

Proyecto final Programación

**INDICE**

Introducción, Funciones y rendimientos deseados --------------------- [3](#Introduccion)

Objetivos deseados ------------------------------------------------------------- [5](#Obj)

Base de datos de Tesla -------------------------------------------------------- [7](#BDTesla)

Programa para Tesla: Creación y conexión con la base de datos- [10](#PpTCcbd)

Anexo de la Base de Datos ------------------------------------------------- [31](#AnexoBD)

Anexo del programa ---------------------------------------------------------- [33](#AnexoP)

1. **INTRODUCCIÓN**

Tesla es una empresa dedicada a la fabricación y venta de vehículos eléctricos. Para mejorar la gestión de la información empleada por la empresa, Elon Musk ha decidido desarrollar una aplicación que permita registrar y almacenar la información de los coches, clientes y revisiones que se realizan en el concesionario. La aplicación permitirá llevar un control de las ventas realizadas y las revisiones realizadas a cada coche.

Para desarrollar esta aplicación se ha realizado un análisis del sistema que incluye la creación de diagramas de clases, casos de uso y diagramas de secuencia.

1. **FUNCIONES Y RENDIMIENTOS DESEADOS**
2. Función para agregar un nuevo coche:

* La función debe permitir ingresar la matrícula, marca, modelo, color y precio de venta del coche.
* El rendimiento deseado es que la información del coche sea agregada correctamente a la base de datos de la empresa, de forma rápida y eficiente.

1. Función para agregar un nuevo cliente:

* La función debe permitir ingresar el NIF, nombre, dirección, ciudad y número de teléfono del cliente.
* El rendimiento deseado es que la información del cliente sea agregada correctamente a la base de datos de la empresa, generando un código interno de la empresa de forma automática y rápida.

1. Función para registrar una venta:

* La función debe permitir asociar un coche a un cliente y actualizar la información de ambos en la base de datos de la empresa.
* El rendimiento deseado es que la venta sea registrada correctamente y que se actualice la información de los coches y clientes involucrados de forma rápida y eficiente.

1. Función para agregar una nueva revisión:

* La función debe permitir ingresar los detalles de la revisión, como el código de la revisión, si se hizo cambio de filtro, cambio de aceite, cambio de frenos u otros, y la matrícula del coche que se revisó.
* El rendimiento deseado es que la información de la revisión sea agregada correctamente a la base de datos de la empresa, actualizando la información del coche involucrado de forma rápida y eficiente.

1. Función para obtener la lista de coches disponibles para la venta:

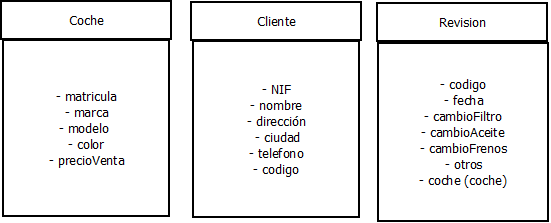
* La función debe permitir obtener la información de todos los coches disponibles para la venta en la base de datos de la empresa.
* El rendimiento deseado es que la información de todos los coches sea recuperada de forma rápida y eficiente.

1. Función para obtener la lista de clientes y sus coches comprados:

* La función debe permitir obtener la información de todos los clientes y los coches que han comprado en la empresa.
* El rendimiento deseado es que la información de todos los clientes y sus coches comprados sea recuperada de forma rápida y eficiente.

En general, todas las funciones deben ser seguras, confiables y mantener la integridad de los datos almacenados en la base de datos de la empresa. Además, se debe asegurar la escalabilidad del sistema, para que pueda manejar grandes volúmenes de información sin comprometer su rendimiento.

**Diagrama de clases:**



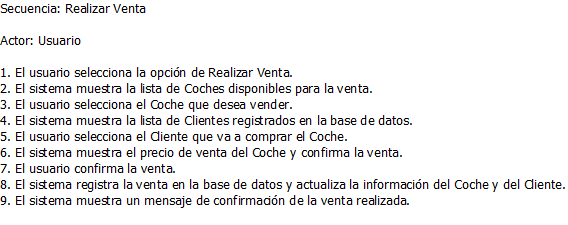
El diagrama de clases muestra las tres entidades principales de la aplicación: Coche, Cliente y Revisión. Un Coche puede estar asociado a muchas Revisiones y una Revisión solo puede estar asociada a un Coche. Además, un Cliente puede comprar muchos Coches y cada Coche solo puede ser comprado por un único Cliente.

Casos de uso:

* Registrar Coche: permite registrar la información de un nuevo Coche en la base de datos.
* Registrar Cliente: permite registrar la información de un nuevo Cliente en la base de datos.
* Registrar Revisión: permite registrar la información de una nueva Revisión en la base de datos.
* Consultar Coche: permite consultar la información de un Coche registrado en la base de datos.
* Consultar Cliente: permite consultar la información de un Cliente registrado en la base de datos.
* Consultar Revisión: permite consultar la información de una Revisión registrada en la base de datos.
* Realizar Venta: permite realizar una venta de un Coche a un Cliente registrado en la base de datos.

**Diagrama de secuencia:**

El diagrama de secuencia muestra el flujo de interacciones entre el usuario y el sistema para realizar una venta de un Coche a un Cliente.



1. **OBJETIVOS DESEADOS**

Los objetivos deseados del proyecto son los siguientes:

* Desarrollar una aplicación que permita almacenar y gestionar la información de los coches que la empresa Tesla tiene para la venta. La aplicación debe incluir campos para la matrícula, marca, modelo, color y precio de venta de cada coche.
* Crear una base de datos que permita almacenar la información de los clientes de Tesla. La base de datos debe incluir campos para el NIF, nombre, dirección, ciudad y número de teléfono de cada cliente. Además, la base de datos debe permitir asignar a cada cliente un código interno de la empresa que se incrementa automáticamente cuando un cliente se da de alta en ella.
* Integrar la base de datos de clientes con la información de los coches para que se pueda asociar a cada coche un único cliente que lo haya comprado.
* Desarrollar una funcionalidad en la aplicación que permita realizar el registro de las revisiones que se realizan a cada coche. Cada revisión debe tener asociado un código que se incrementa automáticamente por cada revisión que se haga.
* Incluir campos en la funcionalidad de registro de revisión para especificar si se ha hecho cambio de filtro, si se ha hecho cambio de aceite, si se ha hecho cambio de frenos u otros.
* Crear una vista en la aplicación que permita ver todas las revisiones que se han realizado a un coche en particular.
* Desarrollar una funcionalidad en la aplicación que permita generar informes con estadísticas sobre las ventas de coches y las revisiones realizadas en el concesionario.
* Garantizar la seguridad y privacidad de la información almacenada en la aplicación.

1. **BASE DE DATOS DE TESLA**

El primer punto para la creacion del sistema de gestion de la base de datos de Tesla es crear nuestra base de datos la cual hacemos las siguientes:

1. Tabla "Coches" [(Ver anexo)](#AnexoBD):

Columnas:

* matrícula (clave primaria)
* marca
* modelo
* color
* precio de venta.

Se usa la siguiente sentencia para crear la tabla:

CREATE TABLE Coches (

matricula VARCHAR(10) PRIMARY KEY,

marca VARCHAR(50),

modelo VARCHAR(50),

color VARCHAR(20),

precioventa DECIMAL(10, 2)

);

La tabla debe quedar así:  
  


Tabla "Clientes" [(ver anexo)](#AnexoBD):

Columnas:

* código interno (clave primaria)
* NIF
* Nombre
* Dirección
* Ciudad
* Número de teléfono.

La sentencia a usar es la siguiente:

CREATE TABLE Clientes (

codigointerno INT PRIMARY KEY AUTO\_INCREMENT,

NIF VARCHAR(10),

nombre VARCHAR(100),

direccion VARCHAR(200),

ciudad VARCHAR(50),

numerotelefono VARCHAR(20)

);

Una vez echo esta sentencia al crearse se debe mostrarse así:



Tabla "Revisiones" [(Ver anexo)](#AnexoBD):

Columnas:

* Código de revisión,
* Matrícula del coche (clave foránea referenciando la tabla "Coches")
* cambio de filtro
* cambio de aceite
* cambio de frenos
* otros.

La sentencia a usar es la siguiente:

CREATE TABLE Revisiones (

IDRevision INT PRIMARY KEY AUTO\_INCREMENT,

MatriculaCoche VARCHAR(10),

CambioFiltro BOOLEAN,

CambioAceite BOOLEAN,

CambioFrenos BOOLEAN,

Otros VARCHAR(200),

FOREIGN KEY (MatriculaCoche) REFERENCES Coches(matricula)

);

Deberá mostrarse como a continuación



Tabla “Compras” [(ver anexo)](#AnexoBD):

Columnas:

* ID: Identificador único de la compra (clave primaria).
* IDCompraCliente: Identificador de la compra del cliente.
* CodigoInternoCliente: Código interno del cliente que realizó la compra.
* MatriculaCoche: Matrícula del coche comprado.

La sentencia a usar es la siguiente:  
CREATE TABLE Compras (

ID INT PRIMARY KEY AUTO\_INCREMENT,

IDCompraCliente INT,

CodigoInternoCliente INT,

MatriculaCoche VARCHAR(10),

FOREIGN KEY (IDCompraCliente) REFERENCES Clientes(codigo\_interno),

FOREIGN KEY (MatriculaCoche) REFERENCES Coches(matricula)

);

Debe quedar la tabla así:



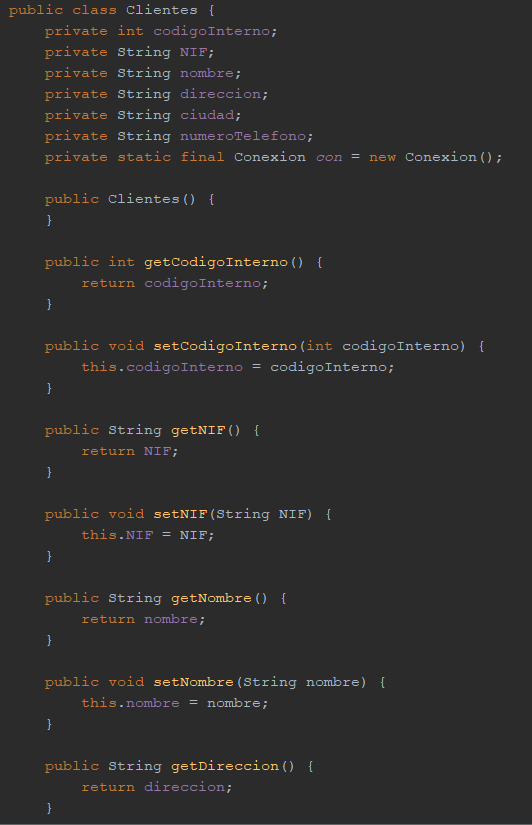
Una vez realizado todo esto tendríamos echo la base de dato a lo que luego en el punto siguiente se explica como hacer el programa junto a la correspondiente conexión a la base de datos para tener realizado el sistema de gestión del concesionario.

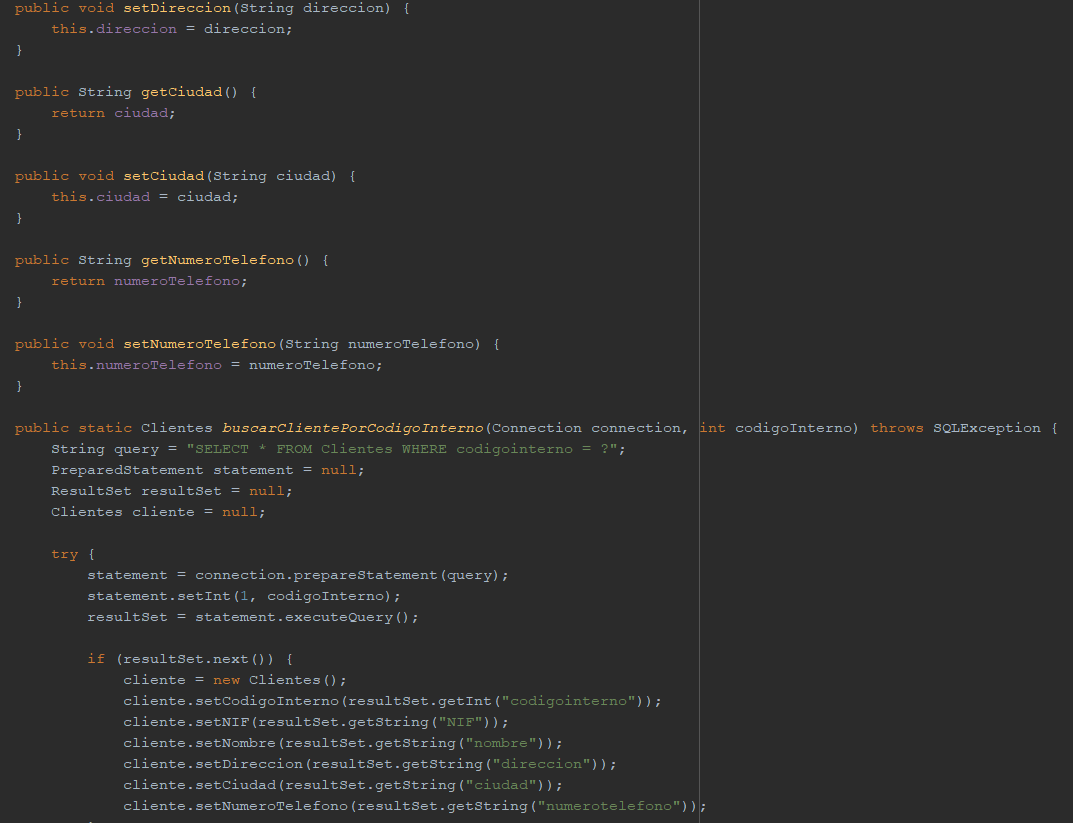
1. **PROGRAMA PARA TESLA: Creación y conexión con la base de datos**

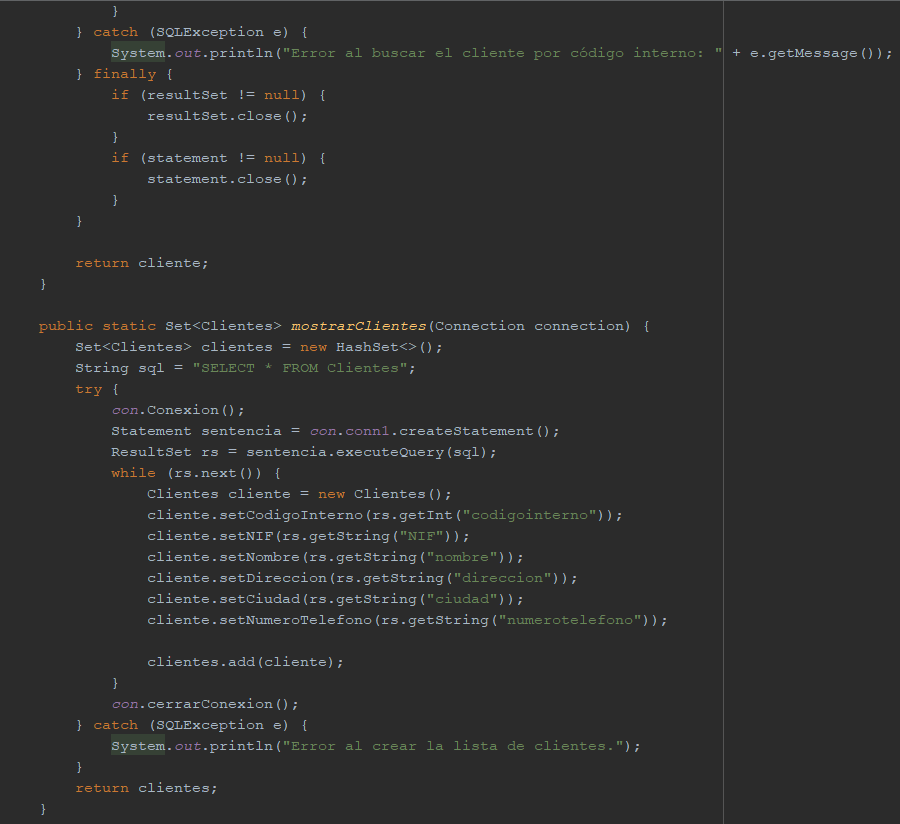
Ya vimos como creabamos nuestra base de datos, lo siguiente que queda para completar el sistema es crear nuestro programa con netbeans la cual vamos a realizar las diferentes clases en función de lo realizado con la base de datos para despues crear clases para realizar correctamente la conexión con la base de datos y realizar diferentes operaciones dentro del concesionario:

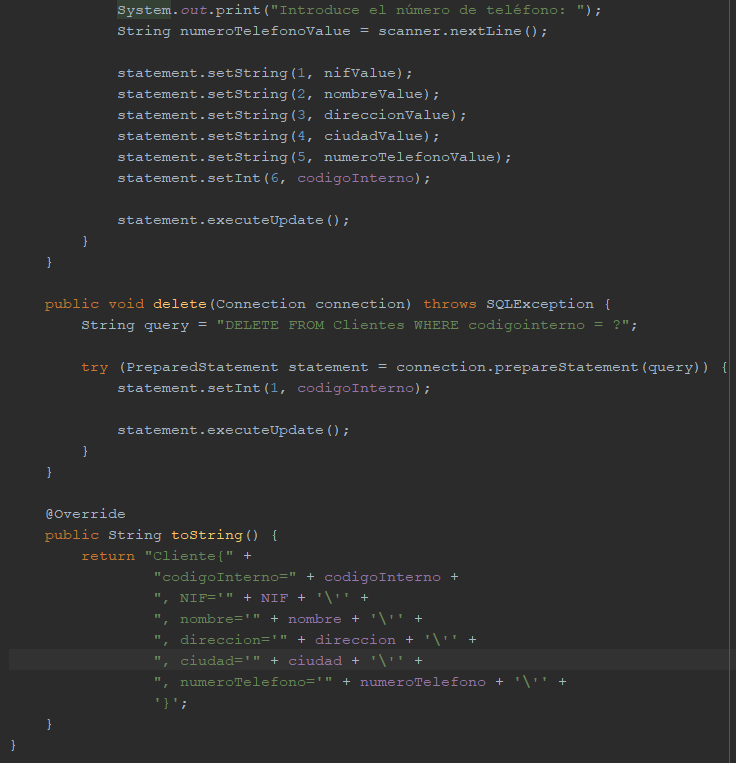
[(ver anexo)](#AnexoP)

1. Para empezar creamos un paquete la cual llamaras ConcesionarioTesla y creamos una clase cualquiera (no importa el orden). En mi caso creo primero la clase Cliente que debe incluir lo mismo que cada columna de la tabla de la base de datos:

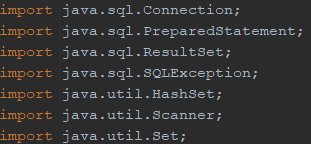




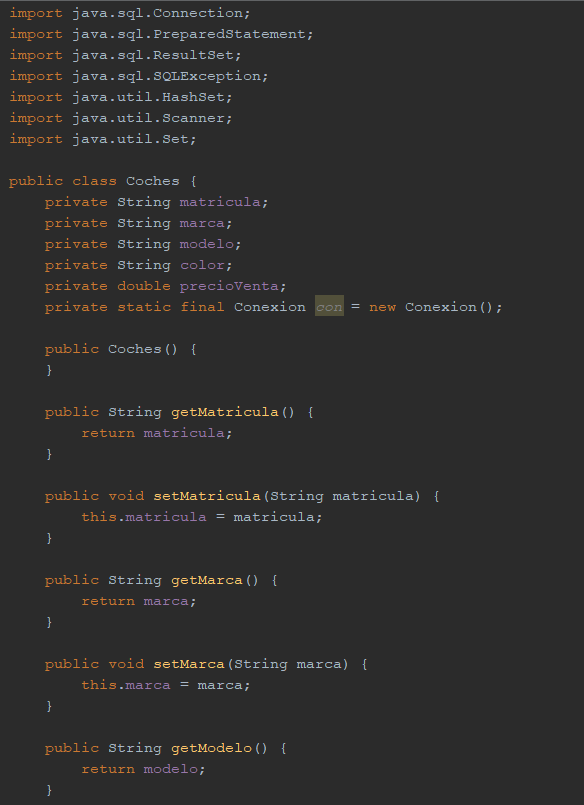


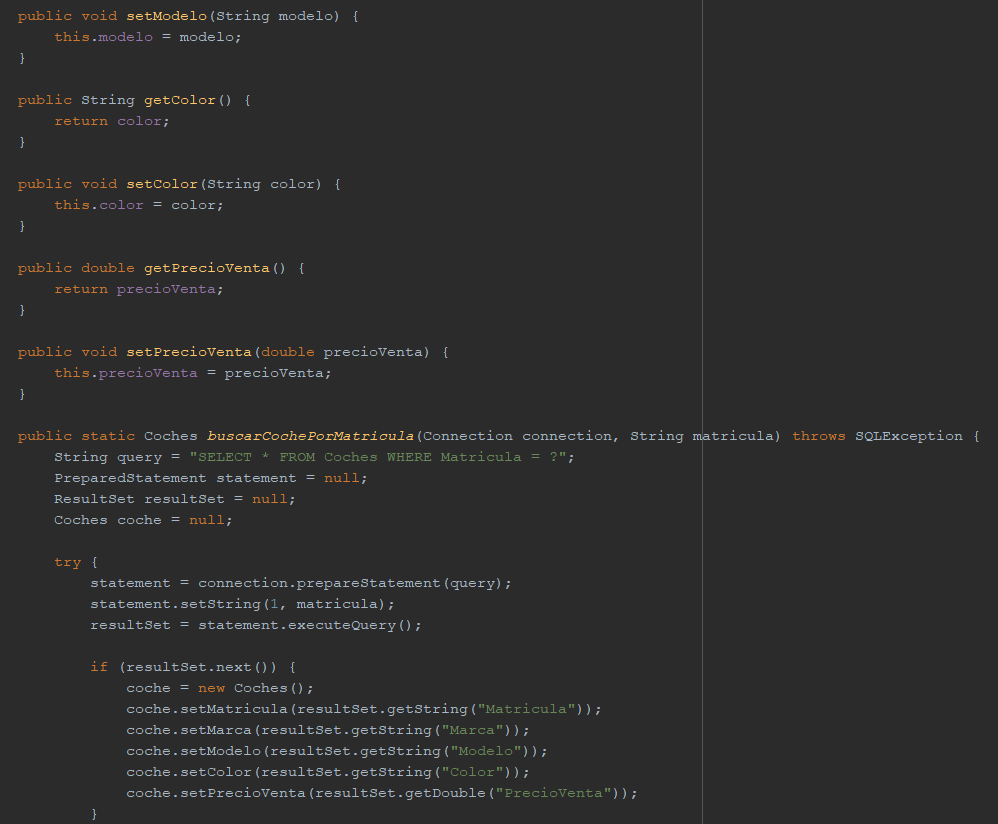


No se debe olvidar que debe tener en el codigo los correspondientes metodos, getters y setters para poder funcionar dentro de la base de datos. Tambien es importante tener importado lo siguiente que ves en la captura que se situa al principio de este y los siguientes que son importantes para que funcione correctamente:

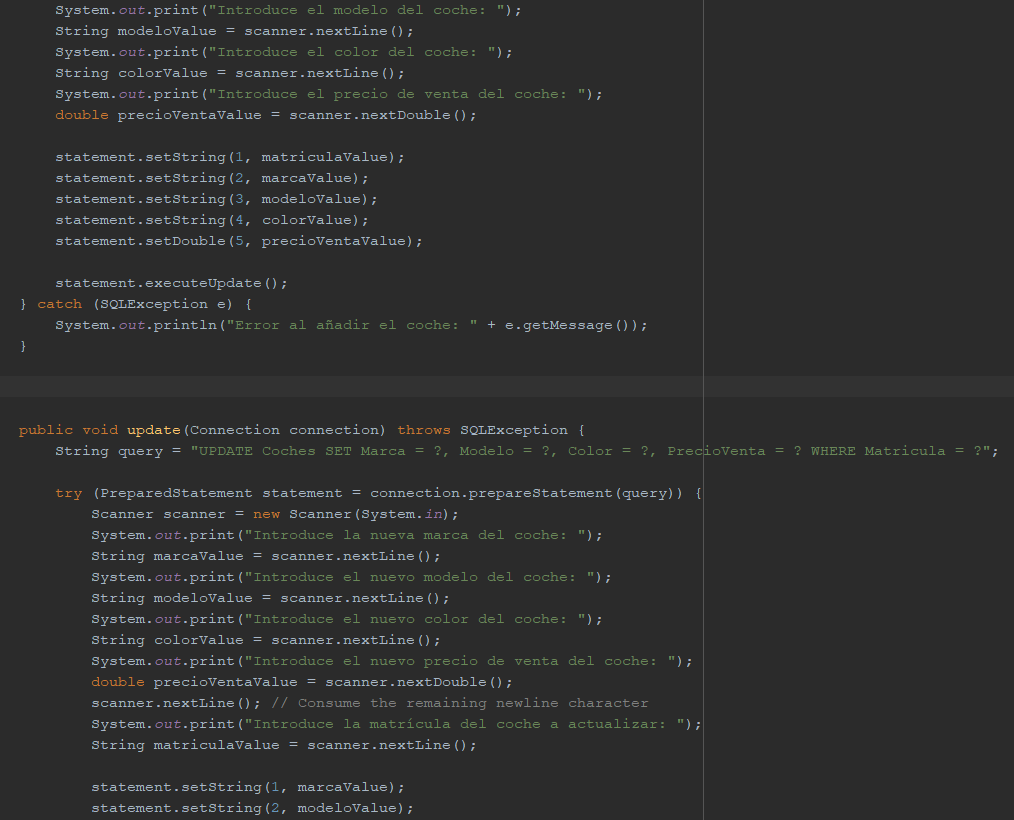


1. Ahora se va a crear la clase coches la cual deberemos colocar en ella cada una de las columnas de la tabla junto con sus getters, setters y las importaciones para que funcione correctamente una vez conectado a la base de datos. Vera a continuacion el codigo que se debe escribar:



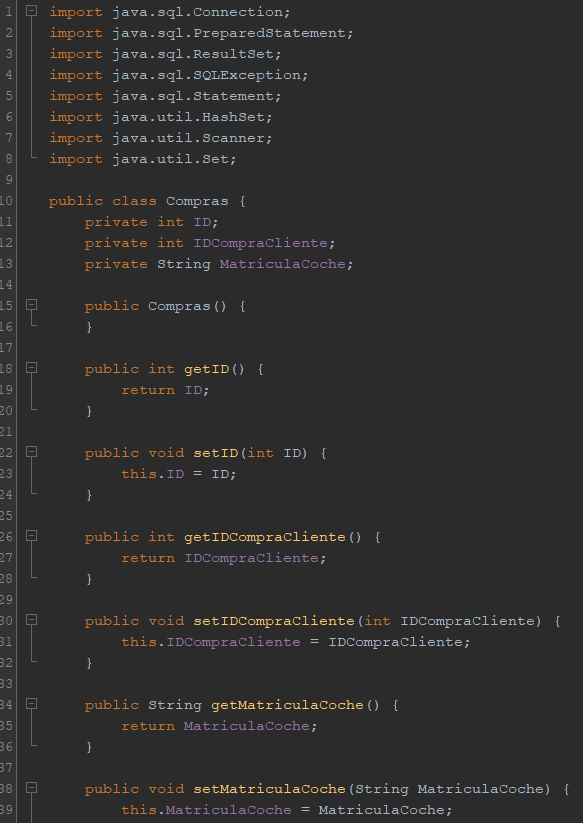






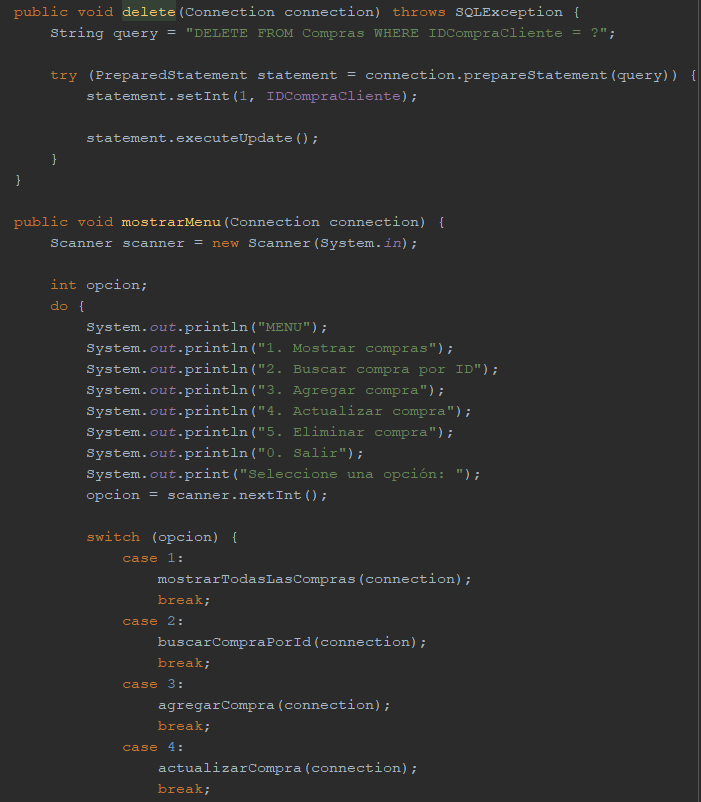


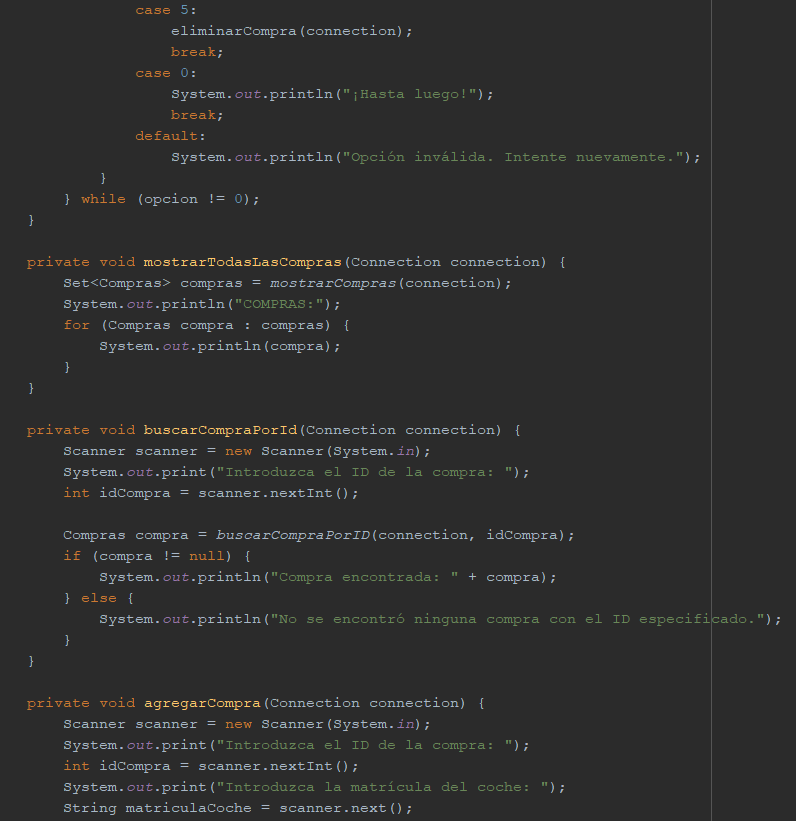
1. Se va a crear la clase Compras donde solo tenemos tres columnas y el codigo es el siguiente teniendo en cuenta los getters, setters y las importaciones para su correcto funcionamiento:

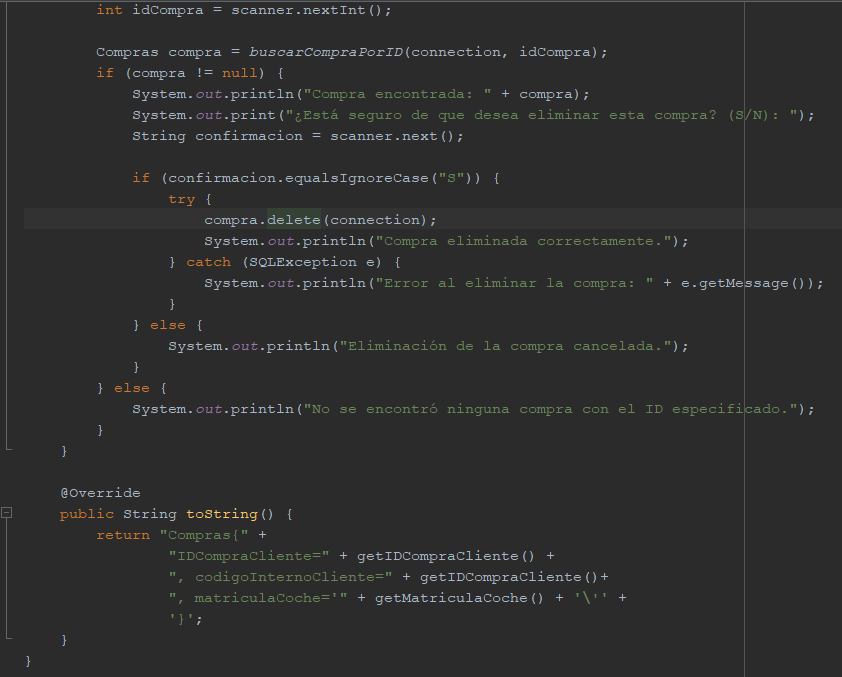












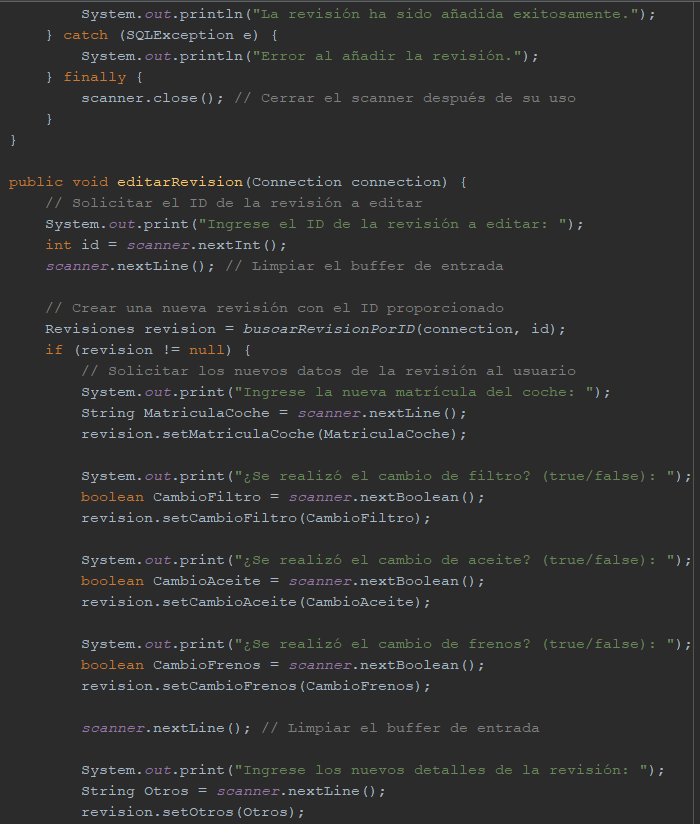
1. Por ultimo lugar antes de lo siguiente para la conexión, hacemos la clase para las revisiones dentro del concesionario para los diferentes coches. A continuación tienes el codigo:

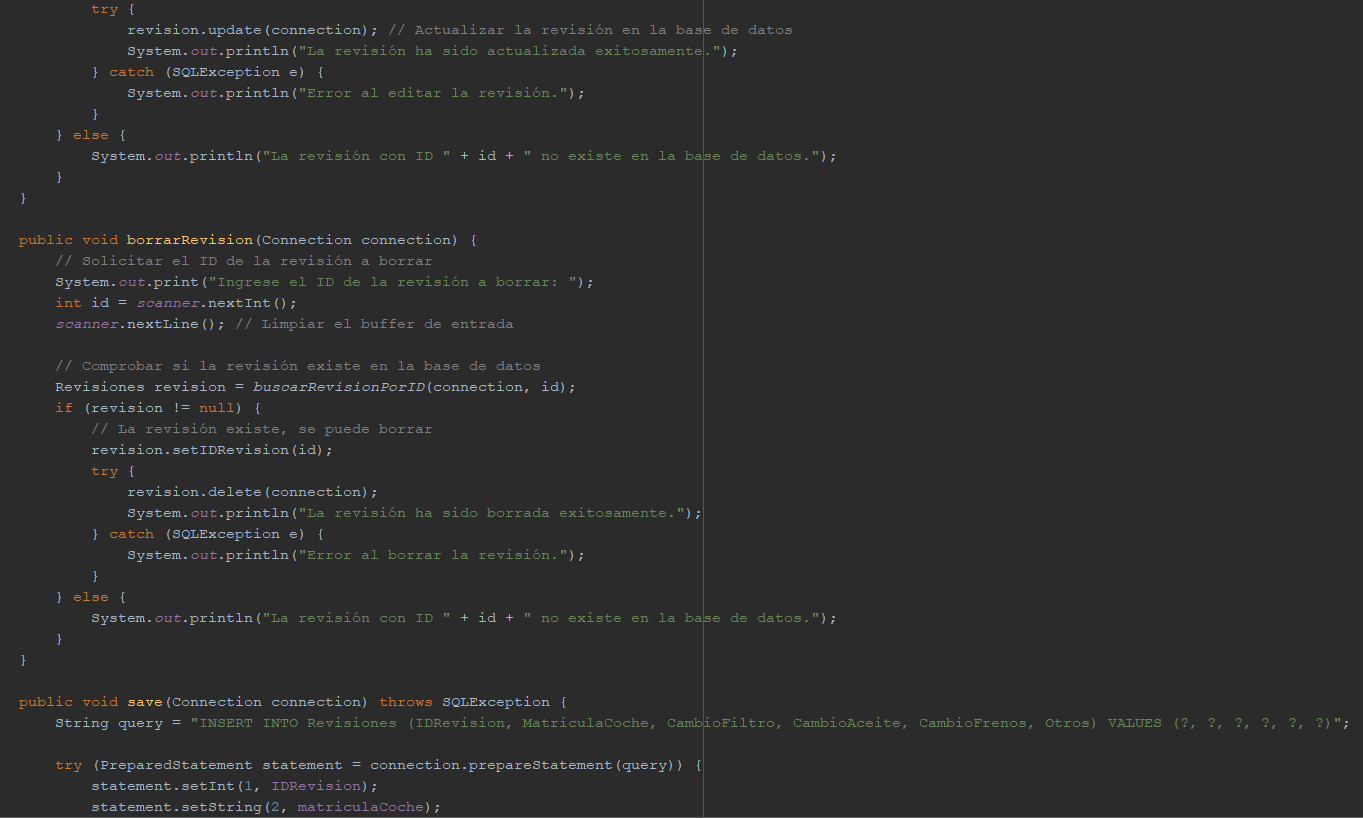


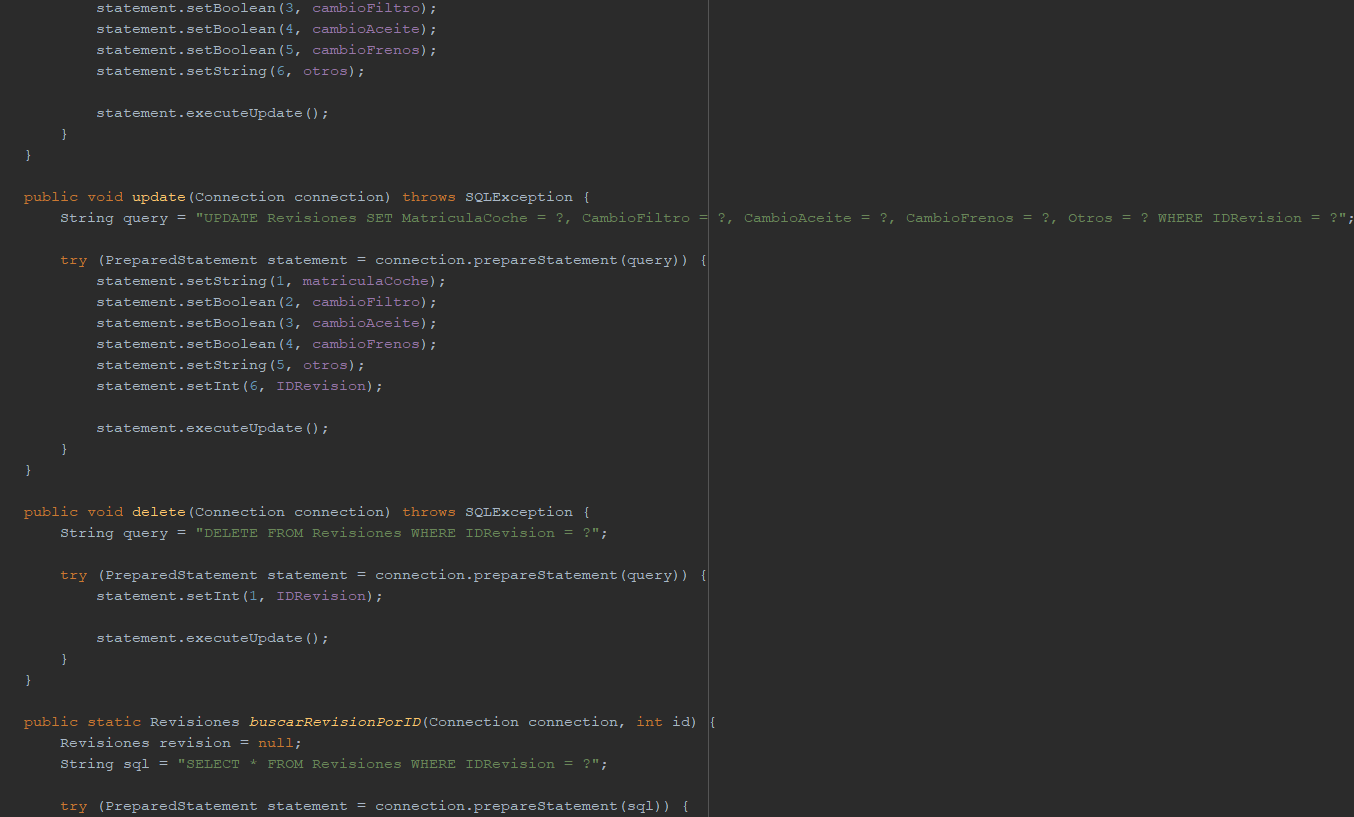


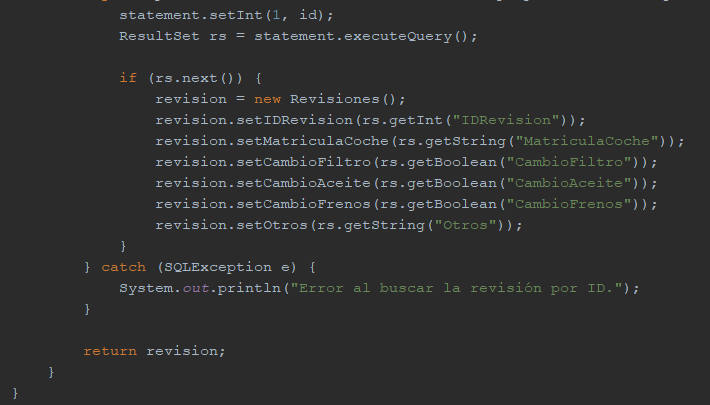




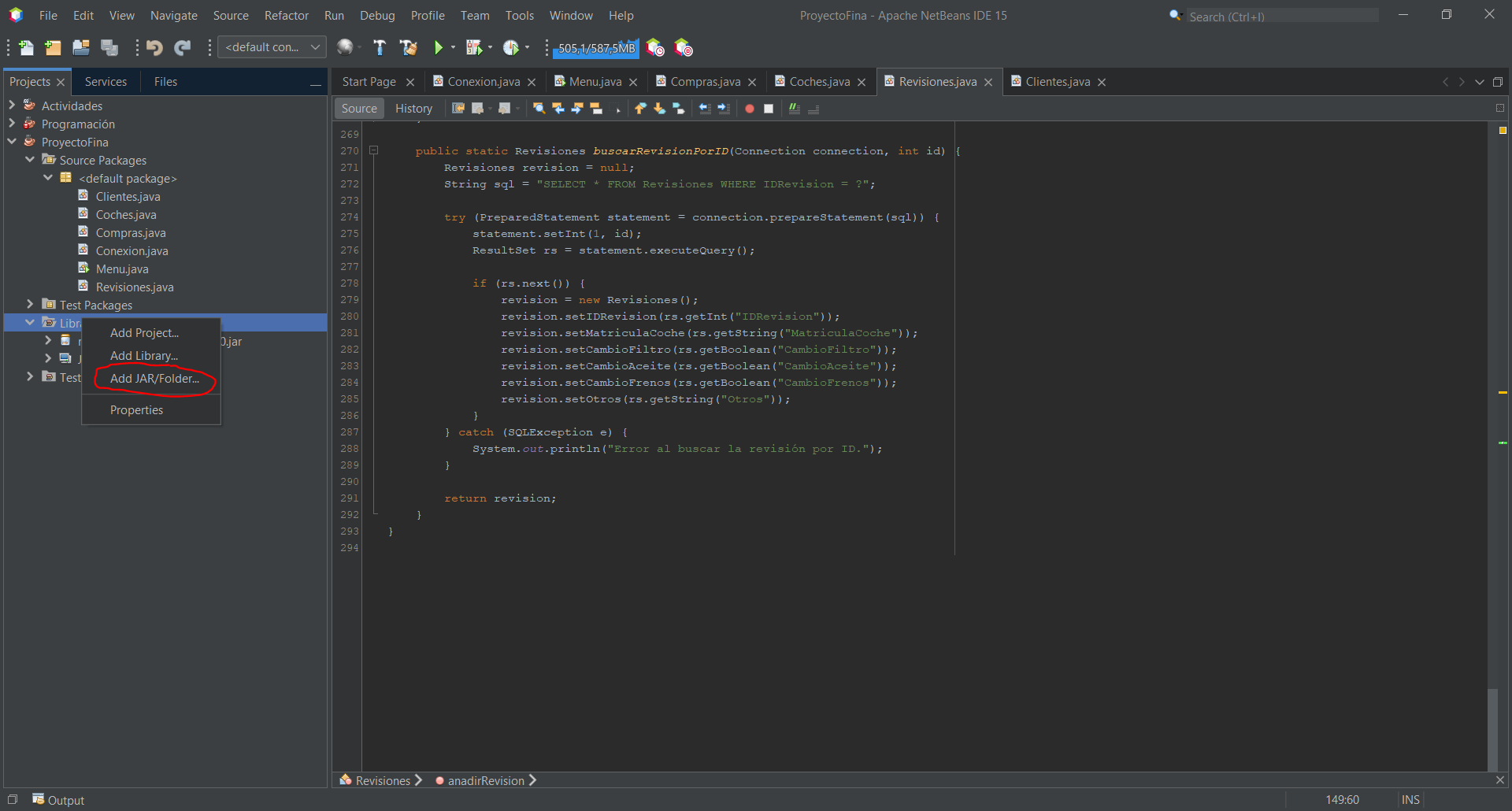


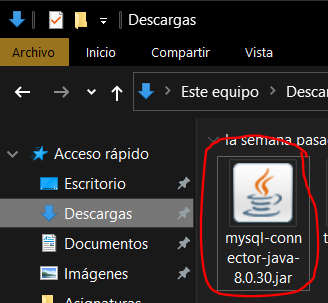




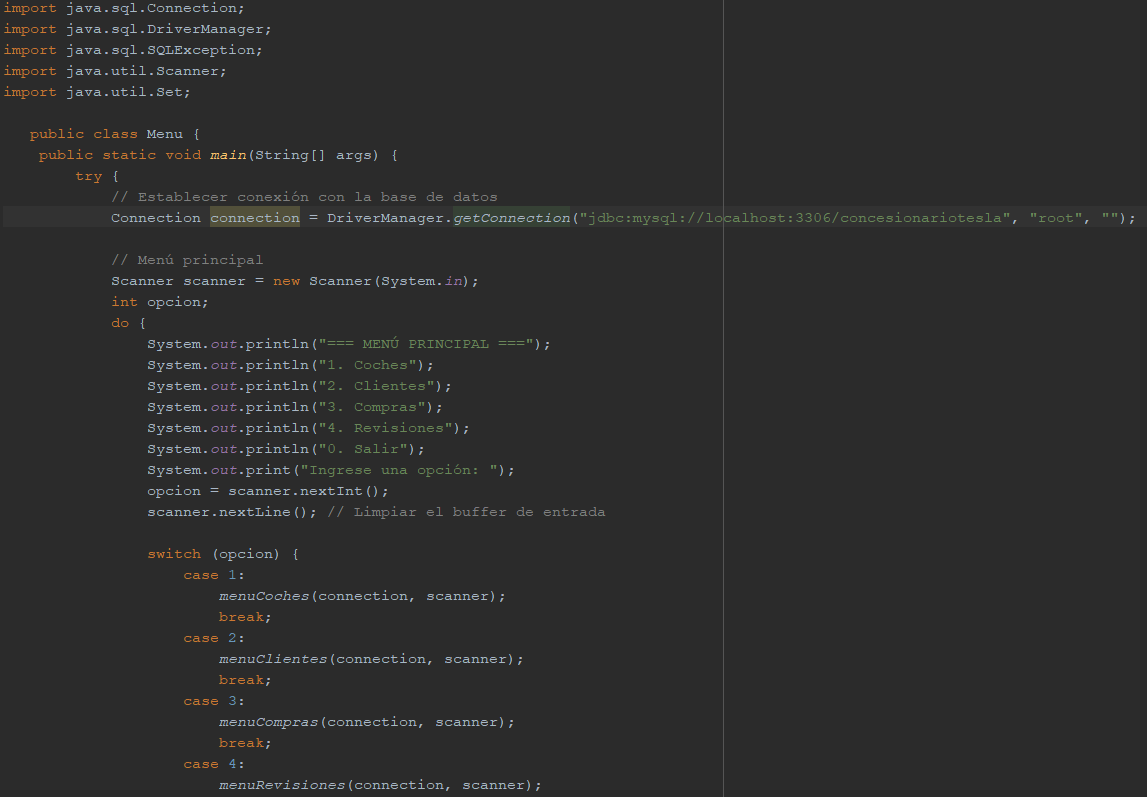


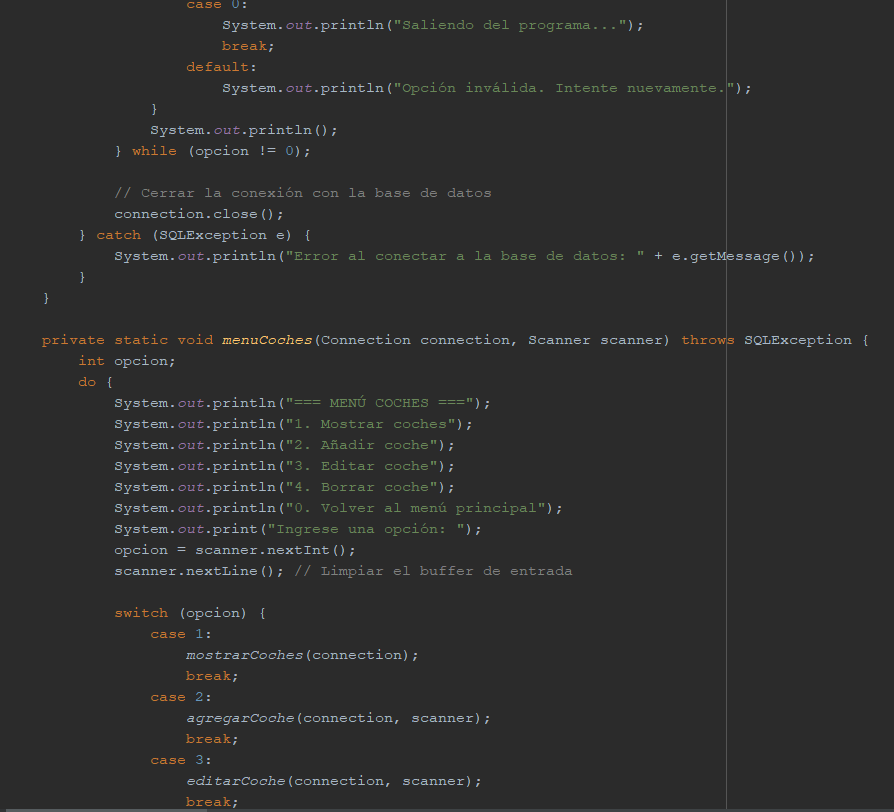
Una vez echo estos clases principales, ahora se va a realizar la conexión del programa con la base de datos en la que primero antes de realizar la conexión se importa un archivo .jar llamado “[mysql-connector-java-8.0.30](https://drive.google.com/file/d/1pjJxjSNnWppEZ6MKGwrPDxZLeiEdTFZh/view?usp=drive_web&authuser=0)” que no lo vamos a poner dentro del proyecto sino que se debe poner en una carpeta de Netbeans llamado “Library”. En mi caso ya lo tengo importado pero muestro como se importa el mismo que previamente deberás haber descargado (esta con vinculo echo para descargar):

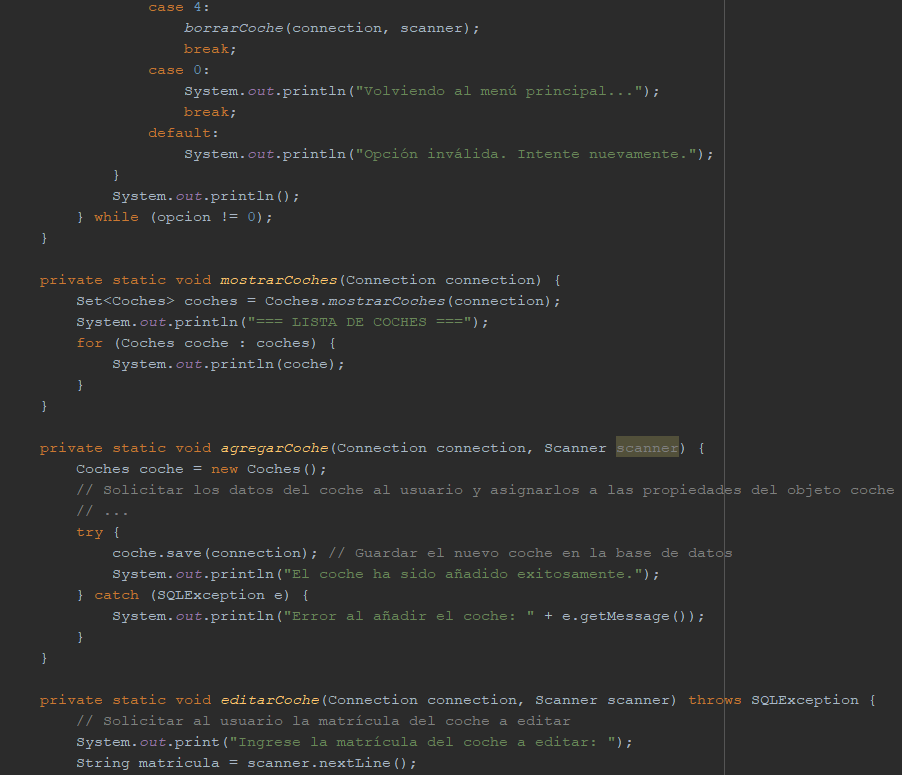


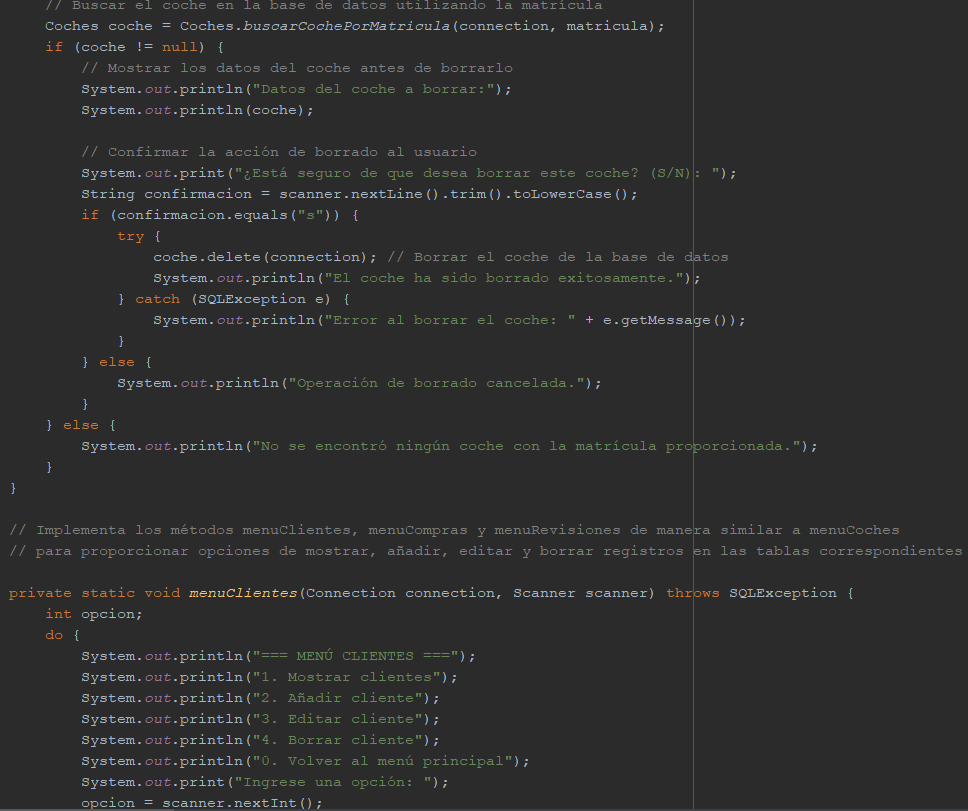


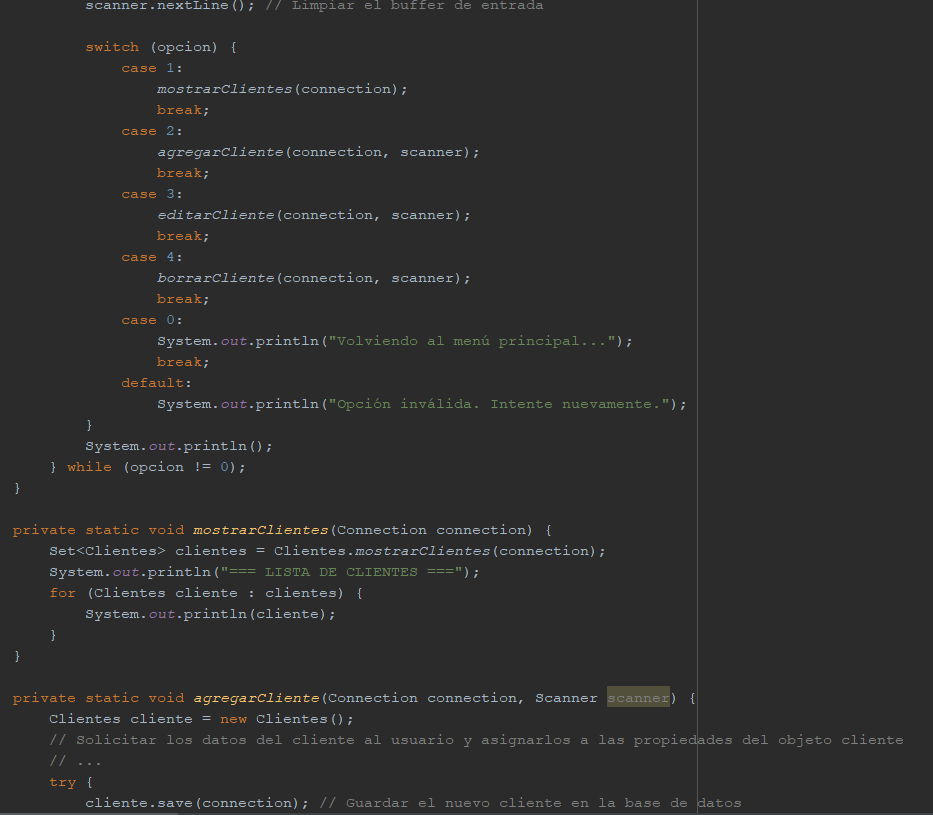
Una vez echo esto ya tendremos echo lo inicial para conectar la base de datos con el programa de Netbeans. Ahora, para que funcione esto primero vamos a crear la clase menú que sera con la que vamos a poder realizar las acciones dentro del sistema funcionando de forma simultanea el programa con la base de datos. Es importante que tengamos importados al igual que con las clases anteriores las clases aquellas fundamentales como lo son Connection, DriveManager, SQLException, Scanner y Set. A continuacion se muestra el codigo que se ha procedido a escribir para esto:

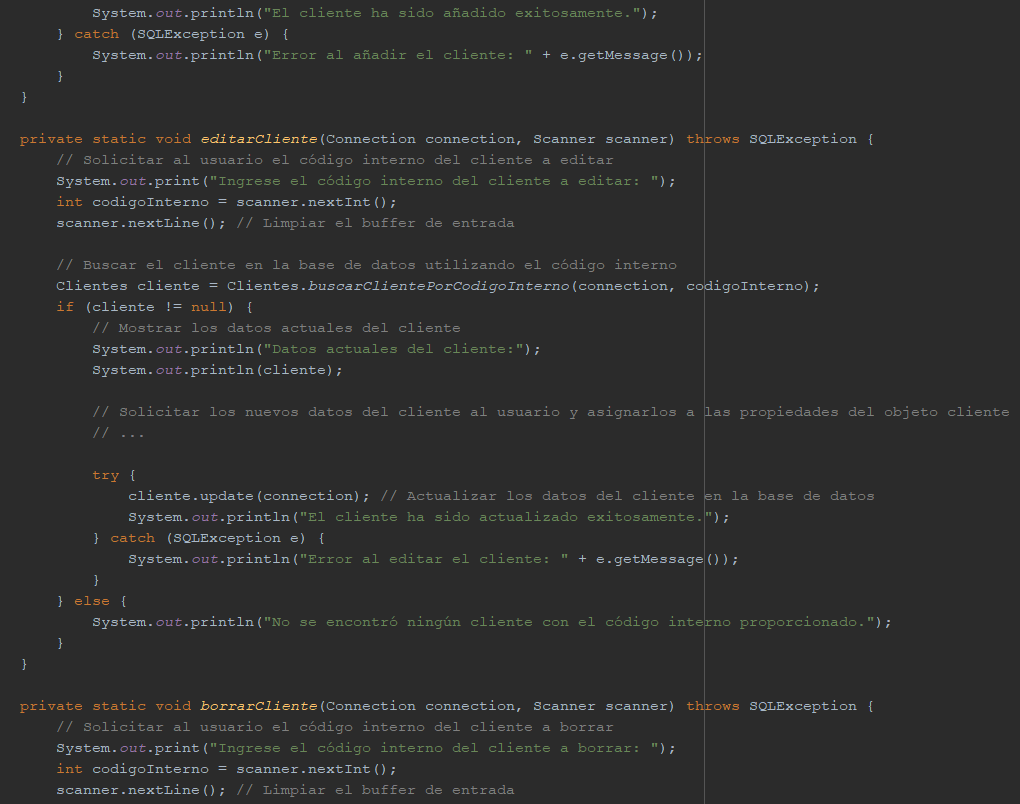


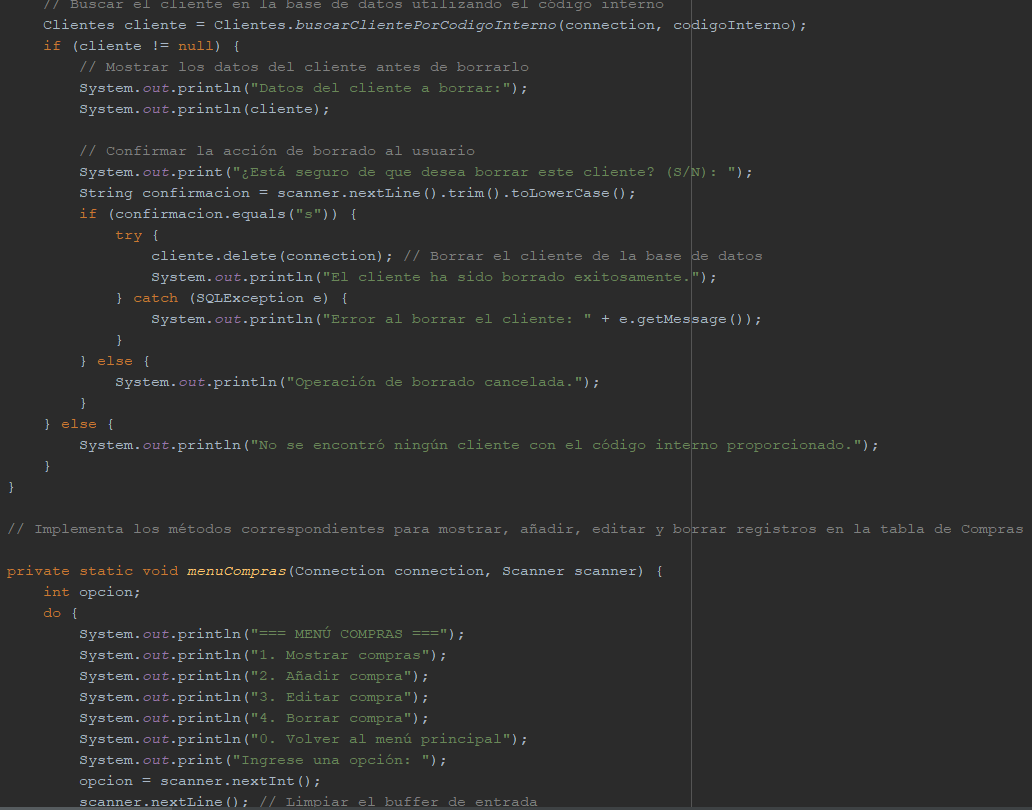


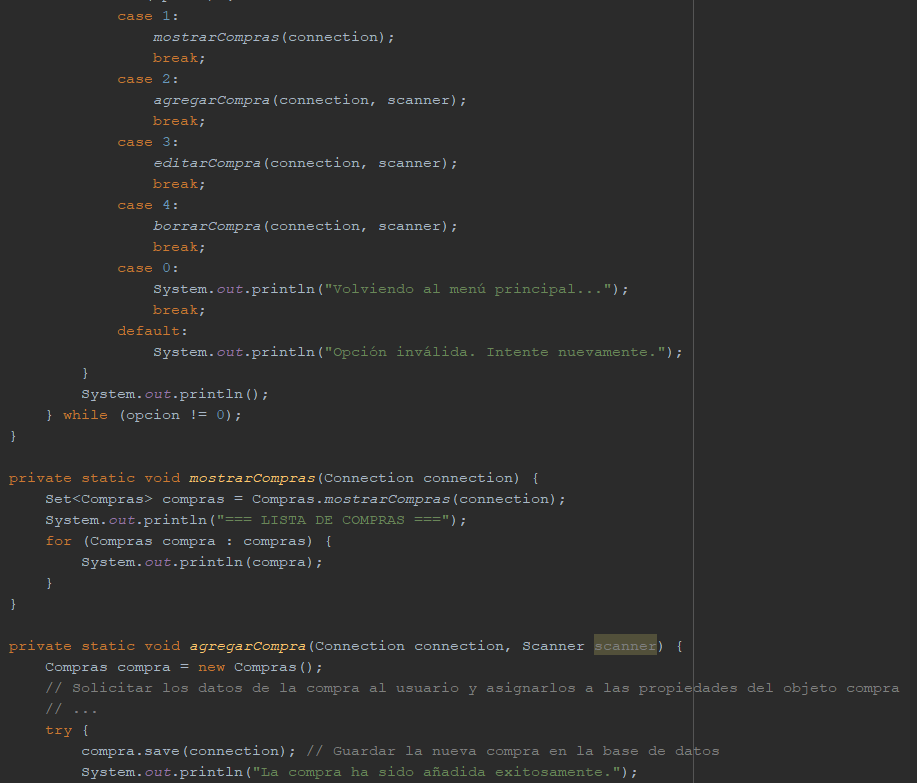


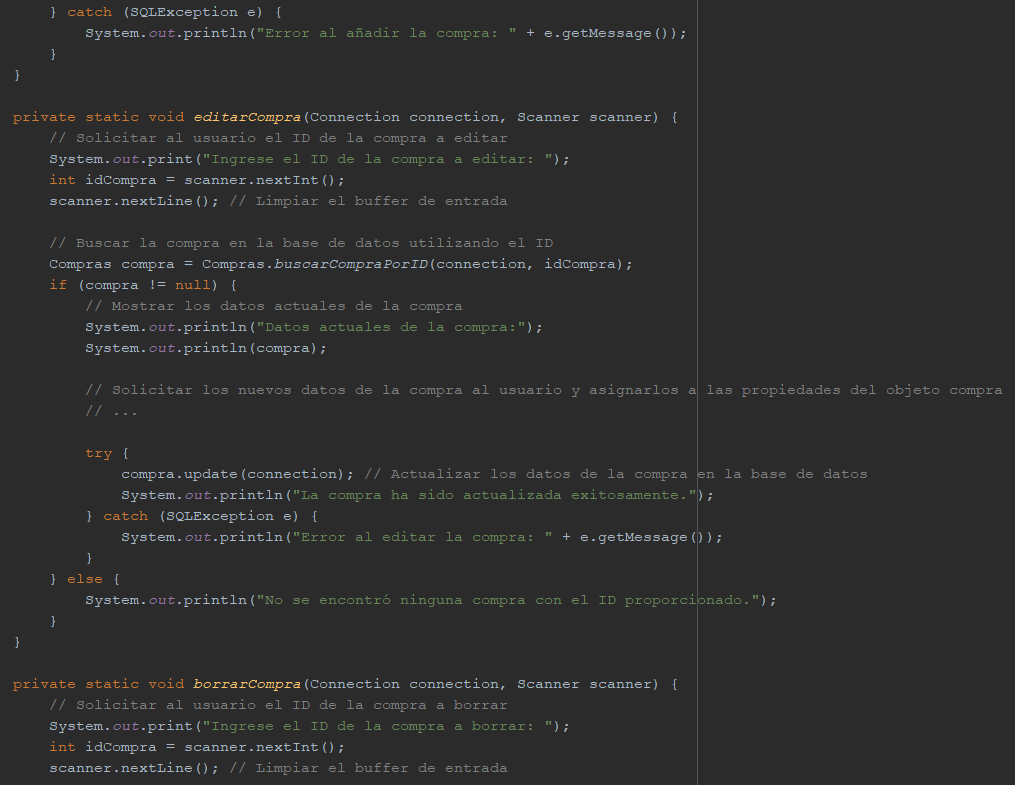


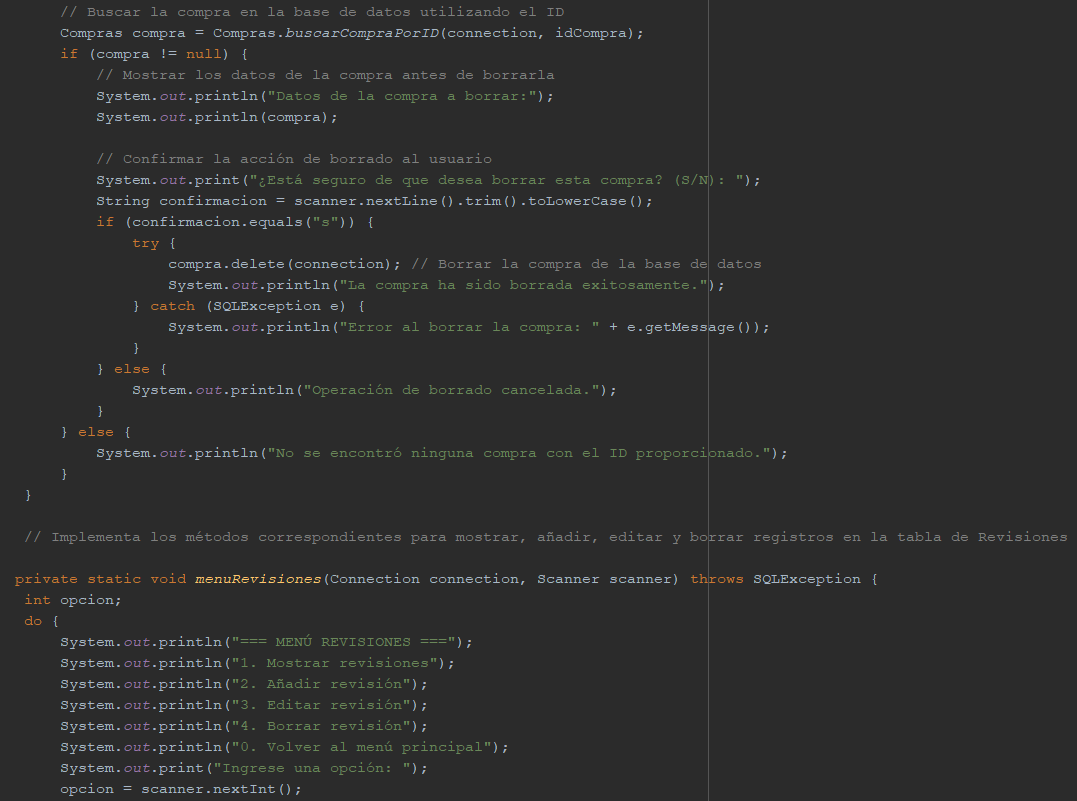


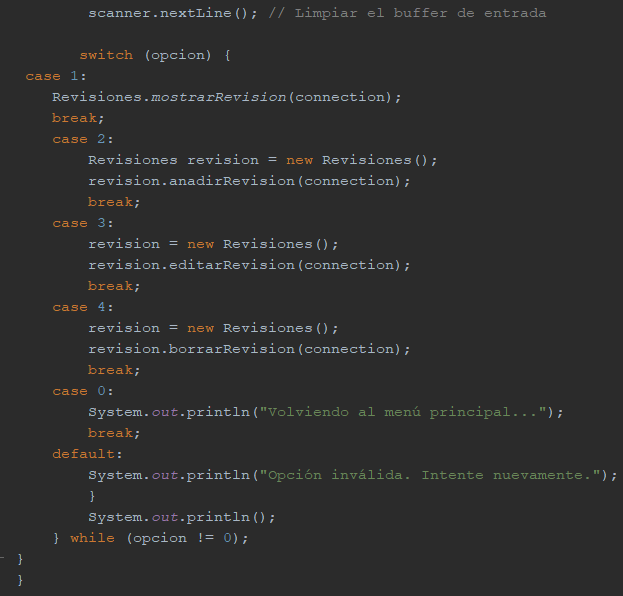




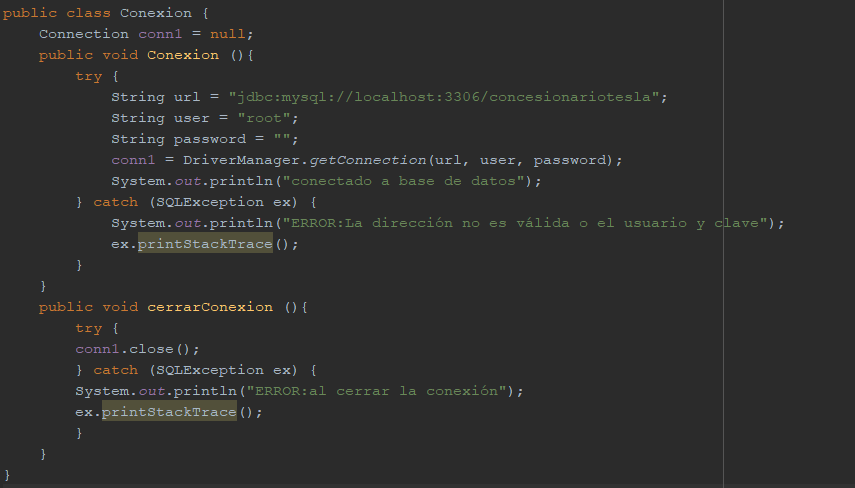








Para finalizar con el sistema ahora es cuando creamos la clase importante para poder realizar la conexión desde el programa a la base de datos que va a permitir su entrada en el. En este vamos a decirle a la base de datos que nos vamos a conectar como “root” y sin la contraseña. Junto a esto damos el enlace hacia nuestra base de datos de PHPMyAdmin. Debes en el mismo tener importado Connection, DriveManager y SQLException. A continuación se muestra el codigo en el que en tu caso debes poner tu base de datos:



Con todo esto ya esta listo el sistema del concesionario Tesla y al probar debe funcionarte el menú y dejarte mostrar listas, añadir según la tabla de la base de datos que estes, editar y borrar registros.

**ANEXO DE LA BASE DE DATOS**

La siguiente es una descripción detallada de la base de datos diseñada para la gestión de información de la empresa Tesla, que incluye las tablas, columnas y relaciones utilizadas.

**Tabla: Coches**

**Columnas:**

* Matricula (tipo: VARCHAR(10), clave primaria): La matrícula del coche.
* Marca (tipo: VARCHAR(50)): La marca del coche.
* Modelo (tipo: VARCHAR(50)): El modelo del coche.
* Color (tipo: VARCHAR(20)): El color del coche.
* PrecioVenta (tipo: DECIMAL(10, 2)): El precio de venta del coche.

**Tabla: Clientes**

**Columnas:**

* CodigoInterno (tipo: INT, clave primaria, autoincremental): El código interno único del cliente.
* NIF (tipo: VARCHAR(10)): El Número de Identificación Fiscal del cliente.
* Nombre (tipo: VARCHAR(100)): El nombre del cliente.
* Direccion (tipo: VARCHAR(200)): La dirección del cliente.
* Ciudad (tipo: VARCHAR(50)): La ciudad del cliente.
* NumeroTelefono (tipo: VARCHAR(20)): El número de teléfono del cliente.

**Tabla: Revisiones**

**Columnas:**

* IDRevision (tipo: INT, clave primaria, autoincremental): El identificador único de la revisión.
* MatriculaCoche (tipo: VARCHAR(10)): La matrícula del coche asociado a la revisión.
* CambioFiltro (tipo: BOOLEAN): Indica si se realizó un cambio de filtro durante la revisión.
* CambioAceite (tipo: BOOLEAN): Indica si se realizó un cambio de aceite durante la revisión.
* CambioFrenos (tipo: BOOLEAN): Indica si se realizó un cambio de frenos durante la revisión.
* Otros (tipo: VARCHAR(200)): Otros detalles o notas relacionados con la revisión.

**Tabla: Compras**

**Columnas:**

* IDCompra (tipo: INT, clave primaria, autoincremental): Identificador único de la compra.
* IDCompraCliente (tipo: INT): Identificador de la compra del cliente.
* CodigoInternoCliente (tipo: INT): Código interno del cliente que realizó la compra.
* MatriculaCoche (tipo: VARCHAR(10)): Matrícula del coche comprado.

**Relaciones:**

* La columna "IDCompraCliente" es una clave foránea que hace referencia a la columna "CodigoInterno" de la tabla "Clientes".
* La columna "MatriculaCoche" es una clave foránea que hace referencia a la columna "Matricula" de la tabla "Coches".

La base de datos diseñada permite almacenar y gestionar la información necesaria para el control de coches, clientes, revisiones y compras realizadas por los clientes en la empresa Tesla. Estas tablas y relaciones proporcionan la estructura necesaria para realizar consultas y operaciones relacionadas con la gestión de la información en el contexto de la venta de automóviles.

**ANEXO DEL PROGRAMA**

El siguiente es un anexo que explica las clases creadas en el programa de NetBeans para la gestión de información de la empresa Tesla, junto con la clase de conexión a la base de datos:

**Clase: Clientes**

Descripción: Esta clase representa a los clientes de la empresa Tesla. Contiene atributos para almacenar información como el código interno, NIF, nombre, dirección, ciudad y número de teléfono de cada cliente.

**Clase: Coches**

Descripción: Esta clase representa los coches disponibles para la venta en Tesla. Tiene atributos para almacenar información como la matrícula, marca, modelo, color y precio de venta de cada coche.

**Clase: Compra**

Descripción: Esta clase representa una compra realizada por un cliente. Contiene atributos para almacenar información como el identificador de compra, el código interno del cliente, la matrícula del coche comprado, entre otros.

**Clase: Revisiones**

Descripción: Esta clase representa las revisiones realizadas a los coches en Tesla. Tiene atributos como el identificador de revisión, la matrícula del coche, indicadores de cambio de filtro, cambio de aceite, cambio de frenos y otros detalles.

**Clase: Menu**

Descripción: Esta clase contiene el menú principal del programa. Proporciona opciones para realizar diferentes operaciones, como agregar clientes, agregar coches, registrar compras, registrar revisiones y realizar consultas.

**Clase: Conexion**

Descripción: Esta clase se encarga de establecer la conexión con la base de datos. Proporciona métodos para conectar, desconectar y ejecutar consultas en la base de datos.

La integración con la base de datos se realiza utilizando la clase "Conexion" para establecer la conexión con la base de datos, y luego se utilizan consultas SQL para realizar operaciones de inserción, actualización, eliminación y consulta en las tablas correspondientes.

La estructura del programa y la lógica de negocio dependerán de la implementación específica en NetBeans, utilizando el lenguaje de programación adecuado, como Java, y las herramientas y bibliotecas proporcionadas por NetBeans para la interacción con la base de datos.