

JOptionPane.showMessageDialog(**null**, "Verifique sus derechos con el administrador", "¡¡¡ERROR!!!",JOptionPane.WARNING\_MESSAGE);

Logger.getLogger(nominaDaoMysql.class.getName()).log(Level.SEVERE, **null**, ex);

}**finally**{

conexion.Salir();

}

**return** listaNomina;

}

@Override

**public** Nomina obtenNomina(int id) {

Conexion conexion = **new** Conexion();

conexion.newConnection();

Nomina nomina = **new** Nomina();

String query = "SELECT \* FROM nomina WHERE id=?";

PreparedStatement stmt;

**try** {

stmt = conexion.getConnection().prepareStatement(query);

stmt.setInt(1,id);

ResultSet rs = stmt.executeQuery();

**while** (rs.next()){

nomina.setId(rs.getInt(1));

nomina.setFechaInicio(rs.getDate(2));

nomina.setFechaFin(rs.getDate(3));

nomina.setTipo(rs.getInt(4));

}

stmt.close();

} **catch** (SQLException ex) {

JOptionPane.showMessageDialog(**null**, "Verifique sus derechos con el administrador", "¡¡¡ERROR!!!",JOptionPane.WARNING\_MESSAGE);

Logger.getLogger(nominaDaoMysql.class.getName()).log(Level.SEVERE, **null**, ex);

}**finally**{

conexion.Salir();

}

**return** nomina;

}

@Override

**public** Usuario getUsuario() {

**return** usuario;

}

@Override

**public** void setUsuario(Usuario usuario) {

**this**.usuario = usuario;

}

}

## nominaEmpleadoDaoMysql

**package** daoMysql;

**import** **Entity.Empleado**;

**import** **Entity.Nomina**;

**import** **Entity.Usuario**;

**import** **Entity.nominaEmpleado**;

**import** **dao.nominaEmpleadoDao**;

**import** **java.sql.PreparedStatement**;

**import** **java.sql.ResultSet**;

**import** **java.sql.SQLException**;

**import** **java.util.ArrayList**;

**import** **java.util.List**;

**import** **java.util.logging.Level**;

**import** **java.util.logging.Logger**;

**import** **javax.swing.JOptionPane**;

*/\*\**

*\**

*\* @author Adan*

*\*/*

**public** **class** **nominaEmpleadoDaoMysql** **implements** nominaEmpleadoDao{

**private** Usuario usuario;

*//Contstante de sueldo minimo*

**private** **static** **final** double threeUMA=241.8;

*/\*\**

*\* Para calcular el efecto de isr y spe*

*\*/*

**public** double obtenerSpe\_Isr(double sueldo, int tipo){

Conexion conexion = **new** Conexion();

conexion.newConnection();

double spe\_isr=0;

String query = "SELECT (s.casMensual-(i.cuotaFija+((?-i.limiteInf)\*i.excedentePorcentaje))) AS SPE\_ISR "

+ "FROM isr i, subsidio s WHERE (i.limiteInf<=? AND i.limiteSup>=?) AND (s.paraIngresos<=? AND s.hastaIngresos>=?)"

+ "AND (i.tipo=? AND s.tipo=?)";

PreparedStatement stmt;

**try** {

stmt = conexion.getConnection().prepareStatement(query);

stmt.setDouble(1, sueldo);

stmt.setDouble(2, sueldo);

stmt.setDouble(3, sueldo);

stmt.setDouble(4, sueldo);

stmt.setDouble(5, sueldo);

stmt.setInt(6, tipo);

stmt.setInt(7, tipo);

ResultSet rs = stmt.executeQuery();

**while**(rs.next()){

spe\_isr = rs.getDouble(1);

}

} **catch** (SQLException ex) {

System.out.println("ERROR EN ISR\_SPE");

Logger.getLogger(nominaDaoMysql.class.getName()).log(Level.SEVERE, **null**, ex);

}

**return** spe\_isr;

}

@Override

**public** void Agrega(nominaEmpleado nominaEmp) {

*//PRIMERO SE HACE LOS CALCULOS*

double sdi = nominaEmp.getEmpleado().getSdi();

double sueldoTrabajo = nominaEmp.getDiasTrabajados()\*nominaEmp.getEmpleado().getSueldo();

double isr\_spe = obtenerSpe\_Isr(sueldoTrabajo,nominaEmp.getNomina().getTipo()); *//Verificar*

double sueldoNeto = 0;

double excedenteSMG = 0;

double prestacionesDinero = 0;

double prestacionesEspecie = 0;

double invalidez = 0;

double cuotaImss = 0;

double censatiaVejez = 0;

**if** (threeUMA < sdi){

excedenteSMG = ((sdi-threeUMA)\*nominaEmp.getDiasTrabajados())\*0.0040;

}

prestacionesDinero = sdi\* nominaEmp.getDiasTrabajados() \* 0.0025;

prestacionesEspecie = sdi\*nominaEmp.getDiasTrabajados() \* 0.00375;

invalidez = sdi\*nominaEmp.getDiasTrabajados()\*0.00625;

cuotaImss = excedenteSMG + prestacionesDinero + prestacionesEspecie + invalidez;

censatiaVejez = sdi\*.01125\*nominaEmp.getDiasTrabajados();

sueldoNeto = sueldoTrabajo + isr\_spe - nominaEmp.getInfonavit() - excedenteSMG - cuotaImss - censatiaVejez;

*//DEDES AQUI SE INSERTA A LA TABLA*

String query = "INSERT INTO NominaEmpleado (idNomina,codigoEmpleado,SueldoDiario,sdi,"

+ "diasTrabajados,spe\_isr, infonavit, cuotaImss,CesantiaVejez,SueldoNeto)"

+ "VALUES (?,?,?,?,?,?,?,?,?,?)";

Conexion conexion = **new** Conexion();

String clave = Integer.toString(getUsuario().getCodigo());

String password = getUsuario().getContrasena();

*//conexion.newConnection();*

PreparedStatement stmt;

**try** {

conexion.newConnetionCont(clave, password);

stmt = conexion.getConnection().prepareStatement(query);

stmt.setInt(1, nominaEmp.getNomina().getId());

stmt.setInt(2, nominaEmp.getEmpleado().getCodigo());

stmt.setDouble(3, nominaEmp.getEmpleado().getSueldo());

stmt.setDouble(4, sdi);

stmt.setInt(5, nominaEmp.getDiasTrabajados());

stmt.setDouble(6, isr\_spe);

stmt.setDouble(7, nominaEmp.getInfonavit());

stmt.setDouble(8, cuotaImss);

stmt.setDouble(9, censatiaVejez);

stmt.setDouble(10, sueldoNeto);

stmt.executeUpdate();

JOptionPane.showMessageDialog(**null**, "Se añadio el emplado en la nómina");

stmt.close();

} **catch** (SQLException ex) {

JOptionPane.showMessageDialog(**null**, "Verifique sus derechos con el administrador", "¡¡¡ERROR!!!",JOptionPane.WARNING\_MESSAGE);

Logger.getLogger(nominaDaoMysql.class.getName()).log(Level.SEVERE, **null**, ex);

}**finally**{

conexion.Salir();

}

}

@Override

**public** void Actualiza(nominaEmpleado nominaEmp) {

Conexion conexion = **new** Conexion();

*//PRIMERO SE HACE LOS CALCULOS*

double sdi = nominaEmp.getEmpleado().getSdi();

double sueldoTrabajo = nominaEmp.getDiasTrabajados()\*nominaEmp.getEmpleado().getSueldo();

double isr\_spe = obtenerSpe\_Isr(sueldoTrabajo,nominaEmp.getNomina().getTipo()); *//Verificar*

double sueldoNeto = 0;

double excedenteSMG = 0;

double prestacionesDinero = 0;

double prestacionesEspecie = 0;

double invalidez = 0;

double cuotaImss = 0;

double censatiaVejez = 0;

**if** (threeUMA < sdi){

excedenteSMG = ((sdi-threeUMA)\*nominaEmp.getDiasTrabajados())\*0.0040;

}

prestacionesDinero = sdi\* nominaEmp.getDiasTrabajados() \* 0.0025;

prestacionesEspecie = sdi\*nominaEmp.getDiasTrabajados() \* 0.00375;

invalidez = sdi\*nominaEmp.getDiasTrabajados()\*0.00625;

cuotaImss = excedenteSMG + prestacionesDinero + prestacionesEspecie + invalidez;

censatiaVejez = sdi\*.01125\*nominaEmp.getDiasTrabajados();

sueldoNeto = sueldoTrabajo + isr\_spe - nominaEmp.getInfonavit() - excedenteSMG - cuotaImss - censatiaVejez;

*//DESDE AQUI ACTUALIZA*

String clave = Integer.toString(getUsuario().getCodigo());

String password = getUsuario().getContrasena();

String query = "UPDATE nominaEmpleado SET diasTrabajados=?, infonavit=?, cuotaImss=?, CesantiaVejez=?, spe\_isr=?, sueldoNeto=?"

+ " WHERE id=?";

PreparedStatement stmt;

System.out.println(nominaEmp.getId());

**try** {

conexion.newConnetionCont(clave , password );

stmt = conexion.getConnection().prepareStatement(query);

stmt.setInt(1, nominaEmp.getDiasTrabajados());

stmt.setDouble(2, nominaEmp.getInfonavit());

stmt.setDouble(3, cuotaImss);

stmt.setDouble(4, censatiaVejez);

stmt.setDouble(5, isr\_spe);

stmt.setDouble(6, sueldoNeto);

stmt.setInt(7, nominaEmp.getId());

stmt.executeUpdate();

JOptionPane.showMessageDialog(**null**, "Se ha actualizado. Vuelva a cargar la tabla");

} **catch** (SQLException ex) {

JOptionPane.showMessageDialog(**null**, "Verifique sus derechos con el administrador", "¡¡¡ERROR!!!",JOptionPane.WARNING\_MESSAGE);

Logger.getLogger(nominaEmpleadoDaoMysql.class.getName()).log(Level.SEVERE, **null**, ex);

}

}

@Override

**public** void Elimina(int id) {

String query = "DELETE FROM nominaEmpleado WHERE id =?";

Conexion conexion = **new** Conexion();

String clave = Integer.toString(getUsuario().getCodigo());

String password = getUsuario().getContrasena();

PreparedStatement stmt;

**try** {

conexion.newConnetionCont(clave, password);

stmt = conexion.getConnection().prepareStatement(query);

stmt.setInt(1, id);

stmt.executeUpdate();

stmt.close();

} **catch** (SQLException ex) {

JOptionPane.showMessageDialog(**null**, "Verifique sus derechos con el administrador", "¡¡¡ERROR!!!",JOptionPane.WARNING\_MESSAGE);

Logger.getLogger(nominaEmpleadoDaoMysql.class.getName()).log(Level.SEVERE, **null**, ex);

}**finally**{

conexion.Salir();

}

}

@Override

**public** List<nominaEmpleado> obtenTodos(int tipo) {

Conexion conexion = **new** Conexion();

conexion.newConnection();

List <nominaEmpleado> nominaList = **new** ArrayList();

String query = "SELECT ne.\*,n.fecha\_inicio,n.fecha\_fin,n.tipo, e.nombre, e.apellidoP, e.apellidoM "

+ "FROM nominaEmpleado ne "

+ "INNER JOIN empleado e ON ne.codigoEmpleado=e.codigo "

+ "INNER JOIN nomina n ON ne.idNomina = n.id WHERE n.tipo =?";

PreparedStatement stmt;

**try** {

stmt = conexion.getConnection().prepareStatement(query);

stmt.setInt(1, tipo);

ResultSet rs = stmt.executeQuery();

**while** (rs.next()){

nominaEmpleado nominaEmp = **new** nominaEmpleado();

Nomina nomina = **new** Nomina();

Empleado empleado = **new** Empleado();

*//Obtener los datos de la nomina*

nomina.setId(2);

nomina.setFechaInicio(rs.getDate(13));

nomina.setFechaFin(rs.getDate(14));

nomina.setTipo(15);

*//obtener los datos de empleado*

empleado.setCodigo(3);

empleado.setSueldo(rs.getDouble(4));

empleado.setSdi(rs.getDouble(5));

empleado.setNombre(rs.getString(16));

empleado.setApellidoP(rs.getString(17));

empleado.setApellidoM(rs.getString(18));

*//obtener los datos de la nomina del empleado*

nominaEmp.setId(rs.getInt(1));

nominaEmp.setEmpleado(empleado);

nominaEmp.setNomina(nomina);

nominaEmp.setDiasTrabajados(rs.getInt(6));

nominaEmp.setSueldoTrabajo(rs.getDouble(7));

nominaEmp.setSpe\_isr(rs.getDouble(8));

nominaEmp.setInfonavit(rs.getDouble(9));

nominaEmp.setCuotaImss(rs.getDouble(10));

nominaEmp.setCensantiaVejez(rs.getDouble(11));

nominaEmp.setSueldoNeto(rs.getDouble(12));

nominaList.add(nominaEmp);

}

} **catch** (SQLException ex) {

JOptionPane.showMessageDialog(**null**, "Verifique sus derechos con el administrador", "¡¡¡ERROR!!!",JOptionPane.WARNING\_MESSAGE);

Logger.getLogger(nominaEmpleadoDaoMysql.class.getName()).log(Level.SEVERE, **null**, ex);

}

**return** nominaList;

}

@Override

**public** List<nominaEmpleado> registroNominaEmpleado(int tipo) {

**throw** **new** UnsupportedOperationException("Not supported yet."); *//To change body of generated methods, choose Tools | Templates.*

}

@Override

**public** List<nominaEmpleado> obtenNominaEmpleado(int id) {

Conexion conexion = **new** Conexion();

conexion.newConnection();

List <nominaEmpleado> nominaList = **new** ArrayList();

String query = "SELECT ne.\*,n.fecha\_inicio,n.fecha\_fin,n.tipo, e.nombre, e.apellidoP, e.apellidoM "

+ "FROM nominaEmpleado ne "

+ "INNER JOIN empleado e ON ne.codigoEmpleado=e.codigo "

+ "INNER JOIN nomina n ON ne.idNomina = n.id WHERE e.codigo=?";

PreparedStatement stmt;

**try** {

stmt = conexion.getConnection().prepareStatement(query);

stmt.setInt(1, id);

ResultSet rs = stmt.executeQuery();

**while** (rs.next()){

nominaEmpleado nominaEmp = **new** nominaEmpleado();

Nomina nomina = **new** Nomina();

Empleado empleado = **new** Empleado();

*//Obtener los datos de la nomina*

nomina.setId(2);

nomina.setFechaInicio(rs.getDate(13));

nomina.setFechaFin(rs.getDate(14));

nomina.setTipo(15);

*//obtener los datos de empleado*

empleado.setCodigo(3);

empleado.setSueldo(rs.getDouble(4));

empleado.setSdi(rs.getDouble(5));

empleado.setNombre(rs.getString(16));

empleado.setApellidoP(rs.getString(17));

empleado.setApellidoM(rs.getString(18));

*//obtener los datos de la nomina del empleado*

nominaEmp.setId(rs.getInt(1));

nominaEmp.setEmpleado(empleado);

nominaEmp.setNomina(nomina);

nominaEmp.setDiasTrabajados(rs.getInt(6));

nominaEmp.setSueldoTrabajo(rs.getDouble(7));

nominaEmp.setSpe\_isr(rs.getDouble(8));

nominaEmp.setInfonavit(rs.getDouble(9));

nominaEmp.setCuotaImss(rs.getDouble(10));

nominaEmp.setCensantiaVejez(rs.getDouble(11));

nominaEmp.setSueldoNeto(rs.getDouble(12));

nominaList.add(nominaEmp);

}

} **catch** (SQLException ex) {

JOptionPane.showMessageDialog(**null**, "Verifique sus derechos con el administrador", "¡¡¡ERROR!!!",JOptionPane.WARNING\_MESSAGE);

Logger.getLogger(nominaEmpleadoDaoMysql.class.getName()).log(Level.SEVERE, **null**, ex);

}

**return** nominaList;

}

@Override

**public** List<nominaEmpleado> obtenNominaFecha(int idNomina) {

Conexion conexion = **new** Conexion();

conexion.newConnection();

List <nominaEmpleado> nominaList = **new** ArrayList();

String query = "SELECT ne.\*,n.fecha\_inicio,n.fecha\_fin,n.tipo, e.nombre, e.apellidoP, e.apellidoM "

+ "FROM nominaEmpleado ne "

+ "INNER JOIN empleado e ON ne.codigoEmpleado=e.codigo "

+ "INNER JOIN nomina n ON ne.idNomina = n.id WHERE n.id=?";

PreparedStatement stmt;

**try** {

stmt = conexion.getConnection().prepareStatement(query);

stmt.setInt(1, idNomina);

ResultSet rs = stmt.executeQuery();

**while** (rs.next()){

nominaEmpleado nominaEmp = **new** nominaEmpleado();

Nomina nomina = **new** Nomina();

Empleado empleado = **new** Empleado();

*//Obtener los datos de la nomina*

nomina.setId(2);

nomina.setFechaInicio(rs.getDate(13));

nomina.setFechaFin(rs.getDate(14));

nomina.setTipo(15);

*//obtener los datos de empleado*

empleado.setCodigo(3);

empleado.setSueldo(rs.getDouble(4));

empleado.setSdi(rs.getDouble(5));

empleado.setNombre(rs.getString(16));

empleado.setApellidoP(rs.getString(17));

empleado.setApellidoM(rs.getString(18));

*//obtener los datos de la nomina del empleado*

nominaEmp.setId(rs.getInt(1));

nominaEmp.setEmpleado(empleado);

nominaEmp.setNomina(nomina);

nominaEmp.setDiasTrabajados(rs.getInt(6));

nominaEmp.setSueldoTrabajo(rs.getDouble(7));

nominaEmp.setSpe\_isr(rs.getDouble(8));

nominaEmp.setInfonavit(rs.getDouble(9));

nominaEmp.setCuotaImss(rs.getDouble(10));

nominaEmp.setCensantiaVejez(rs.getDouble(11));

nominaEmp.setSueldoNeto(rs.getDouble(12));

nominaList.add(nominaEmp);

}

} **catch** (SQLException ex) {

JOptionPane.showMessageDialog(**null**, "Verifique sus derechos con el administrador", "¡¡¡ERROR!!!",JOptionPane.WARNING\_MESSAGE);

Logger.getLogger(nominaEmpleadoDaoMysql.class.getName()).log(Level.SEVERE, **null**, ex);

}

**return** nominaList;

}

@Override

**public** Usuario getUsuario() {

**return** usuario;

}

@Override

**public** void setUsuario(Usuario usuario) {

**this**.usuario = usuario;

}

@Override

**public** List<nominaEmpleado> obtenNominaEmpleadoFecha(int idNomina, int idEmpleado) {

Conexion conexion = **new** Conexion();

conexion.newConnection();

List <nominaEmpleado> nominaList = **new** ArrayList();

String query = "SELECT ne.\*,n.fecha\_inicio,n.fecha\_fin,n.tipo, e.nombre, e.apellidoP, e.apellidoM "

+ "FROM nominaEmpleado ne "

+ "INNER JOIN empleado e ON ne.codigoEmpleado=e.codigo "

+ "INNER JOIN nomina n ON ne.idNomina = n.id WHERE n.id=? AND e.codigo=?";

PreparedStatement stmt;

**try** {

stmt = conexion.getConnection().prepareStatement(query);

stmt.setInt(1, idNomina);

stmt.setInt(2, idEmpleado);

ResultSet rs = stmt.executeQuery();

**while** (rs.next()){

nominaEmpleado nominaEmp = **new** nominaEmpleado();

Nomina nomina = **new** Nomina();

Empleado empleado = **new** Empleado();

*//Obtener los datos de la nomina*

nomina.setId(2);

nomina.setFechaInicio(rs.getDate(13));

nomina.setFechaFin(rs.getDate(14));

nomina.setTipo(15);

*//obtener los datos de empleado*

empleado.setCodigo(3);

empleado.setSueldo(rs.getDouble(4));

empleado.setSdi(rs.getDouble(5));

empleado.setNombre(rs.getString(16));

empleado.setApellidoP(rs.getString(17));

empleado.setApellidoM(rs.getString(18));

*//obtener los datos de la nomina del empleado*

nominaEmp.setId(rs.getInt(1));

nominaEmp.setEmpleado(empleado);

nominaEmp.setNomina(nomina);

nominaEmp.setDiasTrabajados(rs.getInt(6));

nominaEmp.setSueldoTrabajo(rs.getDouble(7));

nominaEmp.setSpe\_isr(rs.getDouble(8));

nominaEmp.setInfonavit(rs.getDouble(9));

nominaEmp.setCuotaImss(rs.getDouble(10));

nominaEmp.setCensantiaVejez(rs.getDouble(11));

nominaEmp.setSueldoNeto(rs.getDouble(12));

nominaList.add(nominaEmp);

}

} **catch** (SQLException ex) {

JOptionPane.showMessageDialog(**null**, "Verifique sus derechos con el administrador", "¡¡¡ERROR!!!",JOptionPane.WARNING\_MESSAGE);

Logger.getLogger(nominaEmpleadoDaoMysql.class.getName()).log(Level.SEVERE, **null**, ex);

}

**return** nominaList;

}

}

## usuarioDaoMysql

**package** daoMysql;

**import** **Entity.Empleado**;

**import** **Entity.Usuario**;

**import** **dao.usuarioDao**;

**import** **java.sql.PreparedStatement**;

**import** **java.sql.ResultSet**;

**import** **java.sql.SQLException**;

**import** **java.util.ArrayList**;

**import** **java.util.List**;

**import** **java.util.logging.Level**;

**import** **java.util.logging.Logger**;

**import** **javax.swing.JOptionPane**;

*/\*\**

*\**

*\* @author Adan*

*\*/*

**public** **class** **usuarioDaoMysql** **implements** usuarioDao{

@Override

**public** Usuario iniciaSesion(int codigo, String contrasena) {

boolean entrar = **false**;

Conexion conexion = **new** Conexion();

String query = "SELECT u.\*, e.nombre, e.apellidoP, e.apellidoM FROM usuario u "

+ "INNER JOIN empleado e ON u.codigo=e.codigo WHERE u.codigo = ?";

Usuario usuario = **new** Usuario();

PreparedStatement stmt;

**try** {

conexion.newConnetionCont(Integer.toString(codigo), contrasena);

stmt = conexion.getConnection().prepareStatement(query);

stmt.setInt(1, codigo);

ResultSet rs = stmt.executeQuery();

int i = 0;

**while** (rs.next()){

i++;

*//EL 1 SE IGNORA*

usuario.setCodigo(rs.getInt(2));

usuario.setRole(rs.getString(3));

usuario.setNombre(rs.getString(4));

usuario.setApellidoP(rs.getString(5));

usuario.setApellidoM(rs.getString(6));

}

**if** (i == 0){

usuario.setRole("");

}

} **catch** (SQLException ex) {

Logger.getLogger(usuarioDaoMysql.class.getName()).log(Level.SEVERE, **null**, ex);

}

**return** usuario;

}

@Override

**public** void agrega(Usuario usuario) {

Conexion conexion= **new** Conexion();

conexion.newConnection();

PreparedStatement stmt;

PreparedStatement stmtII;

**try**{

String query = "CREATE USER ?@'localhost' IDENTIFIED BY ? ";

stmt = conexion.getConnection().prepareStatement(query);

stmt.setString(1, Integer.toString(usuario.getCodigo()));

stmt.setString(2, usuario.getContrasena());

int consulta = stmt.executeUpdate();

String ingresa = "INSERT INTO usuario (codigo,role) VALUES (?,?)";

stmtII = conexion.getConnection().prepareStatement(ingresa);

stmtII.setInt(1, usuario.getCodigo());

stmtII.setString(2, usuario.getRole());

stmtII.executeUpdate();

**if** (consulta > 0){

JOptionPane.showMessageDialog(**null**, "Se ha registrado nuevo usuario");

}

stmt.close();

stmtII.close();

System.out.println("Registro");

}**catch**(SQLException ex){

Logger.getLogger(usuarioDaoMysql.class.getName()).log(Level.SEVERE, **null**, ex);

}

}

@Override

**public** void Actualiza(Usuario usuario) {

Conexion conexion= **new** Conexion();

**try**{

conexion.newConnection();

PreparedStatement stmt = **null**;

String query;

**if**(!usuario.getRole().equals("ADMIN")){

**for** (int i=0; i<4; i++){

**if** (!usuario.cuoPriv[i].equals("")){

query = "GRANT "+usuario.cuoPriv[i]+" ON nomina.cuota TO ?@localhost";

stmt = conexion.getConnection().prepareStatement(query);

stmt.setString(1, Integer.toString(usuario.getCodigo()));

stmt.executeUpdate();

}

**if** (!usuario.empPriv[i].equals("")){

query = "GRANT "+usuario.empPriv[i]+" ON nomina.empleado TO ?@localhost";

stmt = conexion.getConnection().prepareStatement(query);

stmt.setString(1, Integer.toString(usuario.getCodigo()));

stmt.executeUpdate();

}

**if** (!usuario.nomPriv[i].equals("")){

query = "GRANT "+usuario.nomPriv[i]+" ON nomina.nomina TO ?@localhost";

stmt = conexion.getConnection().prepareStatement(query);

stmt.setString(1, Integer.toString(usuario.getCodigo()));

stmt.executeUpdate();

query = "GRANT "+usuario.nomPriv[i]+" ON nomina.nominaEmpleado TO ?@localhost";

stmt = conexion.getConnection().prepareStatement(query);

stmt.setString(1, Integer.toString(usuario.getCodigo()));

stmt.executeUpdate();

}

}

query = "GRANT SELECT ON nomina.usuario TO ?@localhost";

stmt = conexion.getConnection().prepareStatement(query);

stmt.setString(1, Integer.toString(usuario.getCodigo()));

stmt.executeUpdate();

}**else**{

query = "GRANT ALL ON nomina.\* TO ?@localhost";

stmt = conexion.getConnection().prepareStatement(query);

stmt.setString(1, Integer.toString(usuario.getCodigo()));

stmt.executeUpdate();

}

query= "FLUSH PRIVILEGES";

stmt = conexion.getConnection().prepareStatement(query);

stmt.executeUpdate();

stmt.close();

}**catch**(SQLException ex){

Logger.getLogger(usuarioDaoMysql.class.getName()).log(Level.SEVERE, **null**, ex);

}**finally**{

conexion.Salir();

}

}

@Override

**public** void Elimina(Usuario usuario) {

Conexion conexion = **new** Conexion();

conexion.newConnection();

PreparedStatement stmt;

**try** {

String query= "DROP USER ?@'localhost'";

stmt = conexion.getConnection().prepareStatement(query);

stmt.setString(1, Integer.toString(usuario.getCodigo()));

stmt.executeUpdate();

query = "DELETE FROM usuario WHERE codigo = ?";

stmt = conexion.getConnection().prepareStatement(query);

stmt.setInt(1, usuario.getCodigo());

stmt.executeUpdate();

query= "FLUSH PRIVILEGES";

stmt = conexion.getConnection().prepareStatement(query);

stmt.executeUpdate();

stmt.close();

} **catch** (SQLException ex) {

Logger.getLogger(usuarioDaoMysql.class.getName()).log(Level.SEVERE, **null**, ex);

}

}

@Override

**public** List<Usuario> obtenTodos() {

Conexion conexion = **new** Conexion();

conexion.newConnection();

List <Usuario> listaUsuario = **new** ArrayList<Usuario>();

String query = "SELECT e.codigo, e.nombre, e.apellidoP, e.apellidoM, u.role "

+ "FROM usuario u INNER JOIN empleado e ON u.codigo=e.codigo WHERE u.role != 'ADMIN'";

PreparedStatement stmt;

**try** {

stmt=conexion.getConnection().prepareStatement(query);

ResultSet rs = stmt.executeQuery();

**while** (rs.next()){

Usuario usuario = **new** Usuario();

usuario.setCodigo(rs.getInt(1));

usuario.setNombre(rs.getString(2));

usuario.setApellidoP(rs.getString(3));

usuario.setApellidoM(rs.getString(4));

usuario.setRole(rs.getString(5));

listaUsuario.add(usuario);

}

} **catch** (SQLException ex) {

Logger.getLogger(usuarioDaoMysql.class.getName()).log(Level.SEVERE, **null**, ex);

}

**return** listaUsuario;

}

@Override

**public** List<Usuario> obtenUsuarioAdmin() {

Conexion conexion = **new** Conexion();

conexion.newConnection();

List <Usuario> listaUsuario = **new** ArrayList<Usuario>();

String query = "SELECT e.codigo, e.nombre, e.apellidoP, e.apellidoM, u.role "

+ "FROM usuario u INNER JOIN empleado e ON u.codigo=e.codigo WHERE u.role = 'ADMIN'";

PreparedStatement stmt;

**try** {

stmt=conexion.getConnection().prepareStatement(query);

ResultSet rs = stmt.executeQuery();

**while** (rs.next()){

Usuario usuario = **new** Usuario();

usuario.setCodigo(rs.getInt(1));

usuario.setNombre(rs.getString(2));

usuario.setApellidoP(rs.getString(3));

usuario.setApellidoM(rs.getString(4));

usuario.setRole(rs.getString(5));

listaUsuario.add(usuario);

}

} **catch** (SQLException ex) {

Logger.getLogger(usuarioDaoMysql.class.getName()).log(Level.SEVERE, **null**, ex);

}

**return** listaUsuario;

}

@Override

**public** void quitarPrivilegios(Usuario usuario) {

Conexion conexion = **new** Conexion();

conexion.newConnection();

PreparedStatement stmt;

**try** {

String query;

query= "REVOKE ALL ON nomina.empleado FROM '"+usuario.getCodigo()+"'@'localhost'";

stmt = conexion.getConnection().prepareStatement(query);

stmt.executeUpdate();

query= "REVOKE ALL ON nomina.nomina FROM '"+usuario.getCodigo()+"'@'localhost'";

stmt = conexion.getConnection().prepareStatement(query);

stmt.executeUpdate();

query= "REVOKE ALL ON nomina.nominaEmpleado FROM '"+usuario.getCodigo()+"'@'localhost'";

stmt = conexion.getConnection().prepareStatement(query);

stmt.executeUpdate();

query= "REVOKE ALL ON nomina.cuota FROM '"+usuario.getCodigo()+"'@'localhost'";

stmt = conexion.getConnection().prepareStatement(query);

stmt.executeUpdate();

String queryF = "FLUSH PRIVILEGES";

stmt = conexion.getConnection().prepareStatement(queryF);

stmt.close();

} **catch** (SQLException ex) {

Logger.getLogger(usuarioDaoMysql.class.getName()).log(Level.SEVERE, **null**, ex);

}

}

@Override

**public** void agregaTodos(Usuario usuario) {

**throw** **new** UnsupportedOperationException("Not supported yet."); *//To change body of generated methods, choose Tools | Templates.*

}

@Override

**public** void otorgaTodosLosDerechos(Usuario usuario) {

String queery = "GRANT ALL ON nomina.\* TO ?@'localhost'";

}

@Override

**public** List<Empleado> obteneEmpleados() {

Conexion conexion = **new** Conexion();

conexion.newConnection();

List <Empleado> listaEmpleado = **new** ArrayList<Empleado>();

String query = "SELECT e.\* FROM empleado e LEFT JOIN usuario u ON e.codigo = u.codigo WHERE u.codigo IS NULL";

PreparedStatement stmt;

**try** {

stmt=conexion.getConnection().prepareStatement(query);

ResultSet rs = stmt.executeQuery();

**while** (rs.next()){

Empleado empleado = **new** Empleado();

empleado.setCodigo(rs.getInt(1));

empleado.setNombre(rs.getString(2));

empleado.setApellidoP(rs.getString(3));

empleado.setApellidoM(rs.getString(4));

empleado.setFehcaIngreso(rs.getDate(5));

empleado.setSueldo(rs.getDouble(6));

listaEmpleado.add(empleado);

}

} **catch** (SQLException ex) {

Logger.getLogger(usuarioDaoMysql.class.getName()).log(Level.SEVERE, **null**, ex);

}

**return** listaEmpleado;

}

}