

# Proyecto Sistema de facturación de Foodtrucks

Por: Jhonathan Pizarra y Bryan Loachamin

**Abstract**—Billing systems are found in the food businesses. In recent years, food businesses called foodtrucks have increased exponentially, however, it is important the computerized management of business billing transactions, both with users and with suppliers. It is necessary to design a database that will help us with this topic

**Resumen**—Los Sistemas de facturación se encuentran en los negocios de comidas. En los últimos años, los negocios de comida denominados foodtrucks se han incrementado exponencialmente, sin embargo, es importante el manejo informático de las transacciones de facturación del negocio, tanto con los usuarios como con los proveedores. Se necesita diseñar una base de datos que nos ayude con esta temática.

## I. INTRODUCTION

POWER DESIGNER es una herramienta para el análisis, diseño inteligente y construcción sólida de una base de datos y un desarrollo orientado a modelos de datos a nivel físico y conceptual, que da a los desarrolladores Cliente/Servidor la más firme base para aplicaciones de alto rendimiento. Existen además, otras herramientas que usaré para crear el proyecto entre ellas será SQL Server, y NetBeans, lo que se pretende hacer con el presente proyecto es enlazar conocimientos adquiridos de la materia “Bases de Datos I” y “Programación Avanzada” para crear una aplicación que nos ayude a resolver el problema propuesto, “Un sistema de facturación” Rápidamente explicaré el uso y características de cada programa en este informe, aun así es posible que sienta “lagunas” y estaría en lo cierto, porque el presente informe no pretende enseñar conocimientos básicos de Java o PowerDesigner, más bien pretende enseñar cómo usar esos programas en este proyecto. [1]

## II. POWER DESIGNER

PowerDesigner es una herramienta para el análisis, diseño inteligente y construcción sólida de una base de datos y un desarrollo orientado a modelos de datos a nivel físico y conceptual, que da a los desarrolladores Cliente/Servidor la más firme base para aplicaciones de alto rendimiento. Comenzamos por crear un nuevo Modelo, para eso abrimos el programa y damos clic en la pestaña “File” y escogemos la opción “Nuevo Modelo”. Obsérvese la figura 1.

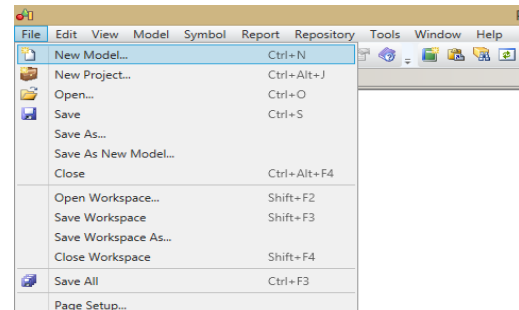


Fig. 1 Creación de modelos

Escogemos en la sección información la opción “Conceptual Data” y enseguida le damos un nombre a nuestro modelo. Lo que habremos hecho es generar un espacio de trabajo el cual contendrá un diagrama. Pero antes nos es necesario hacer todo tal y como se pide en la Figura 2.

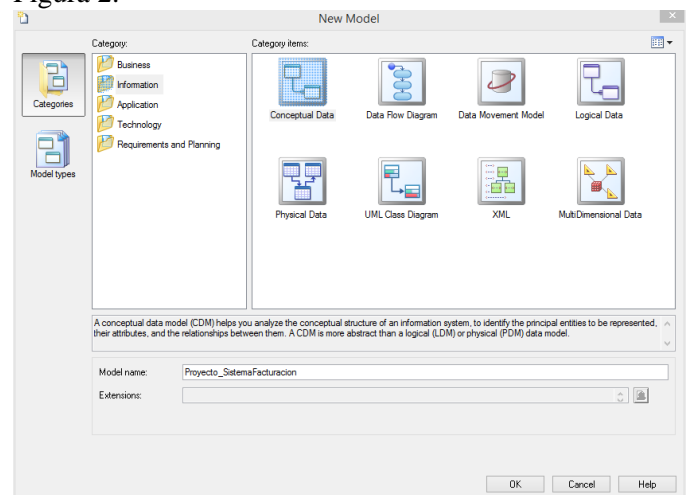


Fig. 2 Selección de Modelo

Luego de ello, aparecerá un espacio de trabajo, veremos dos cosas, a la parte izquierda un conjunto de accesorios (los cuales usaremos en toda la plantilla) y la derecha veremos en cambio todos los objetos existentes dentro de esta. Obsérvese la figura 3 y 4 respectivamente

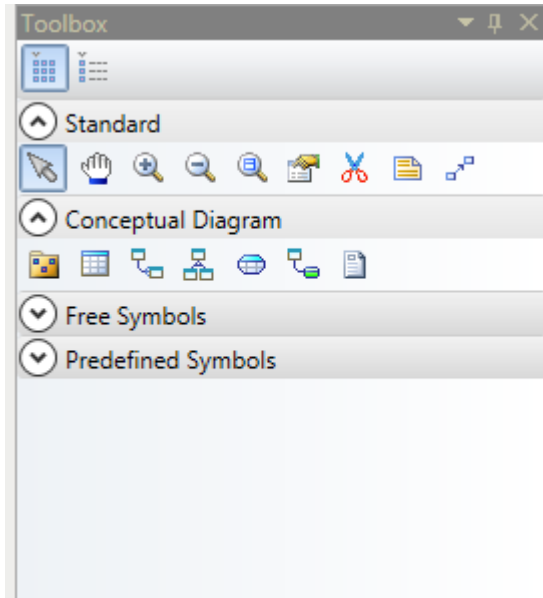


Fig. 3 Presentación de Herramientas

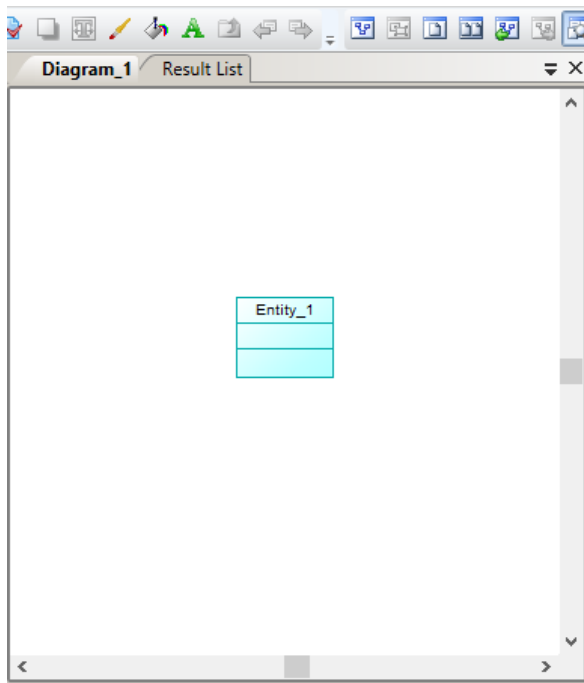


Fig. 4 Plantilla de trabajo

Si observamos el centro, vemos un cuadro “Entidad”, esa misma fue arrastrada desde el componente “entity” de la figura 3, misma que está ubicada en el panel de “Conceptual Diagrama”. Lo que necesitamos hacer es doble clic en ese recuadro entity para que se nos abra una ventana como el de la figura 5.

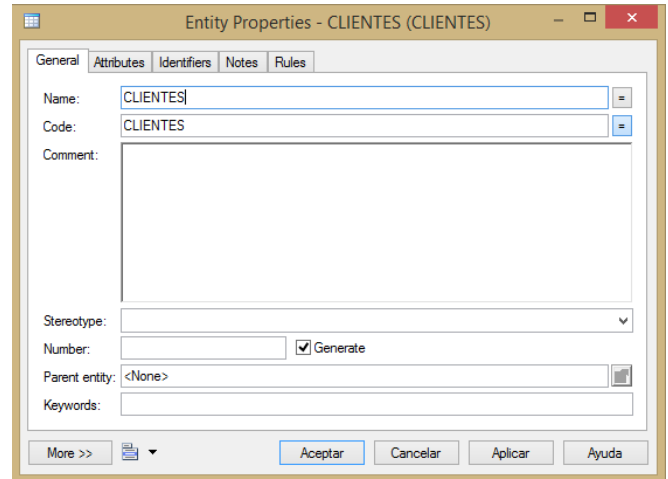


Fig. 5 Pestaña General

Vemos varias pestañas, en la general describimos el nombre, y podemos agregar un comentario sobre esta entidad, en mi caso solo le pondré el nombre de esta entidad y me desplazaré a la pestaña atributos, Figura 6

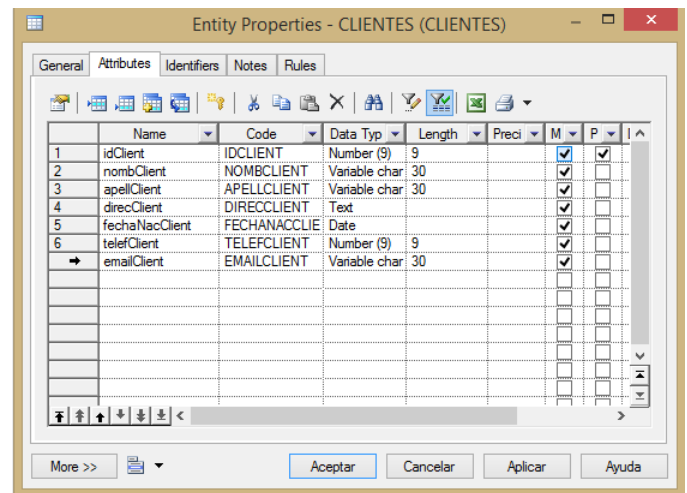


Fig. 6 Pestaña Atributos

En esta pestaña atributos definimos nombres de los atributos, tipos de datos de los mismos, algunas características adicionales como sería la longitud, la PK, y si son “Mandatory”. Y haremos lo mismo para con cualquier entidad que tengamos. Luego de esto, nos desplazaremos a la pestaña “Identifiers” Figura 7.

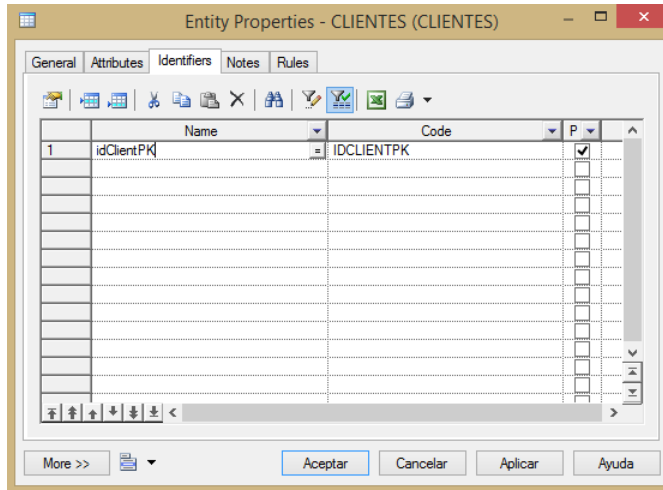


Fig.7 Pestaña Identifiers

Aquí lo que se hace es enfatizar a nuestra Pk, y listo, las siguientes pestañas son solo para agregar notas con respecto a esta entidad. Si pulsamos aceptar, y luego de haber establecido todo lo necesario en las pestañas anteriores, veremos una generación de entidad/es como la de la/s figura 8 y 9.

CLIENTES			
<u>idClient</u>	<pi>	Number (9)	<M>
nombClient		Variable characters (30)	<M>
apelClient		Variable characters (30)	<M>
direcClient		Text	<M>
fechaNacClient		Date	<M>
telefClient		Number (9)	<M>
emailClient		Variable characters (30)	<M>
<u>idClientPK</u>	<pi>		

Fig. 8 Entidad diseñada

CLIENTES			
<u>idClient</u>	<pi>	Number (9)	<M>
nombClient		Variable characters (30)	<M>
apelClient		Variable characters (30)	<M>
direcClient		Text	<M>
fechaNacClient		Date	<M>
telefClient		Number (9)	<M>
emailClient		Variable characters (30)	<M>
<u>idClientPK</u>	<pi>		

PRODUCTOS			
<u>codProd</u>	<pi>	Number (5)	<M>
nombProd		Variable characters (30)	<M>
precioProd		Float (2)	<M>
existenProd		Integer	<M>
<u>codProdOK</u>	<pi>		

CATEGORIAS_PRODUCTOS			
<u>idCategProd</u>		Number (3)	<M>
nombCateg		Variable characters (30)	<M>
descripCateg		Text	<M>
<u>idCategPK</u>	<pi>		

FACTURA			
<u>numFact</u>	<pi>	Number (5)	<M>
fechaEmisFact		Date	<M>
<u>numFactPK</u>	<pi>		

DETALLE_FACTURA			
<u>numDetalle</u>	<pi>	Number (5)	<M>
precProd		Float (2)	<M>
cantProd		Integer	<M>
precSubTotal		Float (2)	<M>
impuestProd		Float (2)	<M>
desctoProd		Float (2)	<M>
precTotProd		Float (2)	<M>
<u>numFactPK</u>	<pi>		

FORMAS_PAGO			
<u>idFormPag</u>	<pi>	Number (3)	<M>
tipoFormPag		Variable characters (30)	<M>
descripFormPag		Text	<M>
<u>idFormPagPK</u>	<pi>		

VALIDACION_USUARIO			
<u>numUsr</u>	<pi>	Number (9)	<M>
contrUsr		Integer	<M>
tipoUsr		Variable characters (30)	<M>
<u>Identificr_1</u>	<pi>		

Fig. 9 Entidades diseñadas

Ahora crearemos lo que sería la relación, de hecho este diseño de modelo se denomina Modelo Entidad-Relación porque creamos dependencias y establecemos condiciones. Si hablamos de un sistema entonces hablamos de conexiones de cualquier forma, por lo que usando una de las herramientas (Que fueron presentadas en la figura 3) denominada “Relationship”, conectamos dos entidades desde sus centros, estas entidades deben tener relación una con otra y entonces podemos darle cardinalidades véase la figura 10:

- 1 a 1
- 1 a n
- n a 1
- n a n

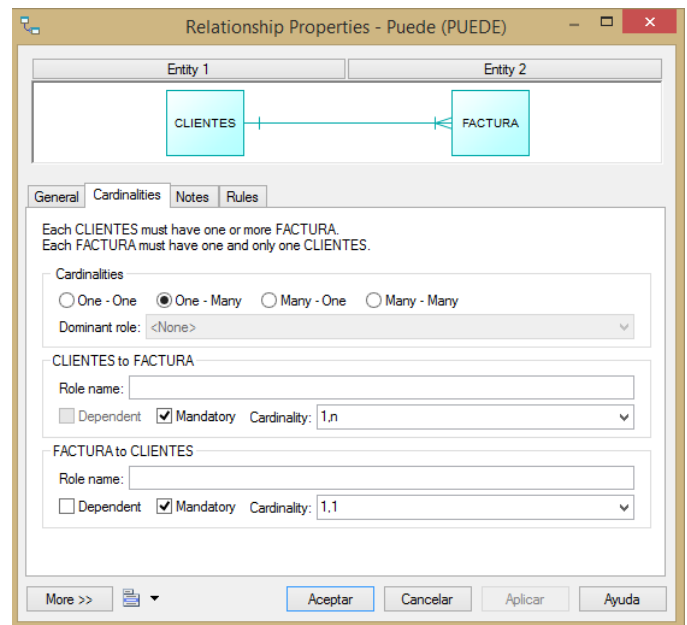


Fig. 10 Relacion de cardinalidad

Una vez hayamos hecho las relaciones habremos terminado con el primer inciso del proyecto, que era crear el modelo Entidad-Relación. Nos quedaría algo parecido a la figura 11.

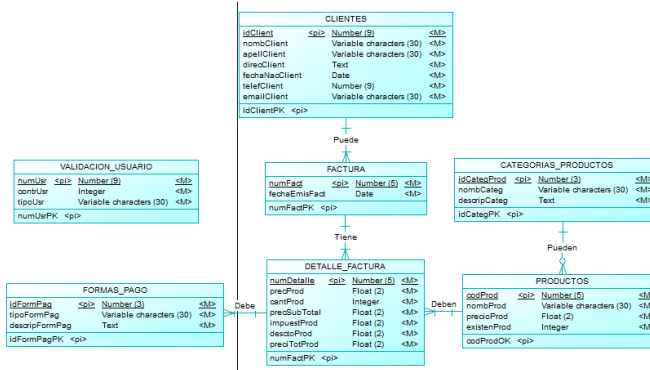
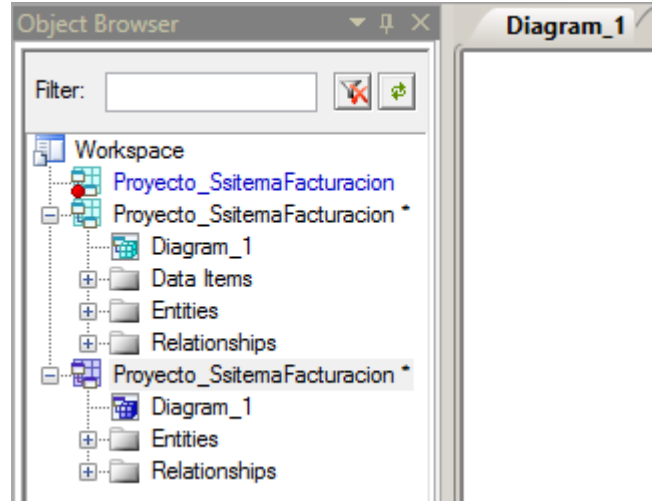


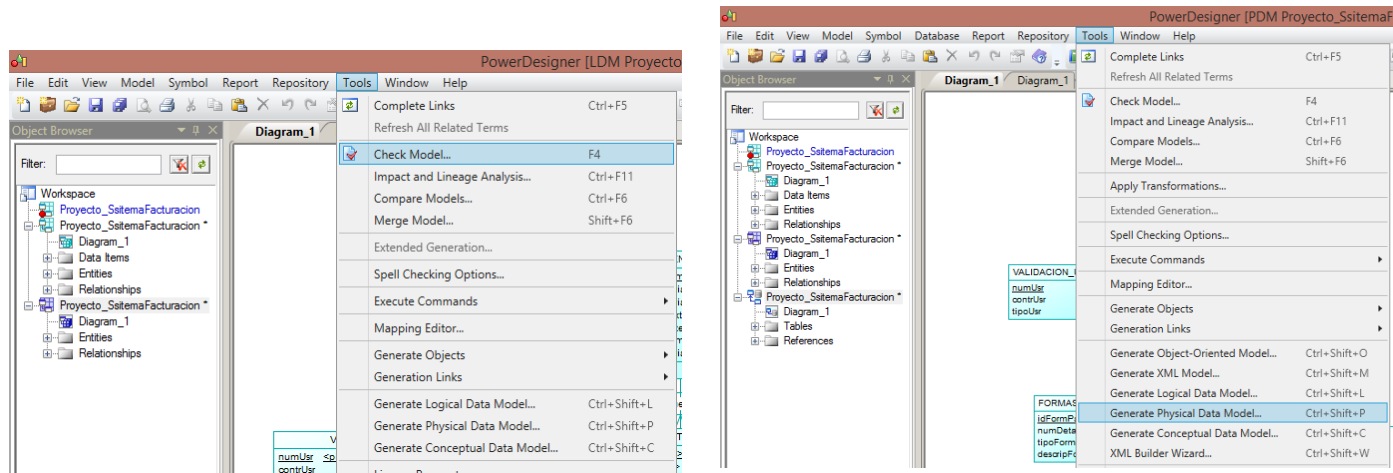
Fig. 11 Modelo Entidad-Relación



### A. Script Tabla

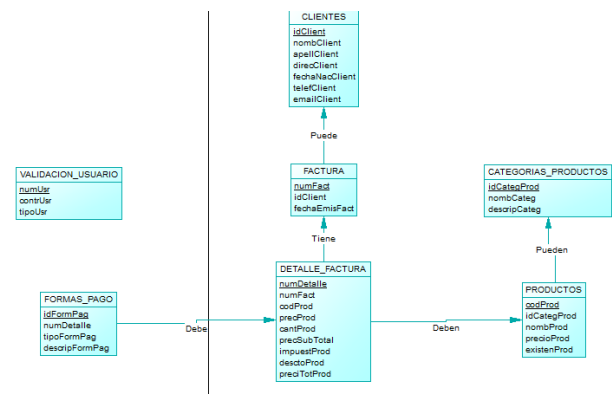
Ahora queremos generar un Script, lo que tenemos que hacer es primero hacer un “check” o un revisado de nuestro modelo entidad relación, para eso nos posicionamos en la pestaña “Herramientas” y escogemos la opción “Check Model”

Una vez revisado el modelo, debemos generar el modelo lógico, en la misma pestaña de herramientas, en la opción “Generar modelo lógico” y enseguida también “Generar modelo físico”, como puede ver en la Figura:

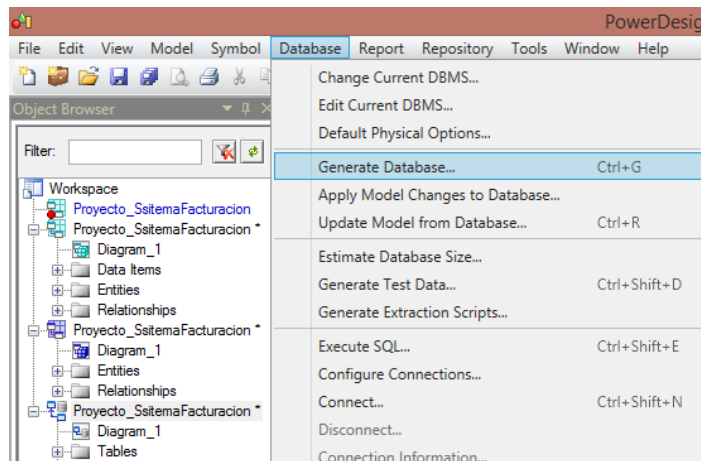


Luego de ello observaremos si nuestro modelo entidad relación no tiene errores, o advertencias, si nos los tiene, se generará en el Browser el mismo diagrama pero este estará “revisado”

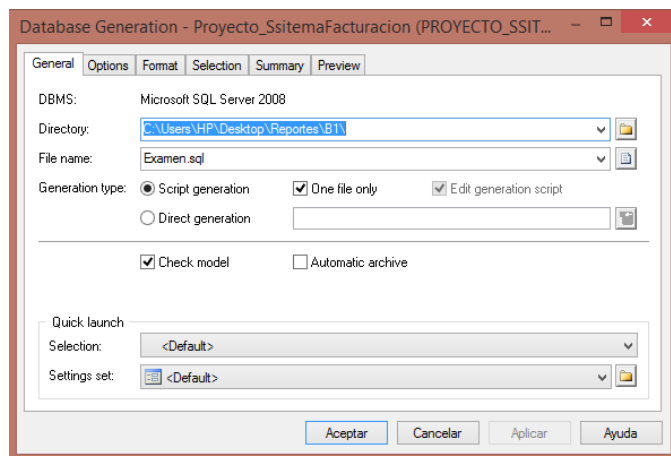
Se preseta el modelo físico generado a partir de nuestro modelo Entidad-Relación



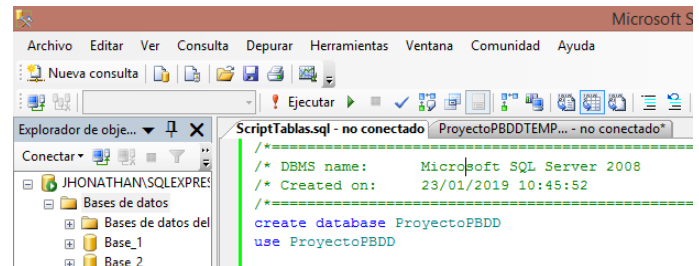
En este modelo físico, seleccionamos en la pestaña “Database” la opción “Generar código” para obtener el script de nuestro proyecto,



Por último, nos saldrá este cuadro de dialogo en dónde nos aseguraremos de verificar el destino, el nombre del Script, y nuestro DMBS:



Sin embargo existe otra manera de crear tablas, sin generarlas automáticamente, y es, usando SQL directamente, entonces si vamos a hacerlo de esta manera, lo que tenemos es que abrir el programa de MySQL Server, y dar click en “Nueva consulta”, entonces se nos genera una plantilla en blanco en donde ejecutaremos comandos para la creación de base de datos y de tablas.



Entonces con comandos “create table” hemos de crear las tablas de nuestro proyecto:

```
/*=====*/
/* Table: CATEGORIAS_PRODUCTOS */
/*=====*/
create table CATEGORIAS_PRODUCTOS (
    IDCATEGPROD numeric(3) primary key not null,
    NOMBCCATEG varchar(30) not null,
    DESCRIPCATEG text not null,
)

/*=====*/
/* Table: CLIENTES */
/*=====*/
create table CLIENTES (
    IDCLIENT numeric(9) primary key not null,
    NOMBCLIENT varchar(30) not null,
    APELLCLIENT varchar(30) not null,
    DIRECCCLIENT text not null,
    FECHANACCLIENT datetime not null,
    TELEFCLIENT numeric(9) not null,
    EMAILCLIENT varchar(30) not null,
)
```

También deberemos usar comandos para alterar o modificar tablas y hacerles una relación o bien conocida como Foreign Key en dónde la clave primaria de una instancia, es la secundaria en otra, pero tienen alguna conexión.

```
/*=====*/
/* FOREIGN KEYS: */
/*=====*/
alter table DETALLE_FACTURA
    add constraint FK_DETALLE_DEBEN_PRODUCTO foreign key (CODPROD) /*Det=
    references PRODUCTOS (CODPROD)

alter table DETALLE_FACTURA
    add constraint FK_DETALLE_TIENE_FACTURA foreign key (NUMFACT) /*Una
    references FACTURA (NUMFACT)

alter table FACTURA
    add constraint FK_FACTURA_PUEDE_CLIENTES foreign key (IDCLIENT) /*Una
    references CLIENTES (IDCLIENT)

alter table FORMAS_PAGO
    add constraint FK_FORMAS_P_DEBE_DETALLE foreign key (NUMDETALLE)/*Un
    references DETALLE_FACTURA (NUMDETALLE)

alter table PRODUCTOS
    add constraint FK_PRODUCTO_PUEDEN_CATEGORI foreign key (IDCATEGPROD) /
    references CATEGORIAS_PRODUCTOS (IDCATEGPROD)
```

### B. Script Inserción Datos

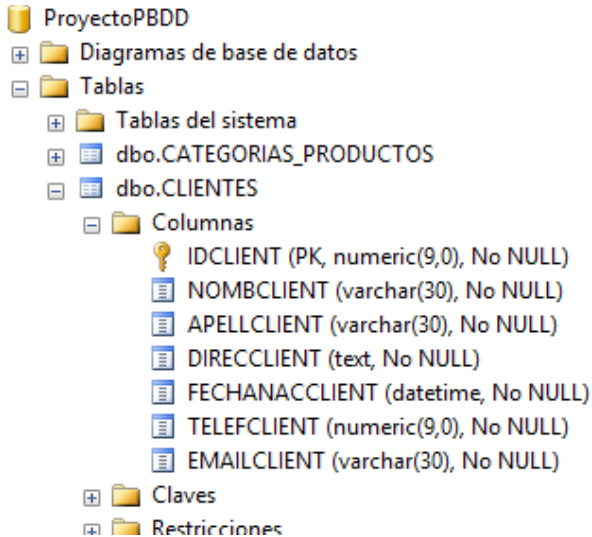
Ahora nos toca insertar datos en nuestra base de datos, con la creación de tablas necesitamos darles



**ESCUELA POLITÉCNICA NACIONAL**  
**ESCUELA DE FORMACIÓN DE TECNÓLOGOS**  
**PROGRAMACIÓN AVANZADA**

6

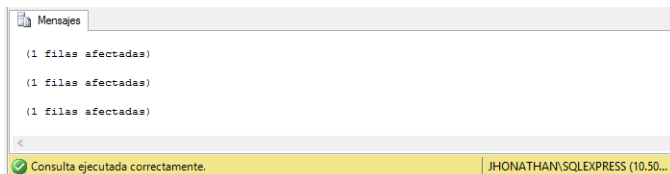
características propias a nuestros atributos, entonces haremos una inserción de datos, como podemos observar, si desplegamos nuestra base de datos, y tablas, veremos todas las tablas que tenemos creadas, también si desplegamos alguna, veremos lo que contienen estas tablas (Columnas, Claves, etc...)



Usamos el comando insert into (Tabla) para insertar los valores "Values" en el orden en el que está descrita la tabla en su columna, por ejemplo en Clientes, tenemos ID, Nombre, Apellido, etc., en ese orden, entonces mediante comandos escribiríamos al igual que la figura:

```
/*-----*/
/* INGRESO DE DATOS: CLIENTES */
/*-----*/
insert into CLIENTES values
(172535852, 'Jhonathan', 'Pizarra', 'Solanda sector 4', 19/11/1996, 05
```

Ejecutamos los comandos y tendremos:



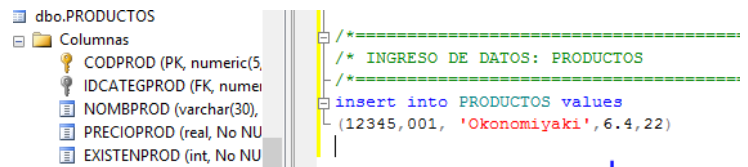
¿Cómo podemos observar esos datos ingresados?  
Usando el comando select \* from (Tabla)

```
select * from CLIENTES
```

	IDCLIENT	NOMBCLIENT	APELLCLIENT	DIRECCCLIENT	FECHANACCLIENT	TELEFCLIENT	EMAILCLIENT
1	172535820	Diego	Pilamunga	Valle de los Chillos	1900-01-01 00:00:00.000	99080321	DiegoPilamunga@
2	172535852	Jhonathan	Pizarra	Solanda sector 4	1900-01-01 00:00:00.000	99080312	jhonathanxavier@
3	172535853	Xavier	Pachacama	Quimiag sector 2	1900-01-01 00:00:00.000	99080313	XavierPachacama
4	172535854	Emilia	Lopez	Mitad del Mundo	1900-01-01 00:00:00.000	99080314	EmiliaLopez21@gr
5	172535855	Mercedes	Hinojosa	Av. Simón Bolívar	1900-01-01 00:00:00.000	99080315	MercedesHinojosa
6	172535856	Dario	Cruz	Av. 25 de Julio	1900-01-01 00:00:00.000	99080316	DarioCruz27@gmai
7	172535857	Isabel	Chirboga	Valle de los Chillos	1900-01-01 00:00:00.000	99080317	IsabelChirboga45
8	172535858	Juan	Valdez	Av. 12 de Octubre	1900-01-01 00:00:00.000	99080319	JuanValdez@gmail
9	172535859	Esteban	Ordoñez	Vicentina sector 2	1900-01-01 00:00:00.000	99080320	EstebanOrdoñez12

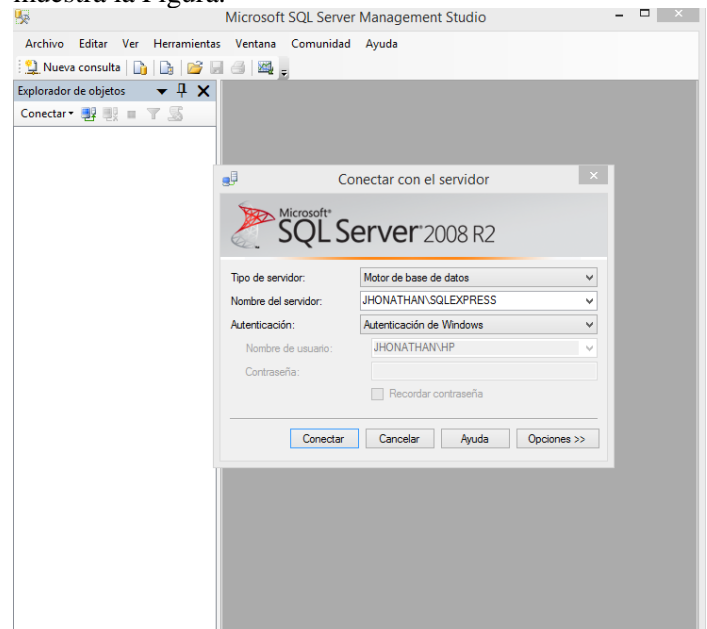
Sin embargo, la inserción que hice al principio fue de una tabla que no tenía una FK, pero ¿Qué pasaría si esta la tuviera?

La verdad es que sería imposible ingresar un registro, lo que tenemos que hacer es ingresar los datos en un orden, es decir, ni puedo por ejemplo, ingresar datos en la tabla factura sin antes haber llenado una tabla de productos.

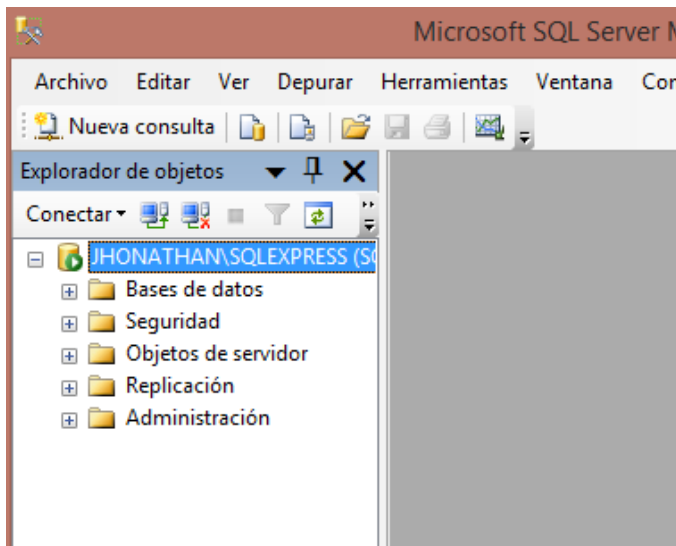


### C. Programación (Permitir al usuario ingresar)

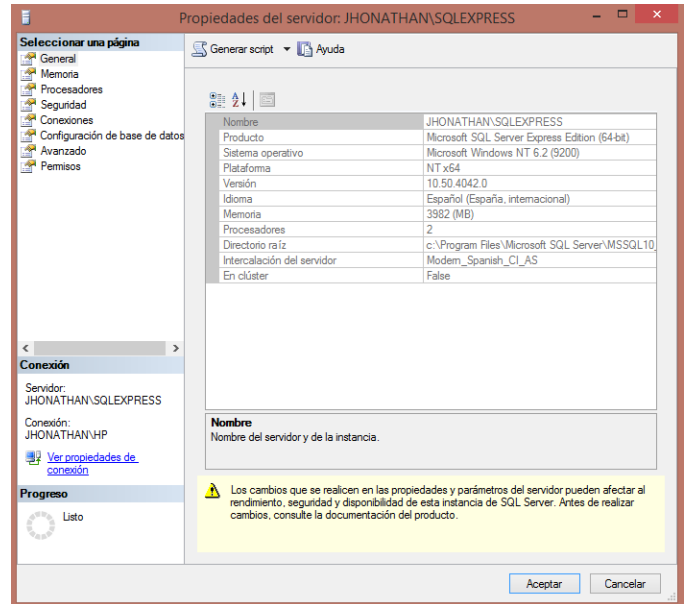
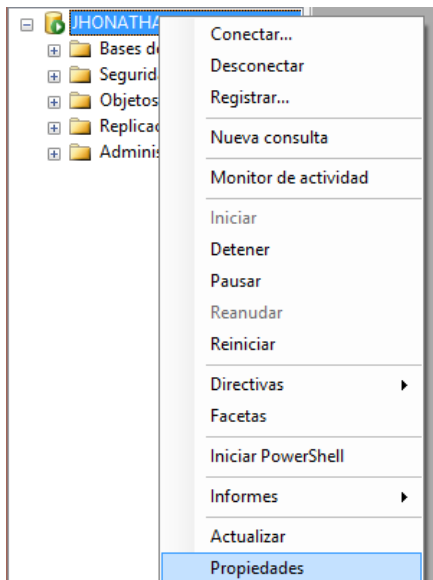
Llegamos al punto en dónde debemos hacer nuestra primera conexión con una base de datos, para ello en mi caso estoy usando NetBeans 8.2 y SQL Server 2008 R2. Abrimos ambos programas y comenzaremos por SQL, y al principio notaremos que nos sale un cuadro de dialogo en el que nos pide conectarnos (Si no es así, les recomiendo que configuren su SQL server) como el que muestra la Figura.



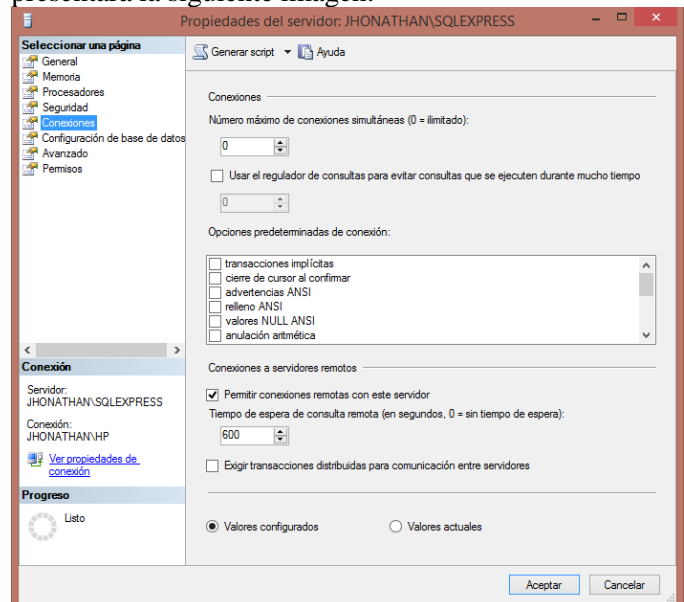
Damos clic en conectar y listo, ya estaremos conectados a nuestra base de datos:



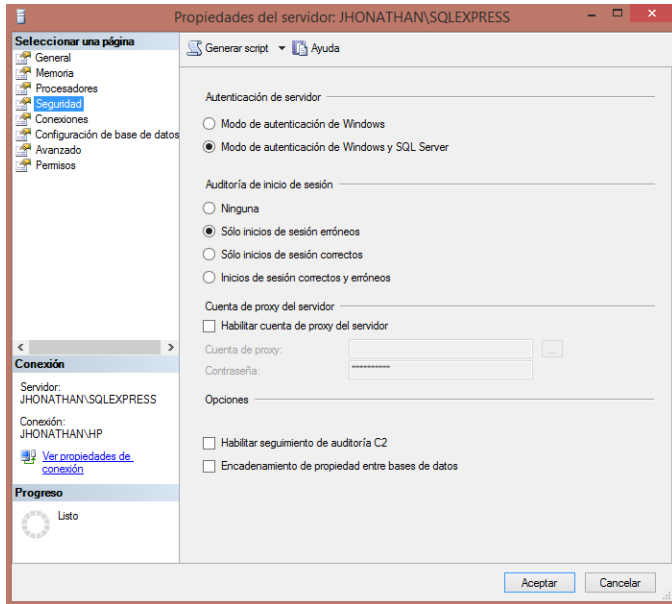
Muy bien, ahora damos click izquierdo sobre la base que engloba a todas las demás y seleccionamos propiedades, veremos que se nos abre un cuadro de dialogo con las propiedades del servidor:



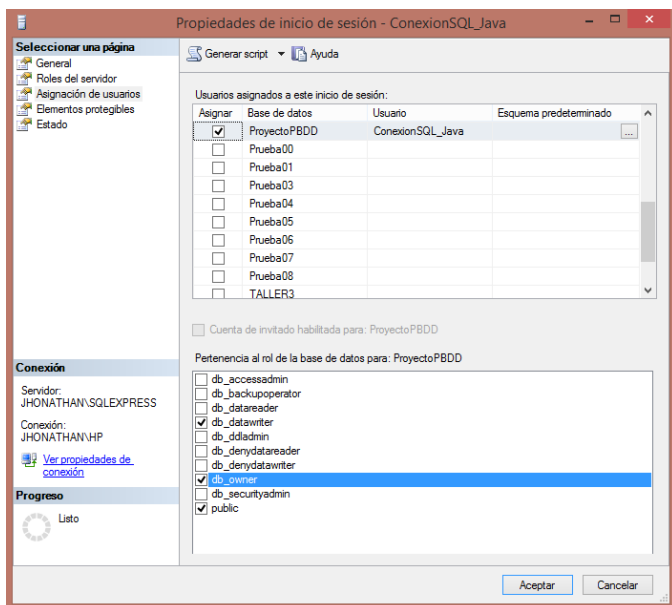
Nos desplazamos hacia “Conexiones” en dónde se presentará la siguiente imagen:



Debemos asegurarnos que esté habilitada la opción de “Permitir conexiones con este servidor”. Luego, también deberemos ir a la pestaña “Seguridad” y escoger el modo de autenticación de Windows y de SQL Server, y listo, con ello pulsaremos aceptar.

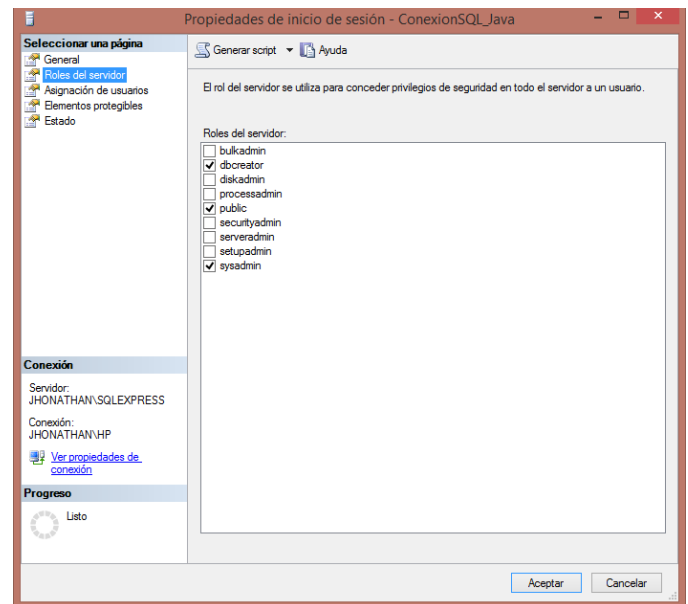


También, vamos a la pestaña de agregación de usuarios y seleccionamos nuestra base de datos que hayamos creado previamente (De ser el caso) y le habilitamos los permisos que aparecen en la figura. Cabe señalar que si usted no tiene una base de datos creada, la puede hacer desde cero, pero en la nueva base creada es decir, si usted tiene una base de datos ya creada y la quiere usar en este nuevo modelo “ConexiónSQL\_Java” deberá hacerse este paso, de lo contrario, saltárselo y empezar como si fuera a crear una nueva base de datos, pero dentro del moto de “Conexión SQL\_Java”.

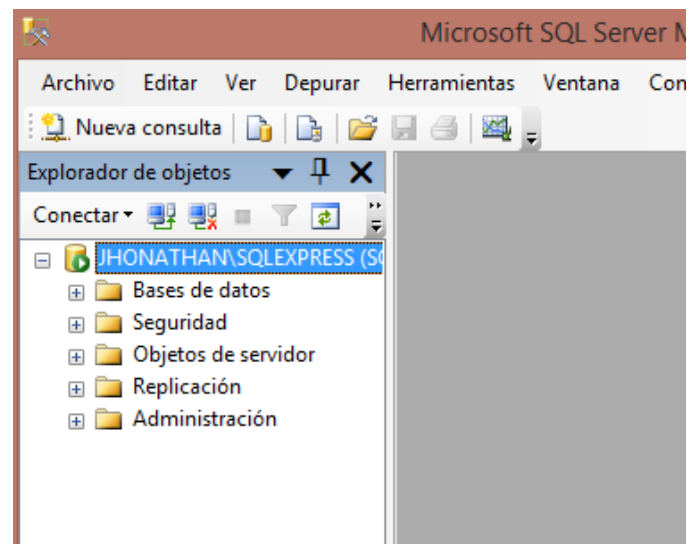


Entre otras cosas, también es oportuno darle los roles al servidor, con ello nos será o no permitido hacer ciertas cosas en nuestra base de datos, para efectos de esta

conexión solo usare los 3pero bien podría habilitarlos todos.

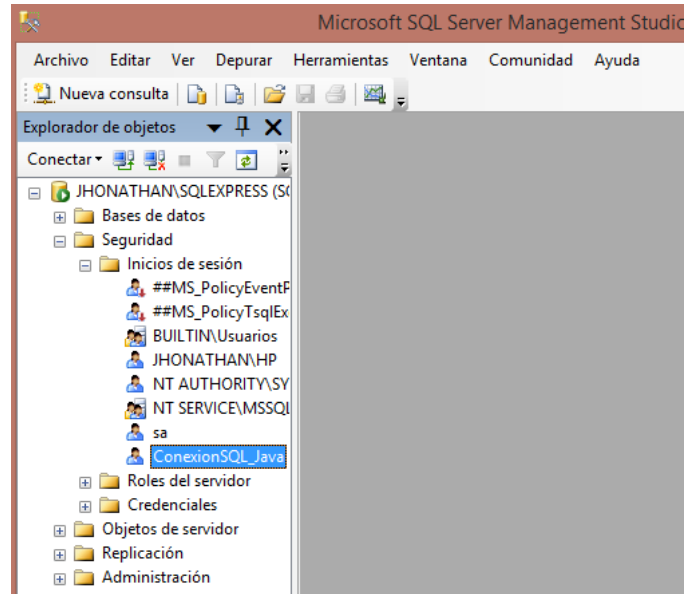
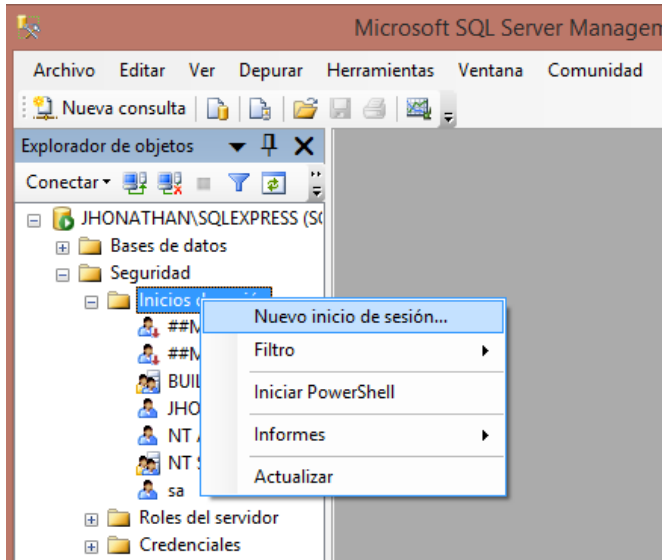


Ahora, volvemos a nuestro panel inicial, y desplegamos lo que sería Seguridad:



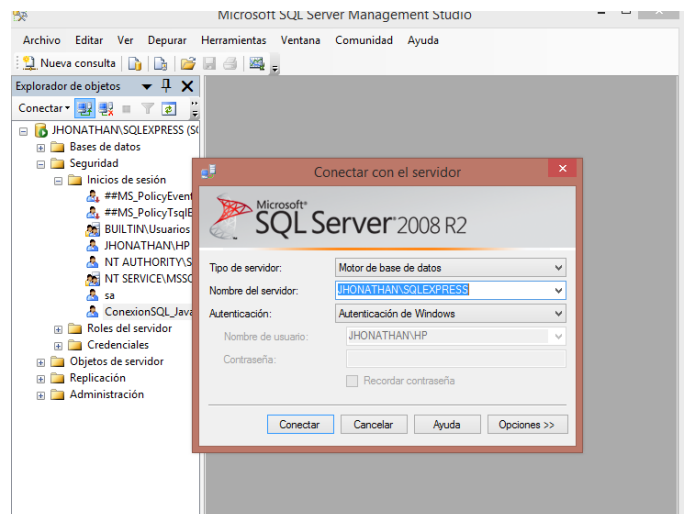
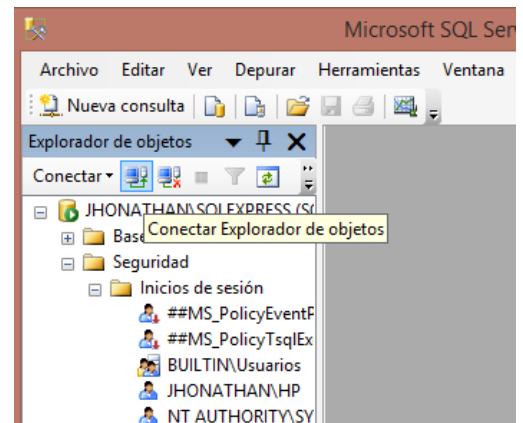
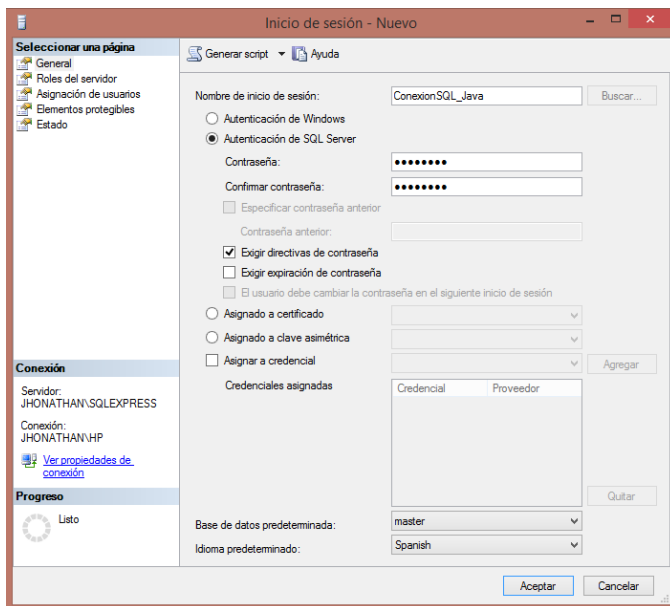
Damos clic izquierdo en “Inicios de Sesión” y escogemos la opción “Nuevo inicio de sesión”





Nos saldrá un cuadro en el que tendremos que asegurarnos de darle un nombre, y debemos seleccionar en “Autenticación de SQL Server”, para que se nos habilite la opción de asignar una contraseña; la mía es Java1996. También, le daré unos ajustes para comodidad, como des enmarcar la opción Exigir expiración de contraseña, y el idioma Español, también tenga presente que debe ser la base de datos predeterminada la Master, por último, pulsamos aceptar:

Ahora nos vamos a conectar con el explorador de objetos, damos click en el botno de conectar:

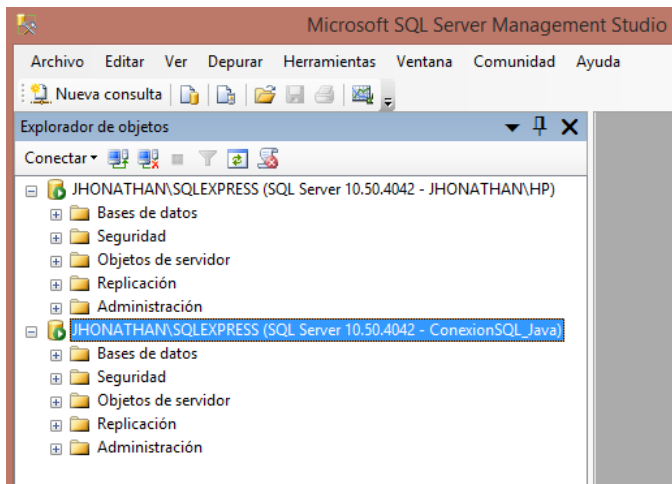
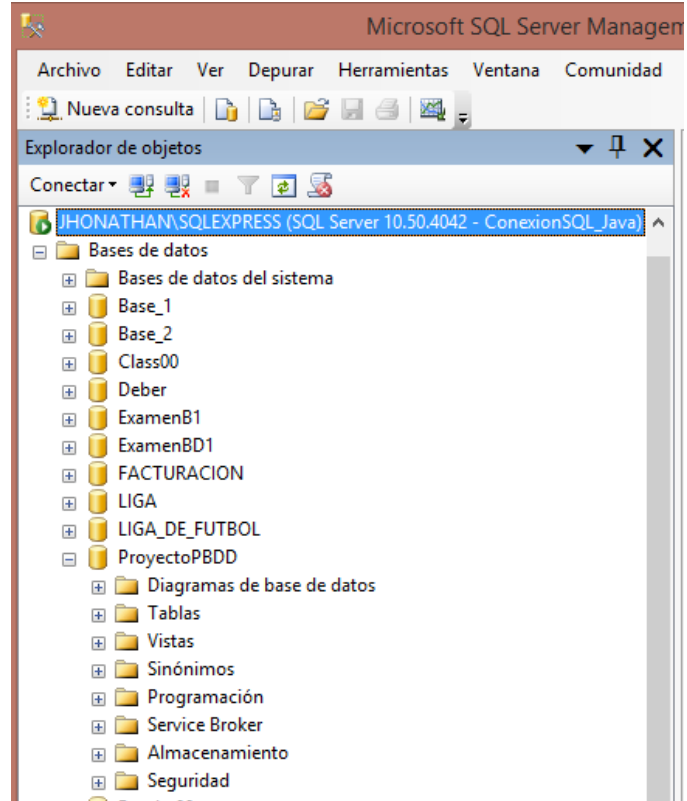


Actualizamos nuestra base de datos y nuevamente desplegamos Seguridad para comprobar si se ha creado correctamente:

Y listo, podría pensarse que hemos vuelto al comienzo pero no es así, porque ahora escogemos ya no la autenticación de Windows sino la de SQL:



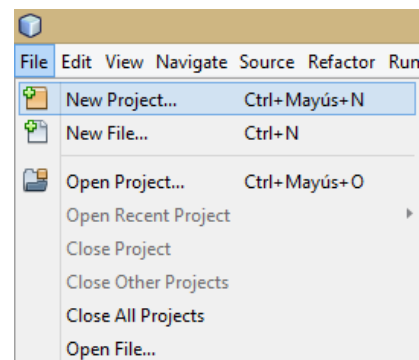
Entonces, en nuestro inicio de sesión le damos en nombre que introducimos en el cuadro de autenticación y la contraseña que también le proporcionamos y click en conectar:



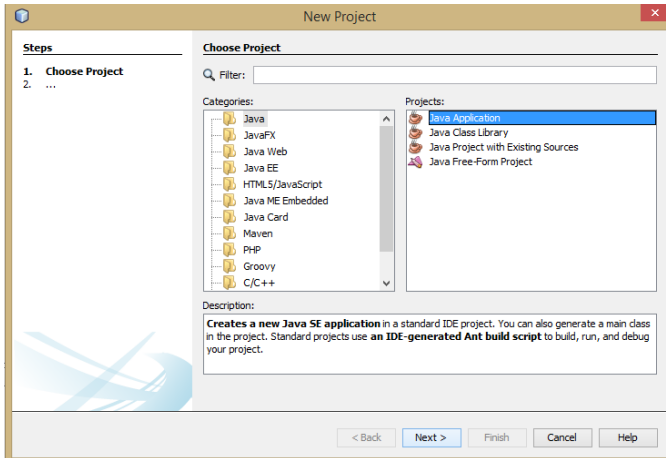
La tenemos ahí, si han seguido mis pasos, entonces se les desplegará algo como:

Esa sería la configuración del SQL

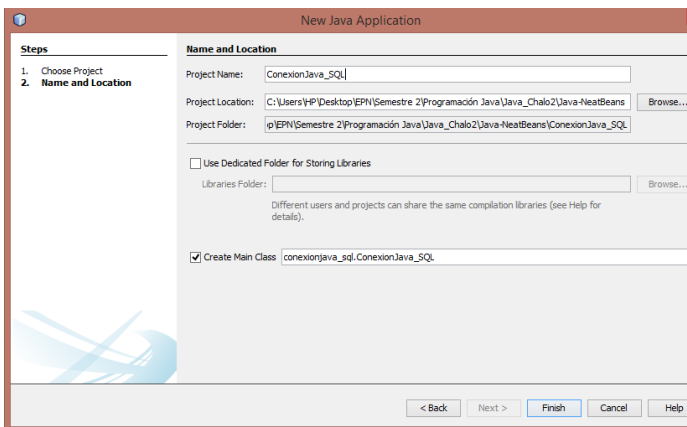
Ahora vamos con Netbean, entonces abrimos el IDE y seleccionamos en file la opción “Nuevo proyecto”



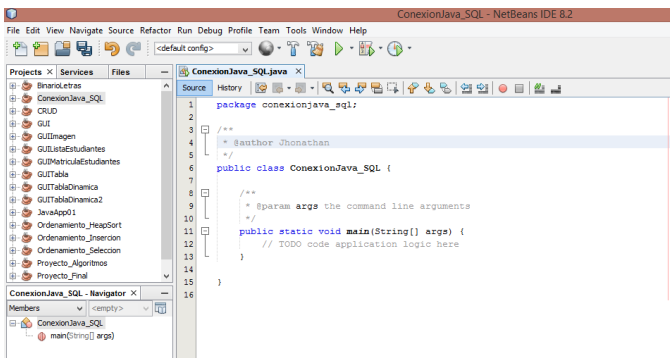
A lo que se desplegará la siguiente ventana:



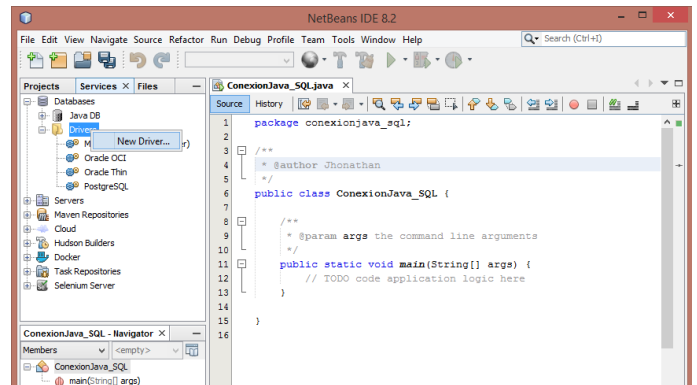
Escogemos “JavaAplication” de la categoría “Java”, y enseguida clickeamos en “Next”. Y entonces le aparecerá una ventana como la que verá a continuación:



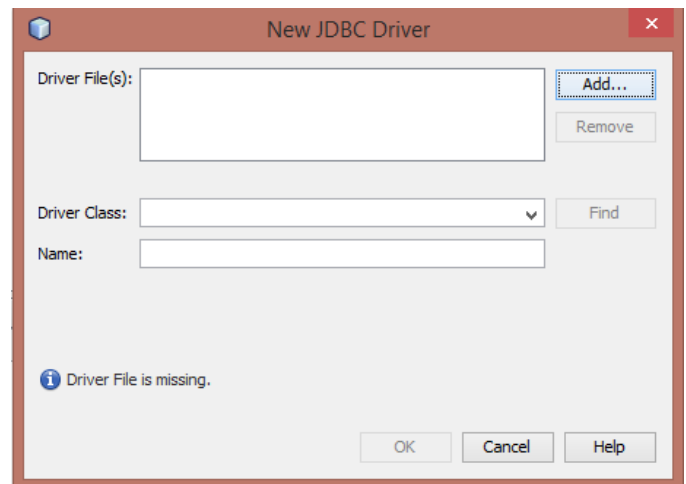
Clickeamos en Finish, y listo! Ahora se nos presentará algo así:



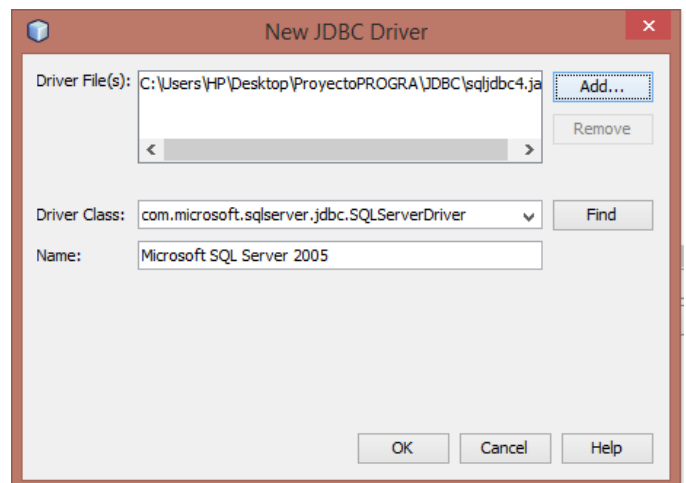
Aquí, vamos a “Services” y en “Database” seleccionamos Driver, con un clic izquierdo escogemos “Nuevo Driver”



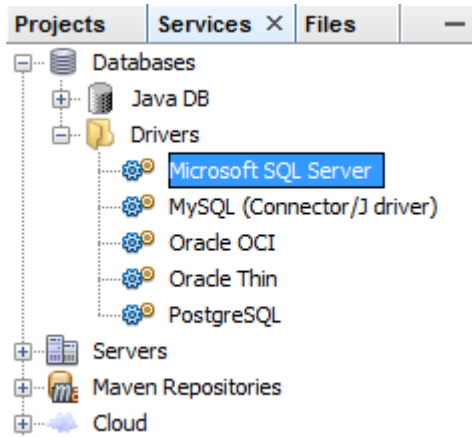
Se nos abrirá este recuadro, en dónde pulsaremos ADD y buscaremos el driver



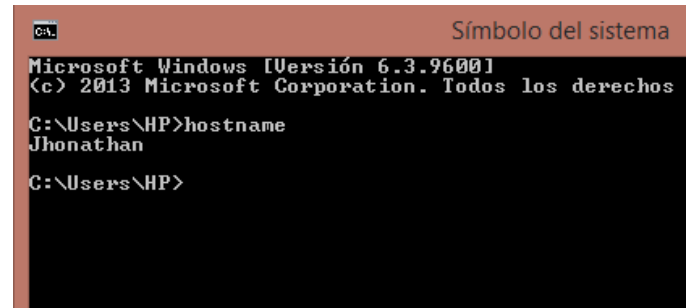
Buscamos y seleccionamos el Driver que nos hemos descargado:



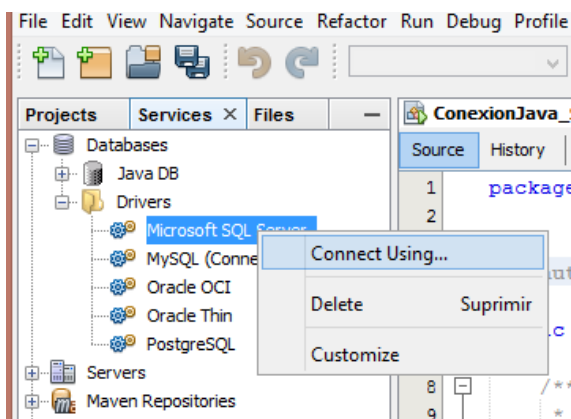
Pulsando Ok, se nos agregará el Driver



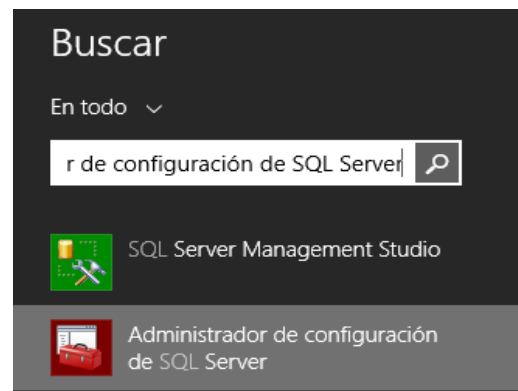
Se puede ver que mi nombre de host es Jhonathan, en su caso deberían poner el suyo o simplemente “localhos”



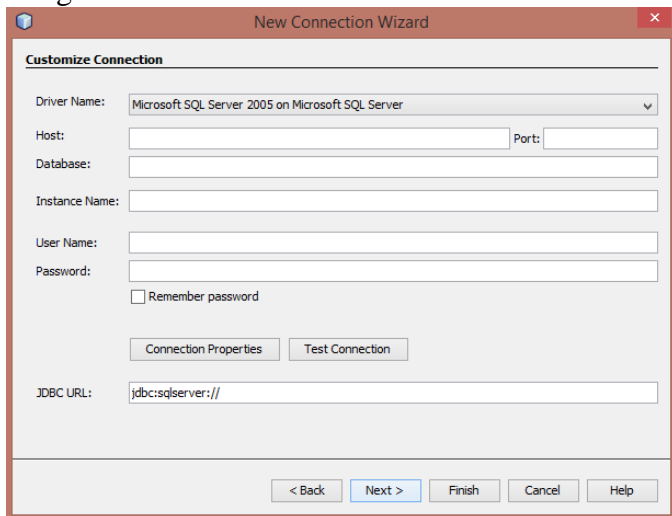
Ahora le damos clic izquierdo y seleccionamos usar conector:



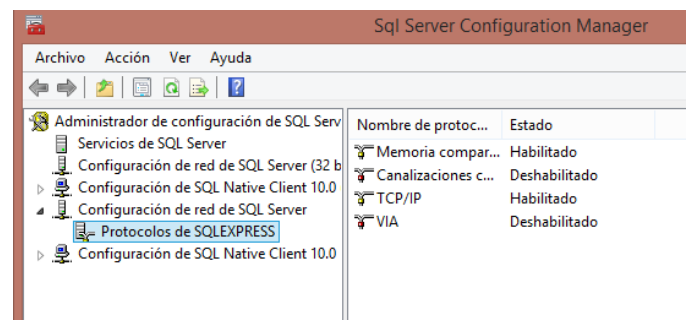
El puerto, para ver el puerto, vamos al asistente de configuración de SQL



Luego se nos abrirá esta ventana:

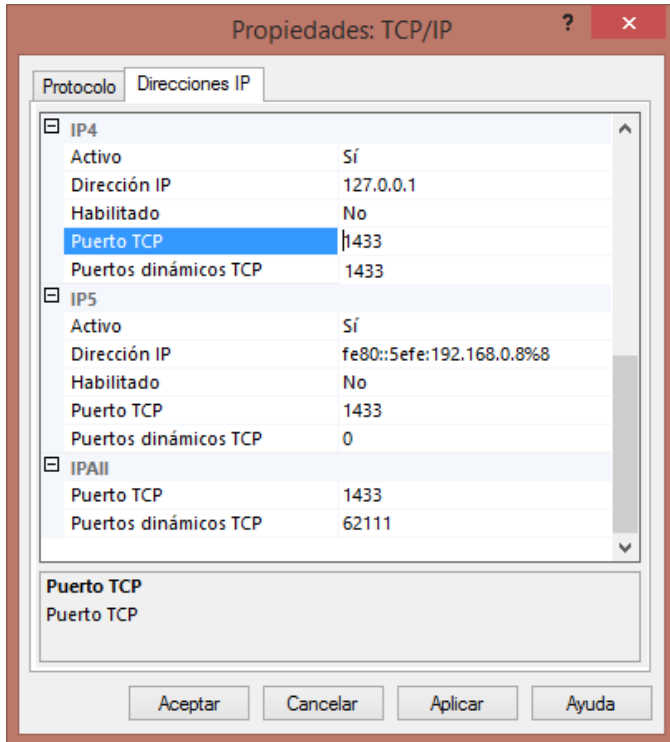


Y enseguida, vamos a dirigirnos a la pestaña



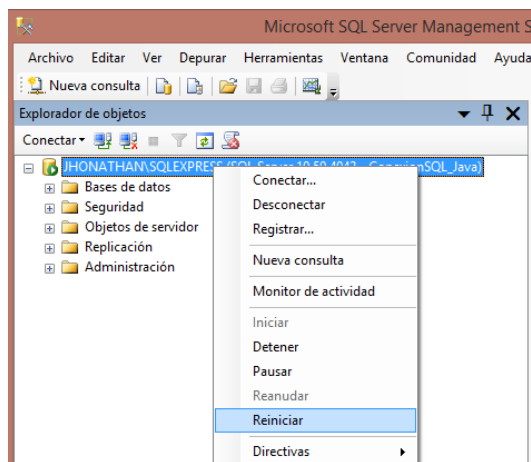
Luego en donde dice TCP/IP damos doble clic o click izquierdo y opción propiedades, y seleccionamos Direcciones IP. Recuerde que el TCP/IP debe estar habilitado, si no lo está habilítelo.

En Host debemos ingresar nuestro host, pero si no sabemos cual es el nuestro, podemos abrir desde la terminal y ejecutar el comando “hostname”

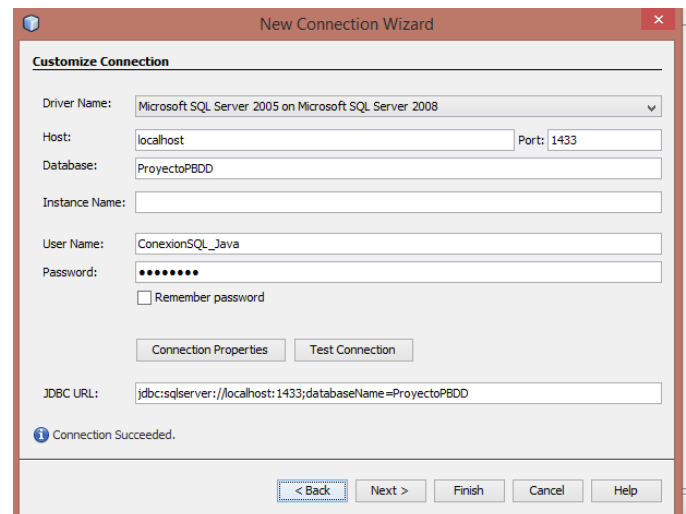
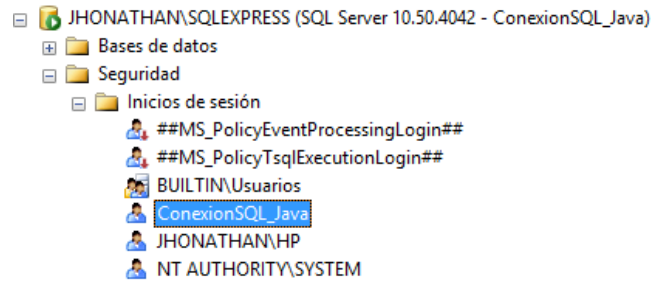


Podemos observar que en todas las IP el puerto TCP es 1433, es importante que nos aseguremos que en todos los puertos este ese numero, si no lo està, usted deberá ponerlo, por último pulsamos aceptar, y listo.

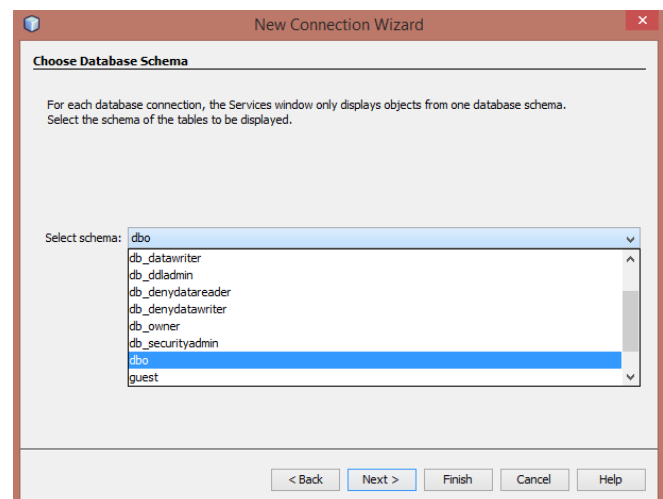
Ahora, nos debemos dirigir a nuestro SQL Server y reiniciar el servicio.



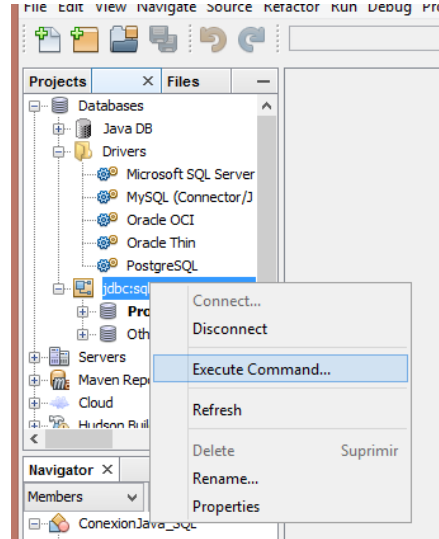
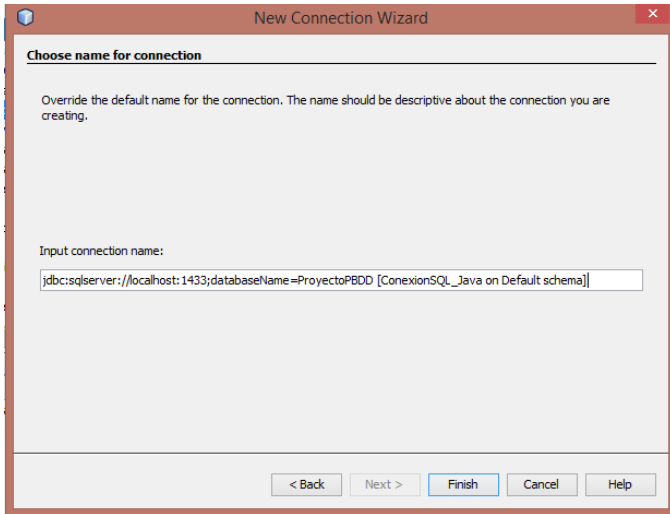
Por último, volvemos a Java-Netbeans, e intentamos llenar todos los campos, como dije puede ser localhost o el hostname que tengan, el puerto (Que hemos habilitado) la base de datos en la que trabajaremos, y en User name sería el usuario que somos en SQL:



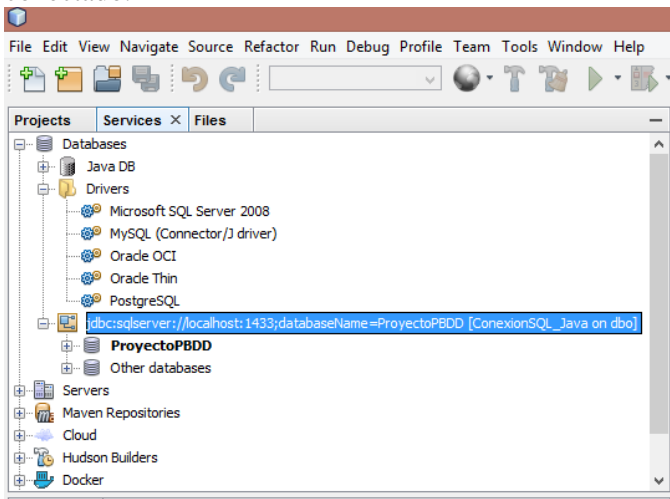
La contraseña, la misma que pusimos para ese usuario, la mía era Java1996, por último, damos click en “Test Connection” y listo, la conexión ha sido exitosa! Rapidamente, damos click en next y se nos abrirá la siguiente ventana:



Aquí escogemos dbo que es el esquema en el que trabajaremos, el de manejo de las base de datos, damos click en next

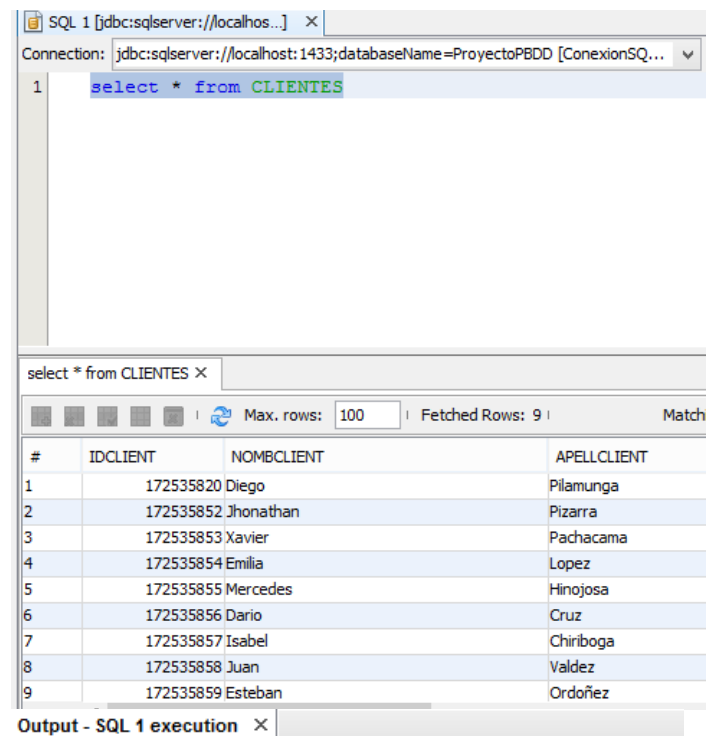


Aquí aparece lo que es el input o el nombre de la entrada de conexión y puslamos finish  
Y aquí ya tenemos la conexión de SQL con Java, lo hemos conectado:



Vamos a hacer nuestra primera consulta!  
Clic izquierdo – ejecutar comando

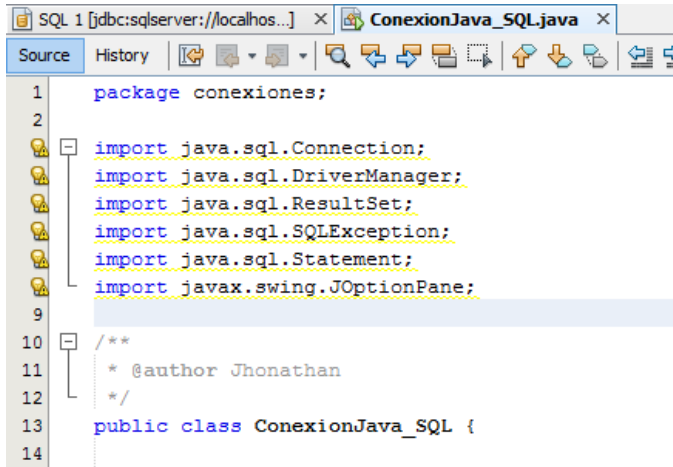
Y escribimos algún comando de los que solemos en SQL, y también el botón ejecutar, entonces nos debería mostrar por consola algo como:



```
Executed successfully in 0,521 s.  
Fetching resultset took 0,437 s.  
Line 1, column 1  
  
Execution finished after 11,656 s, no errors occurred.
```

Ahora trabajaremos en Java

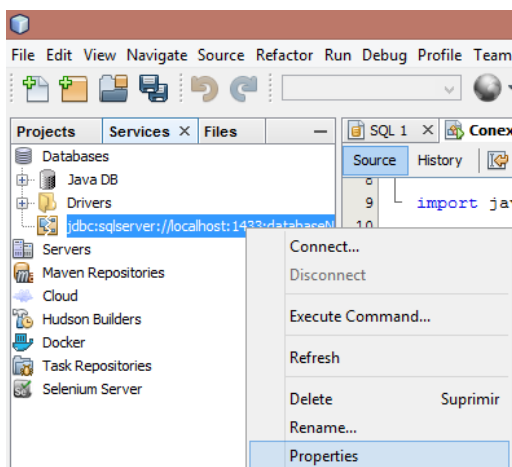




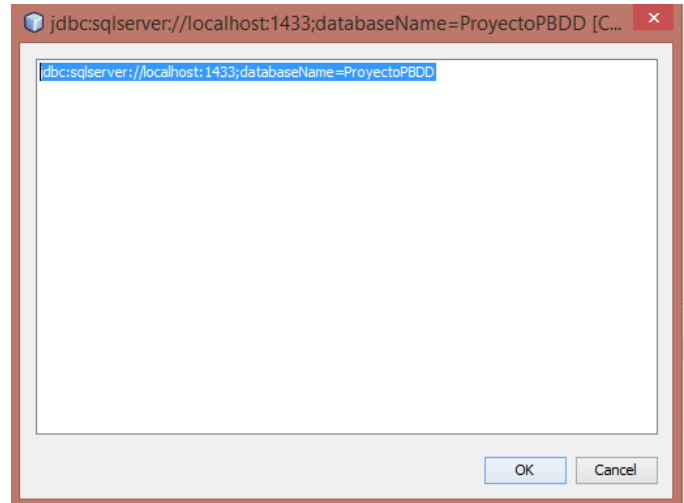
```
1 package conexiones;
2
3 import java.sql.Connection;
4 import java.sql.DriverManager;
5 import java.sql.ResultSet;
6 import java.sql.SQLException;
7 import java.sql.Statement;
8 import javax.swing.JOptionPane;
9
10 /**
11  * @author Jhonathan
12  */
13 public class ConexionJava_SQL {
14
```

Importando (O heredando) de la clase Connection y DriverManager, además de otras clases que nos permitan trabajar conjuntamente con NetBeans y bases de datos

Después en la parte de Services, dando un clic izquierdo escogeremos la opción Propiedades:



Se nos muestra entonces un enlace, lo que necesitamos es copiar ese enlace para poder conectarnos a SQL



Después de eso creamos a manera de String una variable que almacenará esa dirección:



```
package conexiones;

import java.sql.Connection;
import java.sql.DriverManager;
import java.sql.ResultSet;
import java.sql.SQLException;
import java.sql.Statement;

import javax.swing.JOptionPane;

/**
 * @author Jhonathan
 */
public class ConexionJava_SQL {

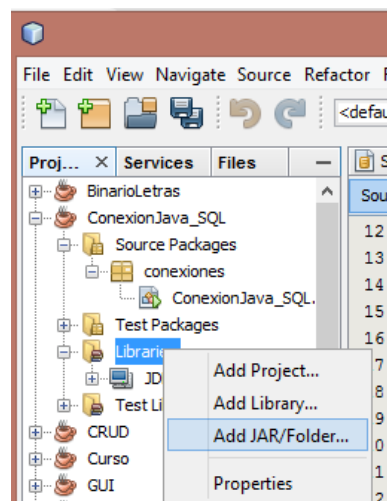
    static Connection contacto = null;

    public static Connection getConexion() {

        String url = "jdbc:sqlserver://localhost:1433;databaseName=ProyectoPBDD";
```

Ahora tenemos que ir en busca de nuestro Drive que nos permitirá

Clickeando izquierdo y seleccionando la opción JAR/Folder empezaremos la búsqueda



```

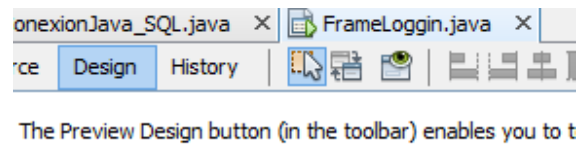
public static void setcuenta(String usuario, String password){
    ConexionJava_SQL.usuario = usuario;
    ConexionJava_SQL.password = password;
}

public static boolean getstatus(){
    return status;
}

//Este metodo nos servirá para hacer una consulta
public static ResultSet Consulta(String consulta){
    Connection con = getConexion();
    Statement declara;
    try{
        declara=con.createStatement();
        ResultSet respuesta = declara.executeQuery(consulta);
        return respuesta;
    }catch (SQLException e){
        JOptionPane.showMessageDialog(null, "Error" + e.getMessage(),
            "Error de Conexion",JOptionPane.ERROR_MESSAGE);
    }
    return null;
}

```

Crearemos nuestro primer Frame, este nos ayudará a saber si nos hemos conectado a la base de datos, y a partir de ahí es dónde vamos a seguir trabajando el resto del proyecto, de JFrames y bases de datos.



Usuario:

Contraseña:

Conectar

Con un poco de diseño nuestro diseño podrá verse estético:

```

public static Conexion getConnection() {
    status = false;
    String url = "jdbc:sqlserver://localhost:1433;databaseName=ProyectoPDDO";

    //Si la conexion del driver no es la correcta
    try {
        Class.forName("com.microsoft.sqlserver.jdbc.SQLServerDriver");

    } catch (ClassNotFoundException e) {
        JOptionPane.showMessageDialog(null, "No se pudo establecer la conexion... revisar Driver" + e.getMessage(),
            "Error de Conexion", JOptionPane.ERROR_MESSAGE);
    }

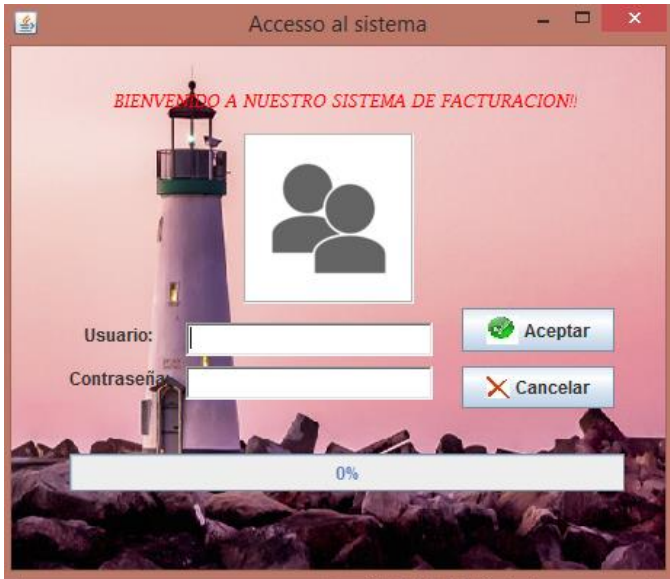
    //Si
    try {
        contacto = DriverManager.getConnection(url, ConexionJava_SQL.usuario, ConexionJava_SQL.password);
        status = true;

    } catch (SQLException e) {
        JOptionPane.showMessageDialog(null, "Error" + e.getMessage(),
            "Error de Conexion", JOptionPane.ERROR_MESSAGE);
    }

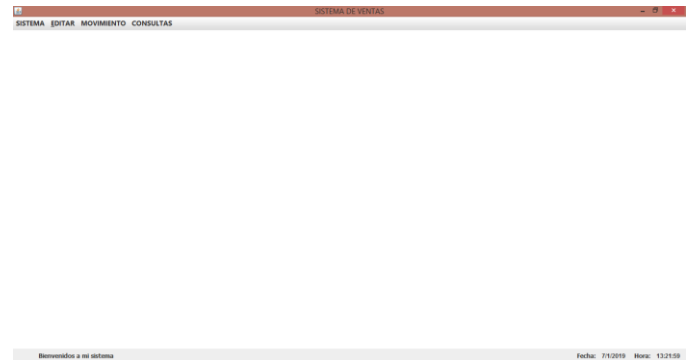
    return contacto;
}

```

Acompañado de estos métodos que nos ayudarán a saber si nos hemos conectado (Un estado de conexión)

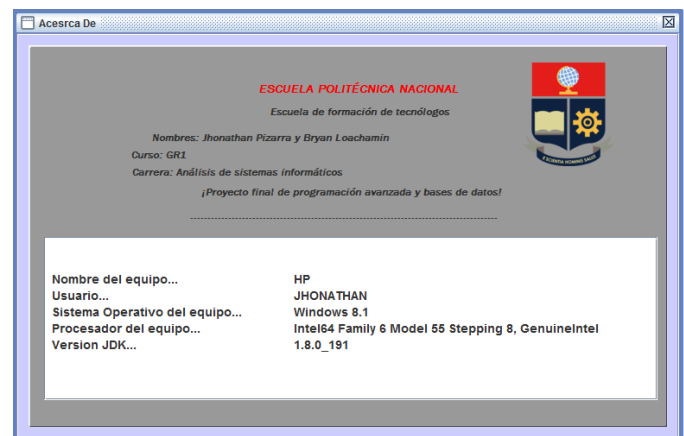


Una vez que el usuario ha entrado correctamente, se nos desplegará una ventana como esta:



Si clickeamos entre las diferentes pestañas entonces podremos encontrar varias funcionalidades:

La primera presenta lo que es información al usuario de lo que es el proyecto:



Listo, entonces iniciamos creando este Frame o Frame Login y la idea es que ingresemos como usuario: Cliente o Administrador, inicialmente lo creamos así, y después podremos crear un ComboBox o algo para controlar eso, lo importante es ahora crear un Login, en dónde:

```
private void btnAceptarActionPerformed(java.awt.event.ActionEvent) {  
    // TODO add your handling code here:  
    IniciarSesion();  
}
```

Inicialmente llama una función que comprobará si los datos fueron correctamente registrados:

```
public void IniciarSesion() {  
    final String u = "Java", p = "123";  
    String user = txtUsuario.getText();  
    String pass = this.txtContraseña.getText();  
    objetotimer = new Timer(80, new claseTimer());  
    try {  
        //chekar si el usuario escribio el nombre de usuario y pw  
        if (txtUsuario.getText().length() > 0 && txtContraseña.getText().length() > 0) {  
            // Si el usuario si fue validado correctamente  
            if (user.equals(u) && pass.equals(p)) //enviar datos a validar  
            {  
                objetotimer.start();  
            }  
        } else {  
            JOptionPane.showMessageDialog(null, "El nombre de usuario y/o contraseña no es correcto");  
            txtUsuario.setText(""); //limpiar campos  
            txtContraseña.setText("");  
            txtUsuario.requestFocusInWindow();  
        }  
    }  
}
```

Entre otras funciones que, simplemente nos servirían para darle estética al programa.

Tenemos por ejemplo una función fondo para proyectar la imagen del faro que se ve ahí atrás, tenemos una función de tipo timer que nos ayudará a darle el efecto de carga, etc.

Simplemente hemos usado Text Area y Labels para su creación. En la siguiente pestaña del menú Editar tenemos lo que son Clientes y Productos, nosotros en este punto somos una especie de súper usuario que tienen acceso a todo esto, pero normalmente un cliente no lo haría

CODIGO	NOMBRES	APELLIDOS	SEXO	CI	TELEFONO	RUC	E_MAIL	DIRECCION
00001	Daniel	X	M	17253585				
00002	Jhonathan	Pizarra	M	17253578	1233	2344		
00003	Emilia	Lopez	F	12245667				

En esta ventana nos está permitido registrar un nuevo cliente, modificarlo o eliminarlo, todo esto se verá afectado en la base de datos, y a su vez, se presentará de forma gráfica en la tabla, así. ¿Cómo logramos esto? Bueno, en este caso hemos estado trabajando en el menuClientes: Aquí está instanciando un objeto de tipo Clientes, el cual sujeto a un Panel se presentará en determinadas posiciones.

```
private void menuClientesActionPerformed(java.awt.event.ActionEvent
// TODO add your handling code here:
try{
    Clientes cl =new Clientes();
    jDesktopPanel1.add(cl);
    cl.show();
    cl.setLocation(50, 5);
} catch (Exception e) {
}
}
```

La Instancia Clientes tendrá la implementación de muchas cosas, pero de las más destacables son:

```
DefaultTableModel tabla = new DefaultTableModel();
try {
    tabla.addColumn("CODIGO");
    tabla.addColumn("NOMBRES");
    tabla.addColumn("APELLIDOS");
    tabla.addColumn("SEXO");
    tabla.addColumn("CI");
    tabla.addColumn("TELEFONO");
    tabla.addColumn("RUC");
    tabla.addColumn("E_MAIL");
    tabla.addColumn("DIRECCION");

    cts = cn.prepareCall("{call mostrarclientes}");
    r = cts.executeQuery();
    while (r.next()) {
        Object dato[] = new Object[9];
        for (int i = 0; i < 9; i++) {
            dato[i] = r.getString(i + 1);
        }
        tabla.addRow(dato);
    }
    this.tblClientes.setModel(tabla);
}
```

Dentro de una función “Cargar” estamos llamando a un proceso que hemos creado en SQL Server, el proceso mostrar clientes simplemente retorna los valores de los campos que se muestras en la tabla: Código, Nombres...etc.

De hecho, cada botón tiene una funcionalidad específica para con la base de datos, por ejemplo el botón modificar:

```
String cod = txtCodigo.getText();
String nom = txtNombre.getText();
String ape = txtApellido.getText();
String sexo = this.cmbSexo.getSelectedItem().toString();
String dni = txtDNI.getText();
String tel = txtTelefono.getText(), ruc = txtRuc.getText(), Emil = txt
String dir = txtDireccion.getText();

//Agrego a la base de datos:
try {
    cts=cn.prepareCall("{call agregarcliente(?,?,?,?,?,?,?,?)");
    cts.setString(1, cod);
    cts.setString(2, nom);
    cts.setString(3, ape);
    cts.setString(4, sexo);
    cts.setString(5, dni);
    cts.setString(6, tel);
    cts.setString(7, ruc);
    cts.setString(8, Emil);
    cts.setString(9, dir);
}
```

Más adelante se verá como se compone un proceso de SQL Server, lo que nos interesa ahora es escrutar en la Programación de esos procesos para la interacción con el usuario. Así mismo se compondría.

CODIGO	DESCRIPCION	CANTIDAD	PRECIO UNITARIO	PRECIO V
P1	Onigiris bolas de arroz recubiertas	2	30	60.0
P1	Onigiris bolas de arroz recubiertas	1	30	30.0
P0	Tempura no hay	5	20	100.0
Subtotal: 190.0    Descuento 10%: 0.0    IVA: 0.0    Total a Pagar: 190.0				

Algo característico, por ejemplo de la lista de productos serían las validaciones, por ejemplo que en el nombre del producto no me pueda escribir un número y cosas así:

```
private void txtNombProductoKeyTyped(java.awt.event.KeyEvent evt) {
    // TODO add your handling code here:
    char car = evt.getKeyChar();
    if ((car < 'a' || car > 'z') && (car < 'A' || car > 'Z') && (car < ' ' || ca:
        evt.consume();
    }
}
```

Esta restricción está aplicada para los textFields, sin embargo cabe señalar que si podría implementarlos por medio de un JOptionPane.

Para un poco resumir: cada vez que estemos interactuando en un Frame con una instancia, también estaremos afectando a la base de datos, esa es la idea detrás de todo el programa, y así se componen los sistemas de facturación.

Siguiendo hacia el Frame de Facturación

Aquí podremos hacer que un cliente haga un pedido y se emita una primera factura:

Cargaríamos los datos, y buscaríamos entre los productos, ordenando los que necesitemos (como clientes) y obteniendo un cálculo por todos ellos, considerando IVA, descuentos, etc.

Sin embargo también es muy recomendable controlar que el usuario no ingrese un dato inadecuado como una letra cuando se le solicita una cantidad.

Por último desplegaremos un Frame que nos muestre las ventas por Fecha:

Detrás siempre habrá un montón de código, mi intención es mostrar los más importantes, principalmente son importantes porque no son de estética sino que trabajan con la base de datos, por ejemplo:

```
private void btnCalcularActionPerformed(java.awt.event.ActionEvent evt) {
    // TODO add your handling code here:
    double suma = 0.0;
    double subtot = 0.0;

    for (int i = 0; i < jta_detalleboleta.getRowCount(); i++) {

        String precio = jta_detalleboleta.getValueAt(i, 3).toString();
        String cantidad = jta_detalleboleta.getValueAt(i, 2).toString();
        int c = Integer.parseInt(cantidad);
        double p = Double.parseDouble(precio);
        suma = c * p;
        subtot += suma;

        jta_detalleboleta.setValueAt(suma, i, 4);
        txtTotal.setText("" + subtot);
    }
}
```

Realiza cálculos y una vez terminado, los envía a la base de datos.

#### D. Script Creación de tablas SQL

Una tabla contendrá filas y columnas, en bases de datos esas se conoce como archivos y a sus componentes como atributos,

Vemos que inicialmente creamos tablas, las cuales contendrán atributos, “código de cliente”, “CI”, etc, acompañadas de un tipo de dato, y un limitante, y condiciones de permitir o no valores nulos

```
create database proyectoBDDP
use proyectoBDDP

/*=====
/* Table: CLIENTES
/*=====
create table CLIENTES
(
    codcli varchar(5) primary key NOT NULL,
    Nombreciente varchar(30) NULL,
    Apellidoscliente varchar(30) NULL,
    sexo char(1) NOT NULL,
    CI varchar(8) NULL,
    telefono varchar(9) NULL,
    ruc varchar(11) NULL,
    e_mail varchar(50) NULL,
    direccion varchar(50) NULL,
)
```

Después (Y una de las partes más importantes) es que podemos incluirle datos:

```
/*=====
/* INGRESO DE DATOS: CLIENTES
/*=====
insert into CLIENTES values
(172535852, 'Jhonathan', 'Pizarra', 'Solanda sector 4', 19/11/1996, 099

insert into CLIENTES values
(172535853, 'Xavier', 'Pachacama', 'Quimiag sector 2', 18/10/1997, 0990

insert into CLIENTES values
(172535854, 'Emilia', 'Lopez', 'Mitad del Mundo', 02/04/1995, 099080314

insert into CLIENTES values
(172535855, 'Mercedes', 'Hinojosa', 'Av. Simón Bolívar', 13/07/1999, 09

insert into CLIENTES values
(172535856, 'Dario', 'Cruz', 'Av. 25 de Julio', 01/01/2001, 099080316,

insert into CLIENTES values
(172535857, 'Isabel', 'Chiriboga', 'Valle de los Chillos', 14/07/1995,
```

#### E. Script Procesos

Hace poco se habló de los procesos en SQL, un proceso es un conjunto de instrucciones que desencadenan otras, es así que podemos tener fluidez de datos entre las bases de datos, todo proceso consta de una serie de pasos, y cada proceso hará algún acción diferente, es por eso que son tan importantes, trabajando en un sistema un proceso nos ayudaría a optimizar recursos y tiempo ejecutando miles de instrucciones con una sola.

```
/*=====
/* Proceso: Buscar Boletos
/*=====
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
create procedure buscarboletos
AS BEGIN
SELECT * FROM BOLETOS
END
```

#### F. Script Foreign Keys

Otra de las partes más importantes es hacer las relaciones, estas tendrán impacto directo en los sistemas contiguos, por lo que es necesario hacerlas bien.

```
/*=====
/* Foreign Key: Factura_codigoCliente
/*=====
ALTER TABLE FACTURA WITH CHECK ADD CONSTRAINT [ FK_factura_codcli_6AEFE058]
FOREIGN KEY ([codcli])
REFERENCES CLIENTES ([codcli])

ALTER TABLE FACTURA CHECK CONSTRAINT [ FK_factura_codcli_6AEFE058]
```

Y por último, estamos presenciando que los registros si se van almacenando, entonces esta funcionando bien nuestra conexión SQL- NetBeans para crear un sistema de facturación.

JHONATHAN\SQLEXP...	dbo.CLIENTES	InsercionDatos.sql...exionSQL_Java (56))	ScriptFinal.sql...exionSQL_Java (52))			
	codcli	Nombreciente	Apellidoscliente	sexo	CI	telefono
		Daniel	X	M		
	00001	Jhonathan	Pizarra	M	17253585	
	00002	Emilia	Lopez	F	17253578	1233
	00003	Brayan	Loachamin	M	12245667	NULL
*	NULL	NULL	NULL	NULL	NULL	NULL

¡GRACIAS!





### III. SOME COMMON MISTAKES

The word “data” is plural, not singular. The subscript for the permeability of vacuum  $\mu_0$  is zero, not a lowercase letter “o.” The term for residual magnetization is “remanence”; the adjective is “remanent”; do not write “remnance” or “remnant.” Use the word “micrometer” instead of “micron.” A graph within a graph is an “inset,” not an “insert.” The word “alternatively” is preferred to the word “alternately” (unless you really mean something that alternates). Use the word “whereas” instead of “while” (unless you are referring to simultaneous events). Do not use the word “essentially” to mean “approximately” or “effectively.” Do not use the word “issue” as a euphemism for “problem.” When compositions are not specified, separate chemical symbols by en-dashes; for example, “NiMn” indicates the intermetallic compound  $\text{Ni}_{0.5}\text{Mn}_{0.5}$  whereas “Ni–Mn” indicates an alloy of some composition  $\text{Ni}_x\text{Mn}_{1-x}$ .

Be aware of the different meanings of the homophones “affect” (usually a verb) and “effect” (usually a noun), “complement” and “compliment,” “discreet” and “discrete,” “principal” (e.g., “principal investigator”) and “principle” (e.g., “principle of measurement”). Do not confuse “imply” and “infer.”

Prefixes such as “non,” “sub,” “micro,” “multi,” and “ultra” are not independent words; they should be joined to the words they modify, usually without a hyphen. There is no period after the “et” in the Latin abbreviation “*et al.*” (it is also italicized). The abbreviation “i.e.,” means “that is,” and the abbreviation “e.g.,” means “for example” (these abbreviations are not italicized).

A general IEEE styleguide is available at <http://www.ieee.org/web/publications/authors/transjnl/index.html>

### IV. CONCLUSION

A conclusion section is not required. Although a conclusion may review the main points of the paper, do not replicate the abstract as the conclusion. A conclusion might elaborate on the importance of the work or suggest applications and extensions.

### REFERENCES AND FOOTNOTES

#### A. References

References need not be cited in text. When they are, number citations on the line, in square brackets inside the punctuation. Multiple references are each numbered with separate brackets. When citing a section in a book, please give the relevant page numbers. In text, refer simply to the reference number. Do not use “Ref.” or “reference” except at the beginning of a sentence: “Reference [3] shows ... .” Please do not use automatic endnotes

<sup>1</sup>It is recommended that footnotes be avoided (except for the unnumbered footnote with the receipt date on the first page). Instead, try to integrate the footnote information into the text.

in *Word*, rather, type the reference list at the end of the paper using the “References” style.

Reference numbers are set flush left and form a column of their own, hanging out beyond the body of the reference. The reference numbers are on the line, enclosed in square brackets. In all references, the given name of the author or editor is abbreviated to the initial only and precedes the last name. Use them all; use *et al.* only if names are not given. Use commas around Jr., Sr., and III in names. Abbreviate conference titles. When citing IEEE transactions, provide the issue number, page range, volume number, year, and/or month if available. When referencing a patent, provide the day and the month of issue, or application. References may not include all information; please obtain and include relevant information. Do not combine references. There must be only one reference with each number. If there is a URL included with the print reference, it can be included at the end of the reference.

Other than books, capitalize only the first word in a paper title, except for proper nouns and element symbols. For papers published in translation journals, please give the English citation first, followed by the original foreign-language citation. See the end of this document for formats and examples of common references. For a complete discussion of references and their formats, see “The IEEE Style Manual,” available as a PDF link off the [Author Digital Toolbox](#) main page.

#### A. Footnotes

Number footnotes separately in superscripts (Insert | Footnote).<sup>1</sup> Place the actual footnote at the bottom of the column in which it is cited; do not put footnotes in the reference list (endnotes). Use letters for table footnotes (see Table I).

### REFERENCES

#### Basic format for books:

- [1] J. K. Author, “Title of chapter in the book,” in *Title of His Published Book*, xth ed. City of Publisher, Country if not
- [2] USA: Abbrev. of Publisher, year, ch. x, sec. x, pp. xxx–xxx.

#### Examples:

- [3] G. O. Young, “Synthetic structure of industrial plastics,” in *Plastics*, 2nd ed., vol. 3, J. Peters, Ed. New York: McGraw-Hill, 1964, pp. 15–64.
- [4] W.-K. Chen, *Linear Networks and Systems*. Belmont, CA: Wadsworth, 1993, pp. 123–135.

#### Basic format for periodicals:

- [5] J. K. Author, “Name of paper,” *Abbrev. Title of Periodical*, vol. x, no. x, pp. xxx–xxx, Abbrev. Month, year.

#### Examples:

- [6] J. U. Duncombe, “Infrared navigation—Part I: An assessment of feasibility,” *IEEE Trans. Electron Devices*, vol. ED-11, no. 1, pp. 34–39, Jan. 1959.
- [7] E. P. Wigner, “Theory of traveling-wave optical laser,” *Phys. Rev.*, vol. 134, pp. A635–A646, Dec. 1965.

- [8] E. H. Miller, "A note on reflector arrays," *IEEE Trans. Antennas Propagat.*, to be published.

**Basic format for reports:**

- [9] J. K. Author, "Title of report," Abbrev. Name of Co., City of Co., Abbrev. State, Rep. xxx, year.

**Examples:**

- [10] E. E. Reber, R. L. Michell, and C. J. Carter, "Oxygen absorption in the earth's atmosphere," Aerospace Corp., Los Angeles, CA, Tech. Rep. TR-0200 (4230-46)-3, Nov. 1988.  
[11] J. H. Davis and J. R. Cogdell, "Calibration program for the 16-foot antenna," Elect. Eng. Res. Lab., Univ. Texas, Austin, Tech. Memo. NGL-006-69-3, Nov. 15, 1987.

**Basic format for handbooks:**

- [12] *Name of Manual/Handbook*, x ed., Abbrev. Name of Co., City of Co., Abbrev. State, year, pp. xxx-xxx.

**Examples:**

- [13] *Transmission Systems for Communications*, 3rd ed., Western Electric Co., Winston-Salem, NC, 1985, pp. 44-60.  
[14] *Motorola Semiconductor Data Manual*, Motorola Semiconductor Products Inc., Phoenix, AZ, 1989.

**Basic format for books (when available online):**

- [15] Author. (year, month day). *Title*. (edition) [Type of medium]. *volume (issue)*. Available: site/path/file

**Example:**

- [16] J. Jones. (1991, May 10). *Networks*. (2nd ed.) [Online]. Available: <http://www.atm.com>

**Basic format for journals (when available online):**

- [17] Author. (year, month). *Title. Journal*. [Type of medium]. *volume (issue)*, pages. Available: site/path/file

**Example:**

- [18] R. J. Vidmar. (1992, Aug.). On the use of atmospheric plasmas as electromagnetic reflectors. *IEEE Trans. Plasma Sci.* [Online]. 21(3), pp. 876-880. Available: <http://www.halcyon.com/pub/journals/21ps03-vidmar>

**Basic format for papers presented at conferences (when available online):**

- [19] Author. (year, month). *Title. Presented at Conference title*. [Type of Medium]. Available: site/path/file

**Example:**

- [20] PROCESS Corp., MA. Intranets: Internet technologies deployed behind the firewall for corporate productivity. Presented at INET96 Annual Meeting. [Online]. Available: <http://home.process.com/Intranets/wp2.http>

**Basic format for reports and handbooks (when available online):**

- [21] Author. (year, month). *Title*. Company. City, State or Country. [Type of Medium]. Available: site/path/file

**Example:**

- [22] S. L. Talleen. (1996, Apr.). *The Intranet Architecture: Managing information in the new paradigm*. Amdahl Corp., CA. [Online]. Available: <http://www.amdahl.com/doc/products/bsg/intra/infra/html>

**Basic format for computer programs and electronic documents (when available online):** ISO recommends that capitalization follow the accepted practice for the language or script in which the information is given.

**Example:**

- [23] A. Harriman. (1993, June). Compendium of genealogical software. *Humanist*. [Online]. Available e-mail: HUMANIST@NYVM.ORG Message: get GENEALOGY REPORT

**Basic format for patents (when available online):**

- [24] Name of the invention, by inventor's name. (year, month day). *Patent Number* [Type of medium]. Available: site/path/file

**Example:**

- [25] Musical toothbrush with adjustable neck and mirror, by L.M.R. Brooks. (1992, May 19). *Patent D 326 189* [Online]. Available: NEXIS Library: LEXPAT File: DESIGN

**Basic format for conference proceedings (published):**

- [26] J. K. Author, "Title of paper," in *Abbreviated Name of Conf.*, City of Conf., Abbrev. State (if given), year, pp. xxxxxx.

**Example:**

- [27] D. B. Payne and J. R. Stern, "Wavelength-switched passively coupled single-mode optical network," in *Proc. IOOC-ECOC*, 1985, pp. 585-590.

**Example for papers presented at conferences (unpublished):**

- [28] D. Ebehard and E. Voges, "Digital single sideband detection for interferometric sensors," presented at the 2nd Int. Conf. Optical Fiber Sensors, Stuttgart, Germany, Jan. 2-5, 1984.

**Basic format for patents:**

- [29] J. K. Author, "Title of patent," U.S. Patent x xxx xxx, Abbrev. Month, day, year.

**Example:**

- [30] G. Brandli and M. Dick, "Alternating current fed power supply," U.S. Patent 4 084 217, Nov. 4, 1978.

**Basic format for theses (M.S.) and dissertations (Ph.D.):**

- [31] J. K. Author, "Title of thesis," M.S. thesis, Abbrev. Dept., Abbrev. Univ., City of Univ., Abbrev. State, year.  
[32] J. K. Author, "Title of dissertation," Ph.D. dissertation, Abbrev. Dept., Abbrev. Univ., City of Univ., Abbrev. State, year.

**Examples:**

- [33] J. O. Williams, "Narrow-band analyzer," Ph.D. dissertation, Dept. Elect. Eng., Harvard Univ., Cambridge, MA, 1993.  
[34] N. Kawasaki, "Parametric study of thermal and chemical nonequilibrium nozzle flow," M.S. thesis, Dept. Electron. Eng., Osaka Univ., Osaka, Japan, 1993.

**Basic format for the most common types of unpublished references:**

- [35] J. K. Author, private communication, Abbrev. Month, year.  
[36] J. K. Author, "Title of paper," unpublished.  
[37] J. K. Author, "Title of paper," to be published.

**Examples:**

- [38] A. Harrison, private communication, May 1995.  
[39] B. Smith, "An approach to graphs of linear forms," unpublished.  
[40] A. Brahms, "Representation error for real numbers in binary computer arithmetic," IEEE Computer Group Repository, Paper R-67-85.

**Basic format for standards:**

- [41] *Title of Standard*, Standard number, date.

**Examples:**

- [42] IEEE Criteria for Class IE Electric Systems, IEEE Standard 308, 1969.  
[43] Letter Symbols for Quantities, ANSI Standard Y10.5-1968.



**First A. Author** (M'76–SM'81–F'87) and the other authors may include biographies at the end of regular papers. Biographies are often not included in conference-related papers. This author became a Member (M) of IEEE in 1976, a Senior Member (SM) in 1981, and a Fellow (F) in 1987. The first paragraph may contain a place and/or date of birth (list place, then

date). Next, the author's educational background is listed. The degrees should be listed with type of degree in what field, which institution, city, state, and country, and year the degree was earned. The author's major field of study should be lower-cased.

The second paragraph uses the pronoun of the person (he or she) and not the author's last name. It lists military and work experience, including summer and fellowship jobs. Job titles are capitalized. The current job must have a location; previous positions may be listed without one. Information concerning previous publications may be included. Try not to list more than three books or published articles. The format for listing publishers of a book within the biography is: title of book (city, state: publisher name, year) similar to a reference. Current and previous research interests end the paragraph.

The third paragraph begins with the author's title and last name (e.g., Dr. Smith, Prof. Jones, Mr. Kajor, Ms. Hunter). List any memberships in professional societies other than the IEEE. Finally, list any awards and work for IEEE committees and publications. If a photograph is provided, the biography will be indented around it. The photograph is placed at the top left of the biography, and should be of good quality, professional-looking, and black and white (see above example). Personal hobbies will be deleted from the biography. Following are two examples of an author's biography.



**Second B. Author** was born in Greenwich Village, New York City, in 1977. He received the B.S. and M.S. degrees in aerospace engineering from the University of Virginia, Charlottesville, in 2001 and the Ph.D. degree in mechanical engineering from Drexel University, Philadelphia, PA, in 2008.

From 2001 to 2004, he was a Research Assistant with the Princeton Plasma Physics Laboratory. Since 2009, he has been

an Assistant Professor with the Mechanical Engineering Department, Texas A&M University, College Station. He is the author of three books, more than 150 articles, and more than 70 inventions. His research interests include high-pressure and high-density nonthermal plasma discharge processes and applications, microscale plasma discharges, discharges in liquids, spectroscopic diagnostics, plasma propulsion, and innovation plasma applications. He is an Associate Editor of the journal *Earth, Moon, Planets*, and holds two patents.



Mr. Author was a recipient of the International Association of Geomagnetism and Aeronomy Young Scientist Award for Excellence in 2008, the IEEE Electromagnetic Compatibility Society Best Symposium Paper Award in 2011, and the American Geophysical Union Outstanding Student Paper Award in Fall 2005.

**Third C. Author, Jr. (M'87)** received the B.S. degree in mechanical engineering from National Chung Cheng University, Chiayi, Taiwan, in 2004 and the M.S. degree in mechanical engineering from National Tsing Hua University, Hsinchu, Taiwan, in 2006. He is currently pursuing the Ph.D. degree in mechanical engineering at Texas A&M University, College Station.

From 2008 to 2009, he was a Research Assistant with the Institute of Physics, Academia Sinica, Taipei, Taiwan. His research interest includes the development of surface processing and biological/medical treatment techniques using nonthermal atmospheric pressure plasmas, fundamental study of plasma sources, and fabrication of micro- or nanostructured surfaces.

Mr. Author's awards and honors include the Frew Fellowship (Australian Academy of Science), the I. I. Rabi Prize (APS), the European Frequency and Time Forum Award, the Carl Zeiss Research Award, the William F. Meggers Award and the Adolph Lomb Medal (OSA).