**INGENIERÍA EN SISTEMAS COMPUTACIONALES**

**INTEGRANTES**

CHRISTIAN GUADALUPE CARRERA SALAS

GUSTAVO ÁNGEL CASTILLO ROJAS

LUIS DAVID HERNÁNDEZ PEREZ

JOSUÉ ISRAEL HUERTA CASTILLO

**MATERIA Y HORA**

INGENIERÍA DE SOFTWARE / 9:00-10:00 HR.

**PROFESORA**

EURI SALGADO ESCOBAR

**MANUAL TÉCNICO DEL SISTEMA DE CONTROL DE EQUIPO DE CÓMPUTO**

# **Introducción**

El presente manual muestra el desarrollo del sistema para el control de equipo de cómputo. Se detallan los aspectos técnicos para un correcto entendimiento.

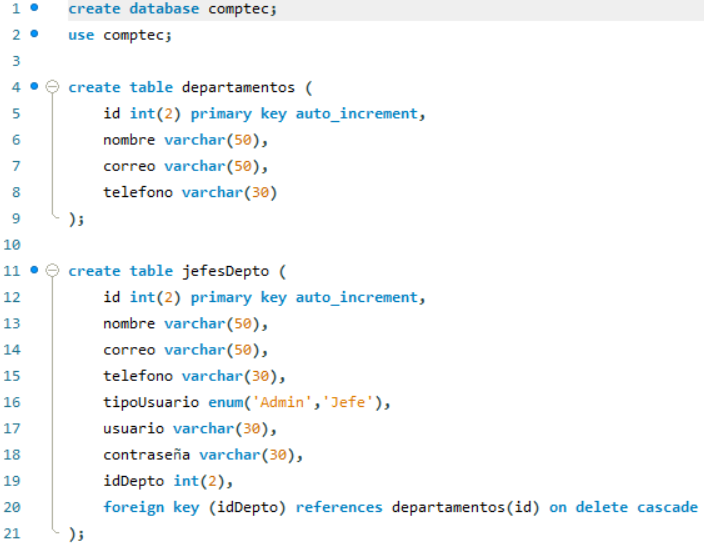
# **Base de datos**

Para poder almacenar toda la información es necesario que nuestro sistema tenga una base de datos. En este caso se trabajó con MySQL ya que es el con el que estamos más familiarizados y es muy sencillo utilizarlo. A continuación, se muestran las tablas y procedimientos almacenados que fueron necesarios para el desarrollo del sistema.

## **Tablas**

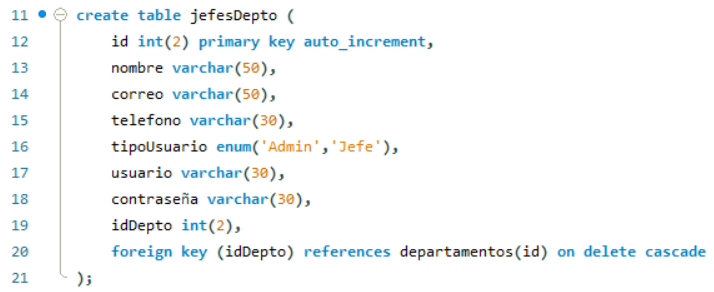
### **Departamentos**

Dentro de esta tabla como su nombre lo indica, se almacenará la información de los distintos departamentos existentes en el Tec. Madero. Podemos ver que se compone de varios atributos como el id, que, en este caso, será nuestra clave primaria y se ha definido como auto incrementable, con lo cual, al momento de realizar alguna inserción ya no será necesario definirlo porque se hará automáticamente. También tenemos otros atributos como el nombre, correo y teléfono.



### **JefesDepto**

Otra de las cosas que se deben almacenar son los jefes de departamento. En esta tabla observamos que se tiene un Id, el cual es nuestra llave primaria y es también auto incrementable, se tienen los datos del jefe como el nombre, correo, teléfono, un tipoUsuario que nos permitirá definir si es un administrador o un jefe, ya que cada uno tiene diferentes permisos en el sistema. De igual forma, se tiene un usuario y contraseña que servirán para poder iniciar sesión, y por último, un idDepto que utilizaremos para referirnos al departamento del que es jefe un determinado usuario. En el atributo idDepto se le añade "on delete cascade”, esto hace que si borramos un departamento también se eliminará la información de su jefe.



### **Equipos**

Si el sistema es para el control de equipo de cómputo, obviamente será necesario almacenar la información de los equipos. Se tienen como atributos, el id, número de inventario, número de serie, modelo, procesador, disco duro, estado, el cual puede ser bueno, regular o defectuoso. Por último, el id del departamento al que pertenece el equipo, también se le agregó “on delete cascade”, por lo cual, si eliminamos un departamento, de igual forma se eliminaran la información de los equipos que están asociados a él.



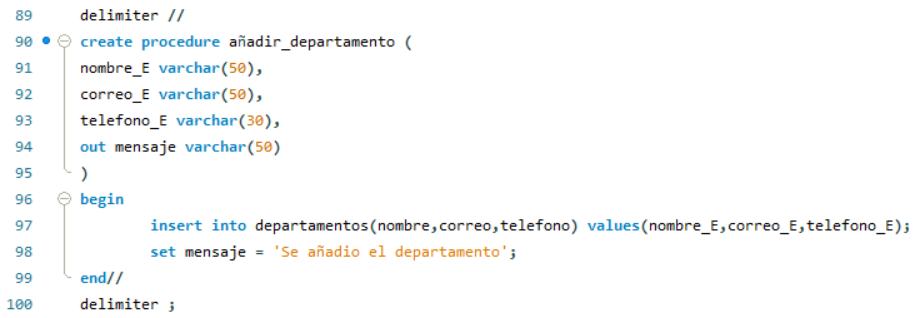
## **Procedimientos almacenados**

Los procedimientos almacenados son como los métodos que utilizamos en los lenguajes de programación. Reciben una serie de parámetros que utilizan para realizar un conjunto de acciones específicas.

Para poder realizar operaciones como mostrar los registros de alguna tabla, añadir, modificar o eliminar registros, desde una aplicación externa al SGBD, es necesario la creación de estos procedimientos almacenados.

En nuestro proyecto hemos definido diversos procedimientos que serán utilizados en las interfaces del sistema, para llevar a cabo las operaciones mencionadas anteriormente. A continuación, se muestran algunos de ellos.

En este procedimiento para añadir departamentos, se reciben los atributos que conforman la tabla de departamentos, a excepción del id, ya que se asigna automáticamente. También vemos que hay un parámetro de salida (out), el cual es un mensaje, el cual nos servirá para mostrar la notificación correspondiente al realizar la operación. Después tenemos el cuerpo del procedimiento, el cual especifica las operaciones que se realizarán.



Este otro procedimiento almacenado es para la modificación de los datos de un jefe de departamento. Se reciben los parámetros que ocuparemos y pasamos a la parte del cuerpo, en la cual podemos ver que se utilizan condicionales if, esto con la finalidad de verificar que el jefe este registrado en el sistema y que el id del departamento sea válido. Si se cumplen estas condiciones, se efectuarán las operaciones, de otra manera, se mostrará el mensaje correspondiente.



Se deben crear los procedimientos de mostrar registros, buscar, insertar, modificar y eliminar para las interfaces de departamentos, jefes de departamento y equipos. También existe otra interfaz que es la de reportes, para ello, ocuparemos procedimientos para generar un reporte de todos los equipos registrados o dependiendo el estado que seleccionemos.

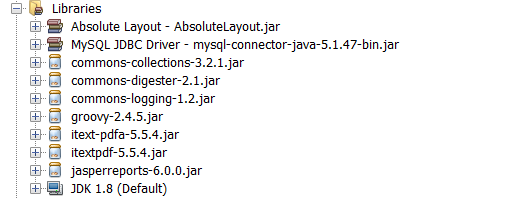
# **Sistema**

Una vez explicada la parte de la base de datos, ahora es el turno de hablar sobre cómo fue el proceso de creación del sistema. Antes que nada, el sistema está programado con Java y se utilizó el IDE NetBeans, ya que es lo que hemos venido manejando durante estos semestres de la carrera y es muy cómodo trabajar con él.

A continuación, se explican las diversas clases que componen el sistema, las librerías que fueron necesarias y la parte de la elaboración de reportes.

## **Librerías necesarias**

Al realizar una aplicación que implemente una base de datos, es necesario añadir el conector, en este caso es el mysql-connector-java-5.1.47-bin.jar, este nos permitirá utilizar instrucciones provenientes de la clase SQL para Java. Las otras librerías que se pueden observar, se ocuparán para la creación de los reportes utilizando el plugin JasperReports. Eso lo veremos más adelante.



## **Clases del sistema**

### **ConexiónBD.java**

Como su nombre lo indica, en esta clase se establece la conexión con la base de datos. Se crea un objeto connection que utilizaremos para asignarle la ruta de nuestra base de datos y que se pueda generar la conexión entre nuestro sistema y la base de datos. La parte de la ruta que dice “root”, “1234”, corresponde al usuario y contraseña de nuestra BD, cada quien podría tener unos diferentes, así que hay que asegurarse de que sea el correcto para evitar fallos.



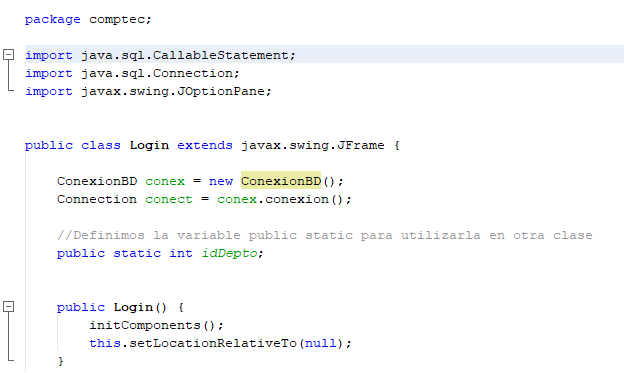
### **Login.java**

Las interfaces del sistema están construidas con JFrames, los cuales tienen una parte que corresponde al diseño de la ventana y otra al código o funcionalidad. Nuestra primera interfaz es la del login o inicio de sesión.

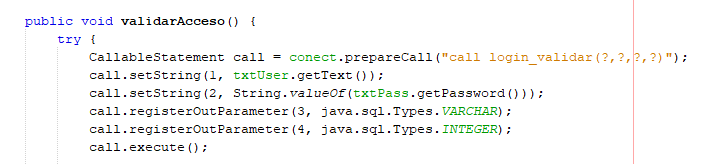
En la parte del diseño tenemos que la interfaz se compone de 2 labels que utilizamos para los textos de usuario y contraseña, 2 TextField para que podamos ingresar los datos y un botón, el cual, al presionarlo mandará llamar al método de validarAcceso() para permitirnos o no, el acceso al sistema.



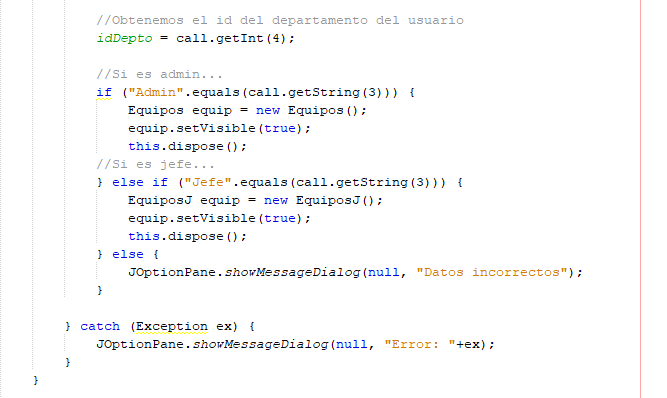
En la parte del código primeramente se declaran los objetos de la conexión que definimos en la clase anterior, además, creamos una variable llamada idDepto que nos servirá para capturar ese atributo del usuario con el que iniciemos sesión y de esta manera poder mostrarle solo lo que tiene permitido ver.



Se creó un método que nos permite validar el acceso al ingresar los datos de algún usuario. En la primera instrucción se crea el objeto call de la clase CallableStatement, con este objeto podremos llamar al procedimiento almacenado correspondiente. Entre comillas se escribe call nombreProcedimiento (?,?,?), esos signos de interrogación corresponden a la cantidad de parámetros tanto de entrada como de salida que ocupe el procedimiento. Después, procedemos a ir registrando los parámetros, en este caso los valores se obtendrán de las 2 cajas de texto de usuario y contraseña. Finalmente, ejecutamos el procedimiento.



Capturamos el idDepto, utilizando la información que nos otorga el 4to parámetro del procedimiento que ejecutamos. Dicho procedimiento al momento de proporcionarle los datos de algún usuario, nos devuelve el tipo de usuario que es y su idDepto. Con el tipo de usuario haremos la validación, si es “Admin” se mostrarán las interfaces donde podemos llevar a cabo operaciones de inserción, modificación o eliminación. Si es “Jefe” se mostraran las interfaces con el mismo diseño que las de administrador pero no será posible realizar las 3 operaciones ya mencionadas.



Finalmente, mediante un evento ActionPerformed que le hemos añadido al botón de iniciar sesión, se ejecutará el método de validarAcceso(), siempre y cuando no este vacío alguno de los 2 campos, usuario o contraseña. Si falta algún dato, se nos notificará.

