

Al iniciar el proyecto se nos mostrará el menú de este

```
Conexion OK
***** Menú de Gestión de Parking *****
* 0. Registrar Cliente *
* 1. Dar de baja cliente *
* 2. Registrar Vehículo *
* 3. Dar de baja Vehículo *
* 4. Crear ticket *
* 5. Borrar ticket *
* 6. Modificar ticket *
* 7. Encontrar vehiculo por matrícula *
* 8. Consultar plaza *
* 9. Mostrar todos del garaje *
* 10. Salir *
*****
```

Métodos de conexión y cerrar conexión

```
1 public static Connection obtenerConexion() {
2     Connection conexion = null;
3
4     try {
5         conexion = DriverManager.getConnection("jdbc:mysql://localhost/parking", "root", "root");
6     } catch (SQLException var2) {
7         System.err.println("Error al conectar a la base de datos: " + var2.getMessage());
8     }
9
10    return conexion;
11 }
12
13 public static void cerrarConexion(Connection conexion) {
14     if (conexion != null) {
15         try {
16             conexion.close();
17         } catch (SQLException var2) {
18             System.err.println("Error al cerrar la conexi\u00f3n: " + var2.getMessage());
19         }
20     }
21 }
22 }
23
```

Se usa para poder conectarse a la base datos y cerrar la conexión con esta

Registrar cliente

```
1 public static void registrarCliente(Connection conexion, Scanner scanner) {
2     String sql = "INSERT INTO Cliente (Nombre, Apellido, Telefono, Email) VALUES (?, ?, ?, ?)";
3     scanner.nextLine();
4     System.out.println("Nombre: ");
5     String nombre = scanner.nextLine();
6     System.out.println("Apellido: ");
7     String apellido = scanner.nextLine();
8     System.out.println("Telefono: ");
9     String telefono = scanner.nextLine();
10    System.out.println("email: ");
11    String email = scanner.nextLine();
12
13    try {
14        PreparedStatement pstmt = conexion.prepareStatement(sql);
15        pstmt.setString(1, nombre);
16        pstmt.setString(2, apellido);
17        pstmt.setString(3, telefono);
18        pstmt.setString(4, email);
19        int filasAfectadas = pstmt.executeUpdate();
20        if (filasAfectadas > 0) {
21            System.out.println("Cliente registrado correctamente.");
22        } else {
23            System.out.println("No se pudo registrar el cliente.");
24        }
25    } catch (SQLException var9) {
26        System.err.println("Error al ejecutar la consulta SQL: " + var9.getMessage());
27    }
28
29 }
```

```
Ingrese su opción: 0
Nombre:
Pablo
Apellido:
Forero
Telefono:
123456789
email:
pablo@gmail.com
Cliente registrado correctamente.
```

Este método solicita datos del cliente, construye una consulta SQL para insertar un nuevo cliente en la base de datos y la ejecuta.

Dar de baja a cliente

```
1 public static void borrarCliente(Connection conexion, Scanner scanner) {
2     scanner.nextLine();
3     System.out.println("Nombre: ");
4     String nombre = scanner.nextLine();
5     System.out.println("Apellido: ");
6     String apellido = scanner.nextLine();
7     String sqlDeleteTickets = "DELETE FROM Ticket WHERE ClienteID IN (SELECT ClienteID FROM Cliente WHERE Nombre = ? AND Apellido = ?)";
8     String sqlDeleteCliente = "DELETE FROM Cliente WHERE Nombre = ? AND Apellido = ?";
9
10    try {
11        PreparedStatement pstmtDeleteTickets = conexion.prepareStatement(sqlDeleteTickets);
12        pstmtDeleteTickets.setString(1, nombre);
13        pstmtDeleteTickets.setString(2, apellido);
14        PreparedStatement pstmtDeleteCliente = conexion.prepareStatement(sqlDeleteCliente);
15        pstmtDeleteCliente.setString(1, nombre);
16        pstmtDeleteCliente.setString(2, apellido);
17        int filasAfectadasCliente = pstmtDeleteCliente.executeUpdate();
18        if (filasAfectadasCliente > 0) {
19            System.out.println("Cliente borrado correctamente.");
20        } else {
21            System.out.println("No se encontr\u00f3 ning\u00fan cliente con ese nombre y apellidos.");
22        }
23    } catch (SQLException var9) {
24        System.err.println("Error al ejecutar la consulta SQL: " + var9.getMessage());
25    }
26 }
27 }
```

```
Ingresa su opción: 1
Nombre:
Pablo
Apellido:
Forero
Cliente borrado correctamente.
```

Este método solicita datos del cliente y construye consultas SQL para eliminar al cliente y sus tickets asociados.

Registrar Vehículo

```
1 public static void registrarVehiculo(Connection conexion, Scanner scanner) {
2     scanner.nextLine();
3     System.out.println("\u00bfEs un coche o una moto? (coche/moto): ");
4     String tipoVehiculo = scanner.nextLine().toLowerCase();
5     System.out.println("Introduce la matr\u00edcula del veh\u00edculo: ");
6     String matricula = scanner.nextLine();
7     System.out.println("Introduce la marca del veh\u00edculo: ");
8     String marca = scanner.nextLine();
9     System.out.println("Introduce el modelo del veh\u00edculo: ");
10    String modelo = scanner.nextLine();
11    System.out.println("Introduce el color del veh\u00edculo: ");
12    String color = scanner.nextLine();
13    String tipoEspecifico = "";
14    if (tipoVehiculo.equals("coche")) {
15        System.out.println("Introduce el tipo de coche (ej. Sed\u00e1n, SUV, etc.): ");
16        tipoEspecifico = scanner.nextLine();
17    } else {
18        if (tipoVehiculo.equals("moto")) {
19            System.out.println("Tipo de veh\u00edculo no v\u00eddido. Debe ser 'coche' o 'moto'.");
20            return;
21        }
22        System.out.println("Introduce el tipo de moto (ej. Deportiva, Scooter, etc.): ");
23        tipoEspecifico = scanner.nextLine();
24    }
25    String sqlVehiculo = "INSERT INTO Vehiculo (Matricula, Marca, Modelo, Color) VALUES (?, ?, ?, ?)";
26    try {
27        PreparedStatement pstmtVehiculo = conexion.prepareStatement(sqlVehiculo);
28        pstmtVehiculo.setString(1, matricula);
29        pstmtVehiculo.setString(2, marca);
30        pstmtVehiculo.setString(3, modelo);
31        pstmtVehiculo.setString(4, color);
32        int filasAfectadasVehiculo = pstmtVehiculo.executeUpdate();
33        if (filasAfectadasVehiculo > 0) {
34            System.out.println("Veh\u00edculo registrado correctamente en la tabla Vehiculo.");
35            String sqlTipoEspecifico = "";
36            if (tipoVehiculo.equals("coche")) {
37                sqlTipoEspecifico = "INSERT INTO Coche (Matricula, TipoCoche) VALUES (?, ?)";
38            } else if (tipoVehiculo.equals("moto")) {
39                sqlTipoEspecifico = "INSERT INTO Moto (Matricula, TipoMoto) VALUES (?, ?)";
40            }
41            PreparedStatement pstmtTipoEspecifico = conexion.prepareStatement(sqlTipoEspecifico);
42            pstmtTipoEspecifico.setString(1, matricula);
43            pstmtTipoEspecifico.setString(2, tipoEspecifico);
44            int filasAfectadasTipoEspecifico = pstmtTipoEspecifico.executeUpdate();
45            if (filasAfectadasTipoEspecifico > 0) {
46                System.out.println("Veh\u00edculo registrado correctamente en la tabla " + (tipoVehiculo.equals("coche") ? "Coche" : "Moto") + ".");
47            } else {
48                System.out.println("No se pudo registrar el tipo espec\u00edfico del veh\u00edculo en la base de datos.");
49            }
50        } else {
51            System.out.println("No se pudo registrar el veh\u00edculo en la tabla Vehiculo.");
52        }
53    } catch (SQLException var4) {
54        System.err.println("Error al ejecutar la consulta SQL: " + var4.getMessage());
55    }
56 }
```

```
Ingrese su opción: 2
¿Es un coche o una moto? (coche/moto):
coche
Introduce la matrícula del vehículo:
1111AAA
Introduce la marca del vehículo:
Ford
Introduce el modelo del vehículo:
Focus
Introduce el color del vehículo:
Verde
Introduce el tipo de coche (ej. Sedán, SUV, etc.):
SUV
Vehículo registrado correctamente en la tabla Vehiculo.
Vehículo registrado correctamente en la tabla Coche.
```

Este método solicita datos del vehículo y realiza inserciones en las tablas Vehiculo, Coche, o Moto, dependiendo del tipo de vehículo.

Dar de baja Vehículo

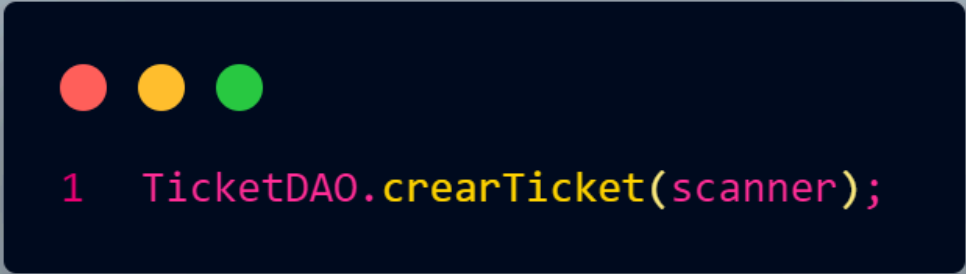
```
1 public static void darBajaVehiculo(Connection conexion, Scanner scanner) {
2     scanner.nextLine();
3     System.out.println("\u00bfEs un coche o una moto? (coche/moto): ");
4     String tipoVehiculo = scanner.nextLine().toLowerCase();
5     System.out.println("Introduce la matr\u00edcula del veh\u00edculo: ");
6     String matricula = scanner.nextLine();
7     String sqlEliminarVehiculo = "DELETE FROM Vehiculo WHERE Matricula = ?";
8     String sqlEliminarEspecifico = "";
9     if (tipoVehiculo.equals("coche")) {
10         sqlEliminarEspecifico = "DELETE FROM Coche WHERE Matricula = ?";
11     } else {
12         if (!tipoVehiculo.equals("moto")) {
13             System.out.println("Tipo de veh\u00edculo no v\u00e1lido. Debe ser 'coche' o 'moto'.");
14             return;
15         }
16         sqlEliminarEspecifico = "DELETE FROM Moto WHERE Matricula = ?";
17     }
18 }
19
20 try {
21     PreparedStatement pstmtEspecifico = conexion.prepareStatement(sqlEliminarEspecifico);
22     pstmtEspecifico.setString(1, matricula);
23     int filasAfectadasEspecifico = pstmtEspecifico.executeUpdate();
24     if (filasAfectadasEspecifico > 0) {
25         PrintStream var10000 = System.out;
26         String var10001 = tipoVehiculo.equals("coche") ? "Coche" : "Moto";
27         var10000.println(var10001 + " eliminado correctamente de la tabla " + (tipoVehiculo.equals("coche") ? "Coche" : "Moto") + ".");
28         PreparedStatement pstmtVehiculo = conexion.prepareStatement(sqlEliminarVehiculo);
29         pstmtVehiculo.setString(1, matricula);
30         int filasAfectadasVehiculo = pstmtVehiculo.executeUpdate();
31         if (filasAfectadasVehiculo > 0) {
32             System.out.println("Veh\u00edculo eliminado correctamente de la tabla Vehiculo.");
33         } else {
34             System.out.println("No se pudo eliminar el veh\u00edculo de la tabla Vehiculo.");
35         }
36     } else {
37         System.out.println("No se encontr\u00f3 el veh\u00edculo en la tabla " + (tipoVehiculo.equals("coche") ? "Coche" : "Moto") + " o ya ha sido eliminado.");
38     }
39 } catch (SQLException var10) {
40     System.err.println("Error al ejecutar la consulta SQL: " + var10.getMessage());
41 }
42
43 }
```

```
Ingrese su opción: 3
¿Es un coche o una moto? (coche/moto):
coche
Introduce la matrícula del vehículo:
1111AAA
Coche eliminado correctamente de la tabla Coche.
Vehículo eliminado correctamente de la tabla Vehiculo.
```

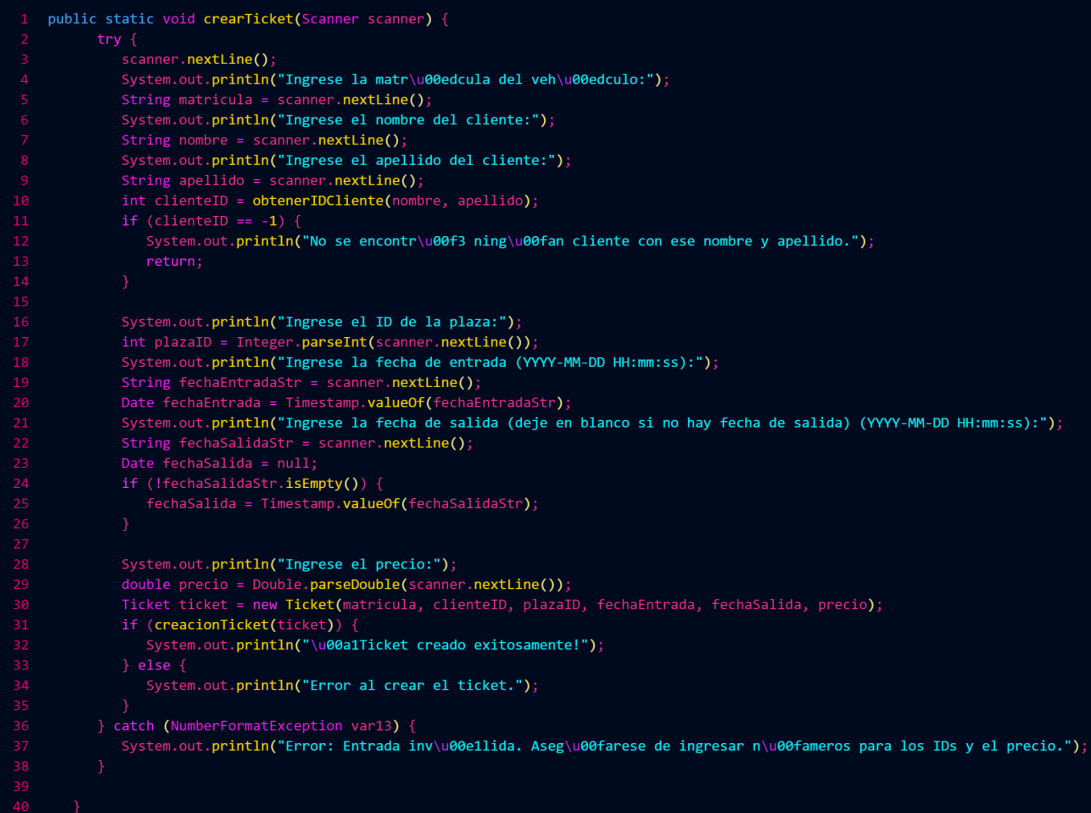
Este método elimina registros de un vehículo específico de las tablas Vehículo, Coche, o Moto.

Crear ticket

En el switch



```
1 TicketDAO.crearTicket(scanner);
```



```
1 public static void crearTicket(Scanner scanner) {
2     try {
3         scanner.nextLine();
4         System.out.println("Ingrese la matr\u00e9dcula del veh\u00e9dculo:");
5         String matricula = scanner.nextLine();
6         System.out.println("Ingrese el nombre del cliente:");
7         String nombre = scanner.nextLine();
8         System.out.println("Ingrese el apellido del cliente:");
9         String apellido = scanner.nextLine();
10        int clienteID = obtenerIDCliente(nombre, apellido);
11        if (clienteID == -1) {
12            System.out.println("No se encontr\u00f3 ning\u00fan cliente con ese nombre y apellido.");
13            return;
14        }
15
16        System.out.println("Ingrese el ID de la plaza:");
17        int plazaID = Integer.parseInt(scanner.nextLine());
18        System.out.println("Ingrese la fecha de entrada (YYYY-MM-DD HH:mm:ss:");
19        String fechaEntradaStr = scanner.nextLine();
20        Date fechaEntrada = Timestamp.valueOf(fechaEntradaStr);
21        System.out.println("Ingrese la fecha de salida (deje en blanco si no hay fecha de salida) (YYYY-MM-DD HH:mm:ss:");
22        String fechaSalidaStr = scanner.nextLine();
23        Date fechaSalida = null;
24        if (!fechaSalidaStr.isEmpty()) {
25            fechaSalida = Timestamp.valueOf(fechaSalidaStr);
26        }
27
28        System.out.println("Ingrese el precio:");
29        double precio = Double.parseDouble(scanner.nextLine());
30        Ticket ticket = new Ticket(matricula, clienteID, plazaID, fechaEntrada, fechaSalida, precio);
31        if (creacionTicket(ticket)) {
32            System.out.println("\u00a1Ticket creado exitosamente!");
33        } else {
34            System.out.println("Error al crear el ticket.");
35        }
36    } catch (NumberFormatException var13) {
37        System.out.println("Error: Entrada inv\u00e9lida. Aseg\u00farese de ingresar n\u00fameros para los IDs y el precio.");
38    }
39 }
40 }
```

```

Ingrese su opción: 4
Ingrese la matrícula del vehículo:
ABC1234
Ingrese el nombre del cliente:
Elena
Ingrese el apellido del cliente:
Torres
Ingrese el ID de la plaza:
1
Ingrese la fecha de entrada (YYYY-MM-DD HH:mm:ss):
2024-05-26 17:30:00
Ingrese la fecha de salida (deje en blanco si no hay fecha de salida) (YYYY-MM-DD HH:mm:ss):

Ingrese el precio:
50

```

Este método guía al usuario para ingresar detalles del ticket a través de la consola y luego inserta un nuevo registro en la base de datos.

-Pasos detallados:

Pide al usuario que ingrese varios detalles (matrícula, nombre, apellido, ID de plaza, fecha de entrada, fecha de salida y precio).

Busca el ID del cliente usando obtenerIDCliente.

Crea un objeto Ticket con los datos recopilados.

Llama al método creacionTicket para insertar el ticket en la base de datos.

Maneja excepciones para entradas inválidas.

método que usa llamado creacionTicket

```

1 public static boolean creacionTicket(Ticket ticket) {
2     String sql = "INSERT INTO Ticket (Matricula, ClienteID, PlazaID, FechaEntrada, FechaSalida, Precio) VALUES (?, ?, ?, ?, ?, ?)";
3
4     try {
5         Throwable var2 = null;
6         Object var3 = null;
7
8         try {
9             Connection conexion = DriverManager.getConnection("jdbc:mysql://localhost/parking", "root", "root");
10
11             boolean var10000;
12             try {
13                 PreparedStatement pstmt = conexion.prepareStatement(sql);
14
15                 try {
16                     pstmt.setString(1, ticket.getMatricula());
17                     pstmt.setInt(2, ticket.getClienteID());
18                     pstmt.setInt(3, ticket.getPlazaID());
19                     pstmt.setTimestamp(4, new Timestamp(ticket.getFechaEntrada().getTime()));
20                     pstmt.setTimestamp(5, ticket.getFechaSalida() != null ? new Timestamp(ticket.getFechaSalida().getTime()) : null);
21                     pstmt.setDouble(6, ticket.getPrecio());
22                     int filasAfectadas = pstmt.executeUpdate();
23                     var10000 = filasAfectadas > 0;
24                 } finally {
25                     if (pstmt != null) {
26                         pstmt.close();
27                     }
28                 }
29             } catch (Throwable var20) {
30                 if (var2 == null) {
31                     var2 = var20;
32                 } else if (var2 != var20) {
33                     var2.addSuppressed(var20);
34                 }
35             }
36
37             if (conexion != null) {
38                 conexion.close();
39             }
40
41             throw var2;
42         }
43
44         if (conexion != null) {
45             conexion.close();
46         }
47
48         return var10000;
49     } catch (Throwable var21) {
50         if (var2 == null) {
51             var2 = var21;
52         } else if (var2 != var21) {
53             var2.addSuppressed(var21);
54         }
55
56         throw var2;
57     }
58 } catch (SQLException var22) {
59     System.err.println("Error al crear el ticket: " + var22.getMessage());
60     return false;
61 }
62 }

```

Borrar ticket

```
1 public static void borrarTicket(Connection conexion, Scanner scanner) {
2     System.out.print("Ingrese la ID del ticket que desea borrar: ");
3     int ticketID = scanner.nextInt();
4     String sql = "DELETE FROM Ticket WHERE TicketID = ?";
5
6     try {
7         PreparedStatement pstmt = conexion.prepareStatement(sql);
8         pstmt.setInt(1, ticketID);
9         int filasAfectadas = pstmt.executeUpdate();
10        if (filasAfectadas > 0) {
11            System.out.println("Ticket borrado correctamente.");
12        } else {
13            System.out.println("No se encontr\u00f3 ning\u00fan ticket con el ID proporcionado.");
14        }
15    } catch (SQLException var6) {
16        System.err.println("Error al borrar el ticket: " + var6.getMessage());
17    }
18 }
19 }
```

```
Ingrese su opción: 5
Ingrese la ID del ticket que desea borrar: 11
Ticket borrado correctamente.
```

Elimina un ticket de la base de datos usando su ID.

Pasos detallados:

Pide al usuario que ingrese el ID del ticket a eliminar.

Prepara y ejecuta una instrucción SQL DELETE.

Informa al usuario si el ticket fue eliminado exitosamente o no se encontró.

Maneja excepciones SQL.

Modificar ticket

```
1 public static void modificarTicket(Connection conexion, Scanner scanner) {
2     System.out.print("Ingrese la ID del ticket que desea modificar: ");
3     int ticketID = scanner.nextInt();
4     System.out.print("Ingrese la nueva fecha de salida (formato: yyyy-MM-dd HH:mm:ss: ");
5     String nuevaFechaSalida = scanner.next();
6     String sql = "UPDATE Ticket SET FechaSalida = ? WHERE TicketID = ?";
7
8     try {
9         PreparedStatement pstmt = conexion.prepareStatement(sql);
10        pstmt.setString(1, nuevaFechaSalida);
11        pstmt.setInt(2, ticketID);
12        int filasAfectadas = pstmt.executeUpdate();
13        if (filasAfectadas > 0) {
14            System.out.println("Ticket modificado correctamente.");
15        } else {
16            System.out.println("No se encontr\u00f3 ning\u00fan ticket con el ID proporcionado.");
17        }
18    } catch (SQLException var7) {
19        System.err.println("Error al modificar el ticket: " + var7.getMessage());
20    }
21 }
22 }
```



```
Ingrese su opción: 6
Ingrese la ID del ticket que desea modificar: 1
Ingrese la nueva fecha de salida (formato: yyyy-MM-dd HH:mm:ss): 2024-05-20 20:00:00
Ticket modificado correctamente.
```

Actualiza la fecha de salida de un ticket existente en la base de datos.

Pasos detallados:

Pide al usuario que ingrese el ID del ticket y la nueva fecha de salida.

Prepara y ejecuta una instrucción SQL UPDATE.

Informa al usuario si el ticket fue modificado exitosamente o no se encontró.

Maneja excepciones SQL.

Encontrar vehiculo por matrícula

```
1 public static void buscarVehiculoMatricula(Connection conexion, Scanner scanner) {
2     scanner.nextLine();
3     System.out.println("Introduce la matrícula del vehículo: ");
4     String matricula = scanner.nextLine();
5     String sql = "SELECT p.ubicacion, t.FechaSalida FROM Plaza p JOIN Ticket t ON p.PlazaID = t.PlazaID WHERE t.Matricula = ?";
6
7     try {
8         Throwable var4 = null;
9         Object var5 = null;
10
11         try {
12             PreparedStatement pstmt = conexion.prepareStatement(sql);
13
14             try {
15                 pstmt.setString(1, matricula);
16                 Throwable var7 = null;
17                 Object var8 = null;
18
19                 try {
20                     ResultSet rs = pstmt.executeQuery();
21
22                     try {
23                         if (rs.next()) {
24                             Timestamp fechaSalidaTimestamp = rs.getTimestamp("FechaSalida");
25                             if (fechaSalidaTimestamp == null) {
26                                 String ubicacion = rs.getString("ubicacion");
27                                 System.out.println("El vehículo con matrícula " + matricula + " se encuentra en la plaza " + ubicacion);
28                             } else {
29                                 System.out.println("El vehículo con matrícula " + matricula + " no está en el garaje.");
30                             }
31                         } else {
32                             System.out.println("No se encontró ning\u00fan veh\u00edculo con esa matr\u00edcula en el parking.");
33                         }
34                     } finally {
35                         if (rs != null) {
36                             rs.close();
37                         }
38                     }
39                 } catch (Throwable var33) {
40                     if (var7 == null) {
41                         var7 = var33;
42                     } else if (var7 != var33) {
43                         var7.addSuppressed(var33);
44                     }
45                 }
46                 throw var7;
47             } finally {
48                 if (pstmt != null) {
49                     pstmt.close();
50                 }
51             }
52         } catch (Throwable var35) {
53             if (var4 == null) {
54                 var4 = var35;
55             } else if (var4 != var35) {
56                 var4.addSuppressed(var35);
57             }
58             throw var4;
59         } catch (SQLException var36) {
60             System.out.println("Error al buscar el veh\u00edculo: " + var36.getMessage());
61         }
62     }
63 }
```

```
Ingrese su opción: 7
Introduce la matrícula del vehículo:
JKL3456
El vehículo con matrícula JKL3456 se encuentra en la plaza Nivel 2 - B2
```

Este método busca la ubicación de un vehículo en el garaje según su matrícula.

Consultar plaza

```
1 public static void consultarPlaza(Connection conexion, Scanner scanner) {
2     System.out.print("Ingrese el ID de la plaza a consultar: ");
3     int idPlaza = scanner.nextInt();
4     String sql = "SELECT p.PlazaID, p.Ubicacion, p.Tipo, t.ClienteID, c.Nombre, c.Apellido FROM Plaza p LEFT JOIN Ticket t ON p.PlazaID = t.PlazaID LEFT JOIN Cliente c ON t.ClienteID = c.ClienteID WHERE p.PlazaID = ?";
5
6     try {
7         Throwable var4 = null;
8         Object var5 = null;
9
10        try {
11            PreparedStatement pstmt = conexion.prepareStatement(sql);
12
13            try {
14                pstmt.setInt(1, idPlaza);
15                Throwable var7 = null;
16                Object var8 = null;
17
18                try {
19                    ResultSet rs = pstmt.executeQuery();
20
21                    try {
22                        if (rs.next()) {
23                            int plazaID = rs.getInt("PlazaID");
24                            String ubicacion = rs.getString("Ubicacion");
25                            String tipo = rs.getString("Tipo");
26                            int clienteID = rs.getInt("ClienteID");
27                            String nombreCliente = rs.getString("Nombre");
28                            String apellidoCliente = rs.getString("Apellido");
29                            System.out.println("Informaci\u00f3n de la plaza:");
30                            System.out.println("ID de la plaza: " + plazaID);
31                            System.out.println("Ubicaci\u00f3n: " + ubicacion);
32                            System.out.println("Tipo: " + tipo);
33                            System.out.println();
34                            if (clienteID != 0) {
35                                System.out.println("Informaci\u00f3n del cliente que
36                                e hizo el ticket:");
37                                System.out.println("ID del cliente: " + clienteID);
38                                System.out.println("Nombre: " + nombreCliente + " "
39                                + apellidoCliente);
40                            } else {
41                                System.out.println("No hay informaci\u00f3n sobre el
42                                cliente que hizo el ticket.");
43                            }
44                            } else {
45                                System.out.println("No se encontr\u00f3 ninguna plaza
46                                con el ID proporcionado.");
47                            }
48                        } finally {
49                            if (rs != null) {
50                                rs.close();
51                            }
52                        }
53                    } catch (Throwable var37) {
54                        if (var7 == null) {
55                            var7 = var37;
56                        } else if (var7 != var37) {
57                            var7.addSuppressed(var37);
58                        }
59                    }
60                    throw var7;
61                } finally {
62                    if (pstmt != null) {
63                        pstmt.close();
64                    }
65                }
66            } catch (Throwable var39) {
67                if (var4 == null) {
68                    var4 = var39;
69                } else if (var4 != var39) {
70                    var4.addSuppressed(var39);
71                }
72            }
73            throw var4;
74        } catch (SQLException var40) {
75            System.err.println("Error al consultar la plaza: " + var40.getMessage());
76        }
77    }
78 }
```

```

Ingrese su opción: 8
Ingrese el ID de la plaza a consultar: 2
Información de la plaza:
ID de la plaza: 2
Ubicación: Nivel 1 - A2
Tipo: Discapacitado

Información del cliente que hizo el ticket:
ID del cliente: 2
Nombre: María González

```

Este método solicita el ID de una plaza y muestra información relacionada, incluyendo detalles del cliente si hay un ticket asociado.

Mostrar todos del garaje

```

1 public static void mostrarVehiculosGaraje(Connection conexion) {
2     String sql = "SELECT p.Ubicacion, v.Modelo, t.Matricula, c.Nombre, c.Apellido FROM Plaza p JOIN Ticket t ON p.PlazaID = t.PlazaID JOIN Vehiculo v ON t.Matricula = v.Matricula JOIN Cliente c ON t.ClienteID = c.ClienteID WHERE t.FechaSalida IS NULL";
3
4     try {
5         Throwable var2 = null;
6         Object var3 = null;
7
8         try {
9             Statement stmt = conexion.createStatement();
10
11             try {
12                 ResultSet rs = stmt.executeQuery(sql);
13
14                 try {
15                     System.out.println("Veh\u00e9culos en el garaje:");
16                     System.out.println("-----");
17                     System.out.printf("%-15s %-20s %-10s %-15s %-15s\n", "Ubicación", "Modelo", "Matr\u00edcula", "Nombre Cliente", "Apellido Cliente");
18                     System.out.println("-----");
19
20                     while(true) {
21                         if (!rs.next()) {
22                             System.out.println("-----");
23                             break;
24                         }
25
26                         String ubicacion = rs.getString("Ubicación");
27                         String modelo = rs.getString("Modelo");
28                         String matricula = rs.getString("Matricula");
29                         String nombreCliente = rs.getString("Nombre");
30                         String apellidoCliente = rs.getString("Apellido");
31                         System.out.printf("%-15s %-20s %-10s %-15s %-15s\n", ubicacion, modelo, matricula, nombreCliente, apellidoCliente);
32                     } finally {
33                         if (rs != null) {
34                             rs.close();
35                         }
36                     }
37                 } catch (Throwable var24) {
38                     if (var2 == null) {
39                         var2 = var24;
40                     } else if (var2 != var24) {
41                         var2.addSuppressed(var24);
42                     }
43                 }
44
45                 if (stmt != null) {
46                     stmt.close();
47                 }
48
49                 throw var2;
50             }
51         } catch (Throwable var25) {
52             if (var2 == null) {
53                 var2 = var25;
54             } else if (var2 != var25) {
55                 var2.addSuppressed(var25);
56             }
57             throw var2;
58         } catch (SQLException var26) {
59             System.err.println("Error al listar los veh\u00e9culos en el garaje: " + var26.getMessage());
60         }
61     }
62 }

```

```
Ingrese su opción: 9
Vehiculos en el garaje:
```

Ubicación	Plaza	Modelo	Matrícula	Nombre Cliente	Apellido Cliente
Nivel 1 - A2		Civic	DEF5678	María	González
Nivel 2 - B2		R1	GHI9012	Carlos	López
Nivel 2 - B2		GSX-R750	JKL3456	Ana	Martínez

Este método muestra una lista de todos los vehículos que actualmente se encuentran en el garaje.