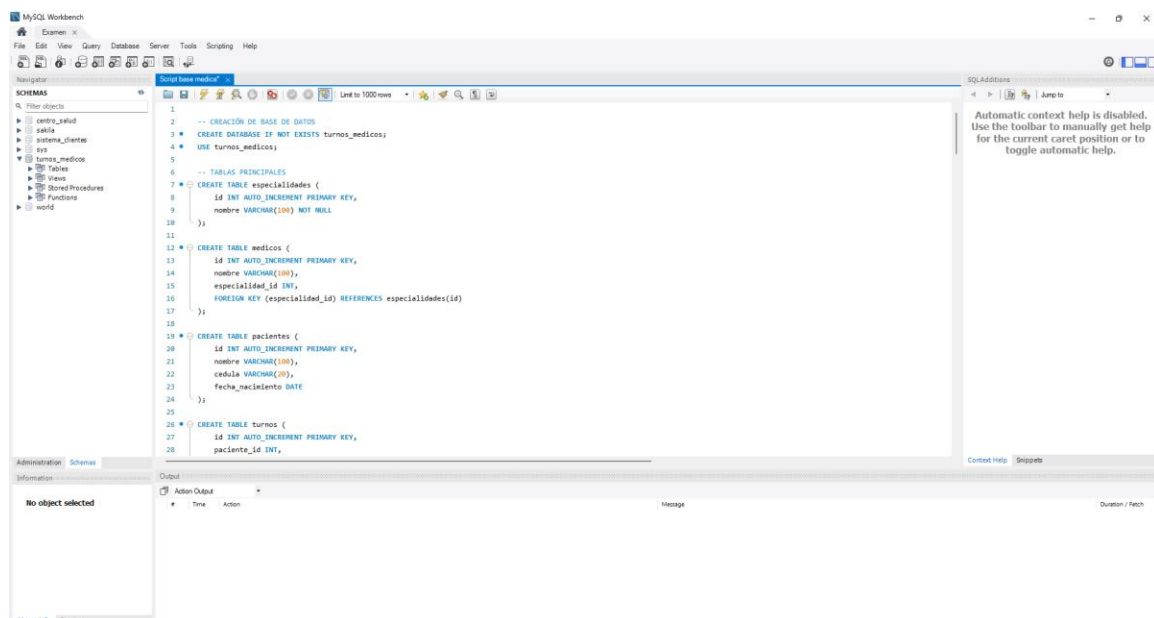


# TALLER Sistema con Base de Datos y Java

Antony Cisneros

## Parte 1: Configuración de la base de datos

1. Descargar el script SQL proporcionado.



2. Crear la base de datos y las tablas ejecutando el script.

3. Insertar los registros necesarios utilizando INSERT INTO.

4. Crear únicamente las funciones definidas en el script.

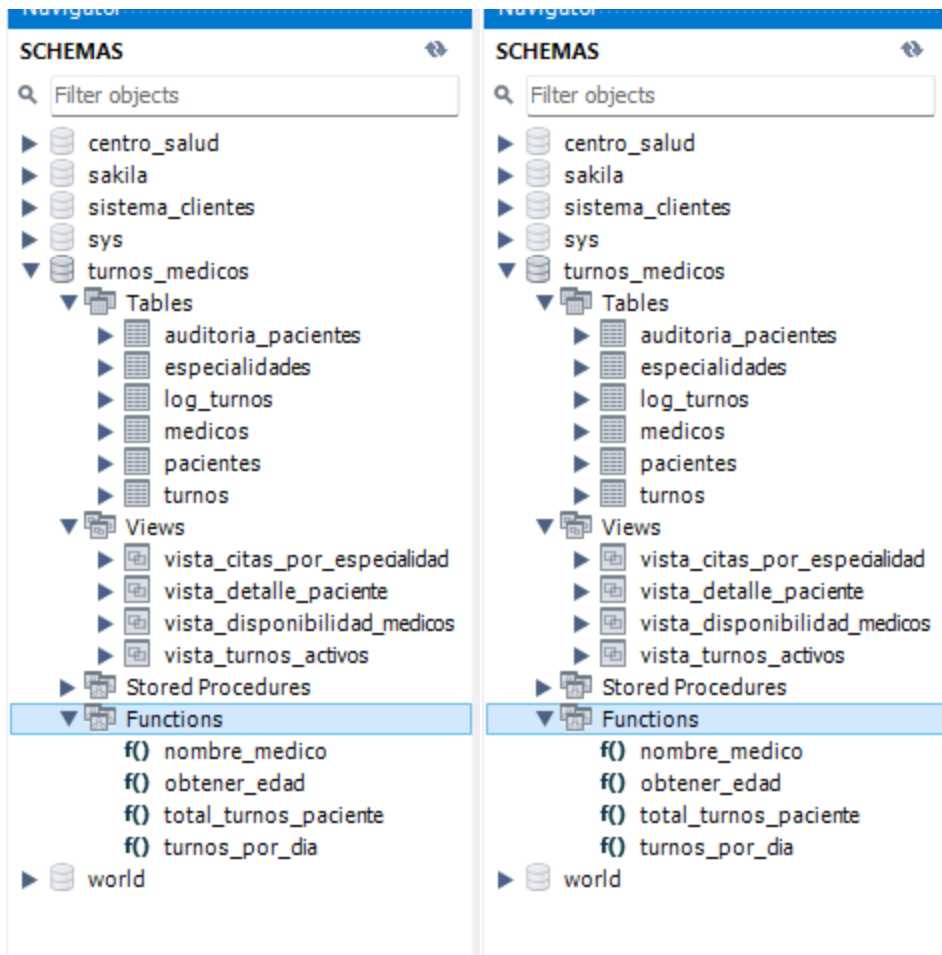
5. Crear únicamente los procedimientos almacenados.

6. Crear únicamente los triggers.

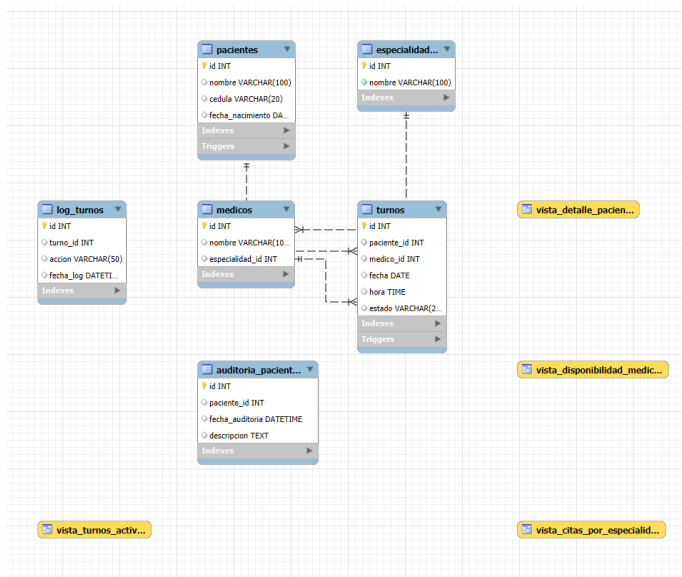
6.1 Verificar que los triggers fueron creados correctamente usando la instrucción:

SHOW TRIGGERS;

7. Crear las vistas definidas.



8. Verificar que todos los elementos anteriores (tablas, funciones, procedimientos, triggers y vistas) fueron creados correctamente.



## Parte 2: Configuración del proyecto Java en IntelliJ IDEA

1. Abrir IntelliJ IDEA y cargar el proyecto.
2. Restaurar el código Java del sistema.
3. Ir a File → Project Structure → Libraries y añadir el conector MySQL (mysql-connector-java-x.x.xx.jar) previamente descargado.
4. Si aún no lo tienes, descargar el conector MySQL desde el sitio oficial:  
<https://dev.mysql.com/downloads/connector/j/>
5. Ejecutar el código Java y comprobar que la conexión con la base de datos funcione.
6. Verificar que el código Java utilice correctamente los elementos de base de datos:
  - Vistas
  - Funciones
  - Procedimientos almacenados
  - Triggers
7. Crear usuarios para probar las acciones del trigger

### **Capturar pantallas.**

Subir información a git HUB y poner conclusión de la práctica realizada

*REGISTRO DE PACIENTES*

Sistema de Turnos Médicos

Nombre: Antony Cisneros

Cédula: 17525757342

Fecha Nac (YYYY-MM-DD): 2005-08-19

Registrar Paciente

Ver Turnos Activos

Paciente registrado correctamente.

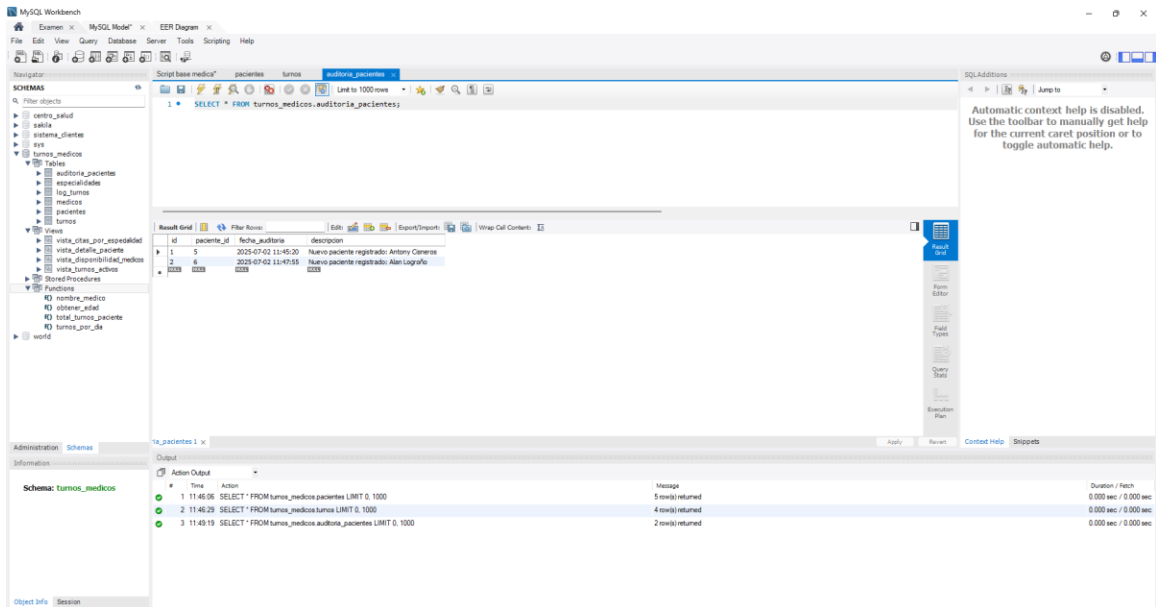
## INGRESO EN LA BASE DE DATOS MYSQL

The screenshot shows the MySQL Workbench interface. The 'Schemas' pane on the left lists the database structure, including tables like 'pacientes' and 'turnos'. The 'Script base medicos' pane shows a query: `SELECT * FROM turnos_medicos.pacientes`. The 'Result Grid' displays the following data:

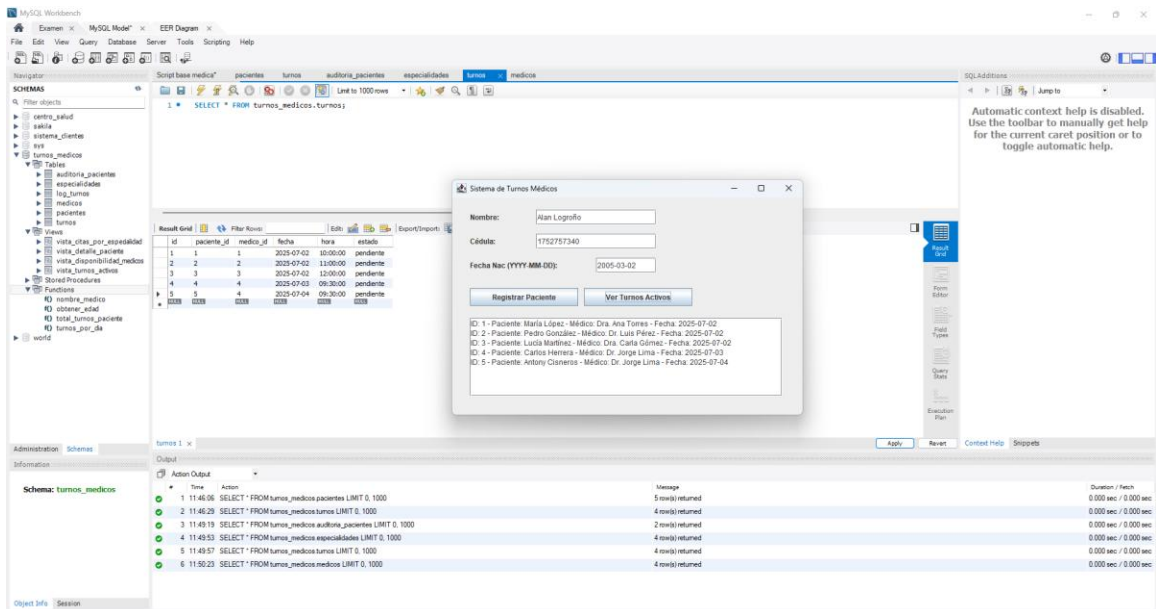
id	nombre	cedula	fecha_registro
1	Maria Lopez	100223940	1980-04-15
2	Pedro Gonzalez	100294036	1985-06-20
3	Luisa Martinez	100405867	2002-09-10
4	Carlos Herrera	100566778	1978-12-05
5	Antony Cisneros	17525757342	2005-08-19
6	Alan Lagrifa	17525757340	2005-03-02

The 'Action Output' pane at the bottom shows the execution of the query, indicating that 5 rows were returned and the operation was successful.

## TRIGGER DE INGRESO DE PACIENTES



## AGENDAMIENTO DEL TURNO EN LA TABLA TURNOS Y SU ACTUALIZACIÓN EN EL SISTEMA



## VISTAS

MySQL Workbench

Examenes x MySQL Model x EER Diagram x

File Edit View Query Database Server Tools Scripting Help

Navigator

SCHEMAS

Filter objects

- centro\_salud
- sakila
- sistema\_clientes
- sys
- turnos\_medicos
  - Tables
    - auditoria\_pacientes
    - especialidades
    - log\_turnos
    - medicos
    - pacientes
    - turnos
  - Views
    - vista\_citas\_por\_especialidad
    - vista\_detalle\_paciente
    - vista\_disponibilidad\_medicos
    - vista\_turnos\_activos
  - Stored Procedures
  - Functions
    - nombre\_medico
    - obtener\_edad
    - total\_turnos\_paciente
    - turnos\_por\_da
  - world

Script base medica\* pacientes turnos auditoria\_pacientes especialidades turnos medicos vista\_turnos\_activos x

Limit to 1000 rows

1 • SELECT \* FROM turnos\_medicos.vista\_turnos\_activos;

Result Grid

	id	paciente	medico	fecha	hora	estado
1	1	María López	Dra. Ana Torres	2025-07-02	10:00:00	pendiente
2	2	Pedro González	Dr. Luis Pérez	2025-07-02	11:00:00	pendiente
3	3	Lucía Martínez	Dra. Carla Gómez	2025-07-02	12:00:00	pendiente
4	4	Carlos Herrera	Dr. Jorge Lima	2025-07-03	09:30:00	pendiente
5	5	Antony Cisneros	Dr. Jorge Lima	2025-07-04	09:30:00	pendiente

## PROCEDIMIENTOS ALMACENADOS

MySQL Workbench

Examenes x MySQL Model x EER Diagram x

File Edit View Query Database Server Tools Scripting Help

Navigator

SCHEMAS

Filter objects

- centro\_salud
- sakila
- sistema\_clientes
- sys
- turnos\_medicos
  - Tables
    - auditoria\_pacientes
    - especialidades
    - log\_turnos
    - medicos
    - pacientes
    - turnos
  - Views
    - vista\_citas\_por\_especialidad
    - vista\_detalle\_paciente
    - vista\_disponibilidad\_medicos
    - vista\_turnos\_activos
  - Stored Procedures
    - actualizar\_estado\_turno
    - cancelar\_turno
    - crear\_turno
    - registrar\_paciente
  - Functions
    - nombre\_medico
    - obtener\_edad
    - total\_turnos\_paciente
    - turnos\_por\_da
  - world

Script base medica\* pacientes turnos auditoria\_pacientes especialidades turnos medicos vista\_turnos\_activos x

Limit to 1000 rows

1 • SELECT \* FROM turnos\_medicos.vista\_turnos\_activos;

Result Grid

	id	paciente	medico	fecha	hora	estado
1	1	María López	Dra. Ana Torres	2025-07-02	10:00:00	pendiente
2	2	Pedro González	Dr. Luis Pérez	2025-07-02	11:00:00	pendiente
3	3	Lucía Martínez	Dra. Carla Gómez	2025-07-02	12:00:00	pendiente
4	4	Carlos Herrera	Dr. Jorge Lima	2025-07-03	09:30:00	pendiente
5	5	Antony Cisneros	Dr. Jorge Lima	2025-07-04	09:30:00	pendiente

Call stored procedure turnos\_medicos.crear\_turno

Enter values for parameters of your procedure and click <Obtuse> to create an SQL editor and run the call:

p\_id [IN] INT

m\_id [IN] INT

f [IN] DATE

h [IN] TIME

Execute Cancel

Information

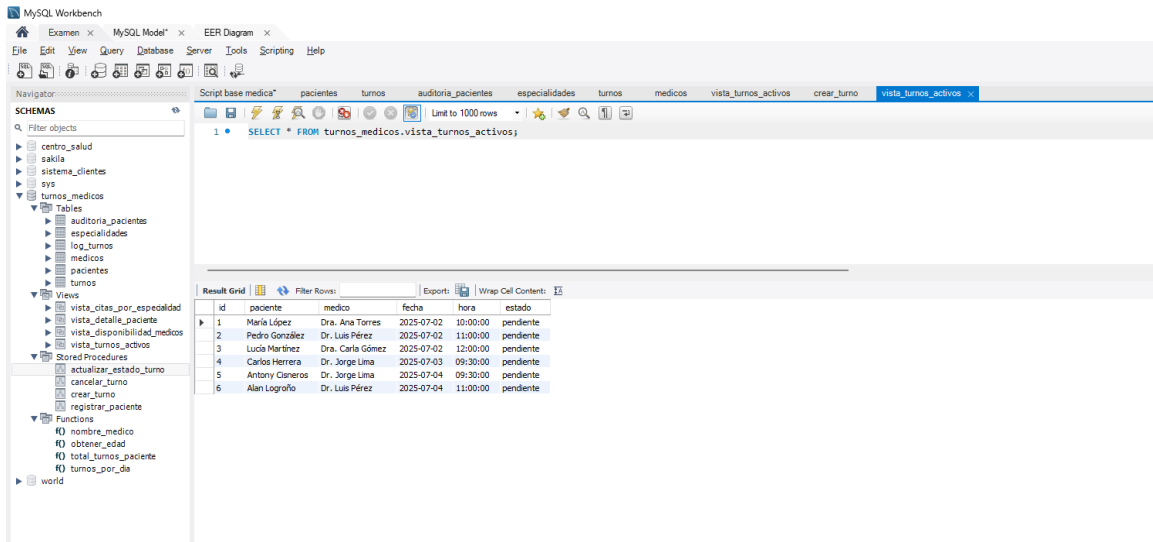
Unable to retrieve node description.

Object Info Session

Output

#	Time	Action	Message	Duration / Fetch
1	11:46:06	SELECT * FROM turnos_medicos.pacientes LIMIT 0, 1000	5 row(s) returned	0.000 sec / 0.000 sec
2	11:46:29	SELECT * FROM turnos_medicos.turnos LIMIT 0, 1000	4 row(s) returned	0.000 sec / 0.000 sec
3	11:49:19	SELECT * FROM turnos_medicos.auditoria_pacientes LIMIT 0, 1000	2 row(s) returned	0.000 sec / 0.000 sec
4	11:49:53	SELECT * FROM turnos_medicos.especialidades LIMIT 0, 1000	4 row(s) returned	0.000 sec / 0.000 sec
5	11:49:57	SELECT * FROM turnos_medicos.turnos LIMIT 0, 1000	4 row(s) returned	0.000 sec / 0.000 sec
6	11:50:23	SELECT * FROM turnos_medicos.medicos LIMIT 0, 1000	4 row(s) returned	0.000 sec / 0.000 sec
7	11:52:46	SELECT * FROM turnos_medicos.vista_turnos_activos LIMIT 0, 1000	5 row(s) returned	0.000 sec / 0.000 sec

Automatic context help is disabled. Use the toolbar to manually get help for the current caret position or to toggle automatic help.



Sistema de Turnos Médicos

Nombre:

Alan Logroño

Cédula:

1752757340

Fecha Nac (YYYY-MM-DD):

2005-03-02

Registrar Paciente

Ver Turnos Activos

ID: 1 - Paciente: María López - Médico: Dra. Ana Torres - Fecha: 2025-07-02

ID: 2 - Paciente: Pedro González - Médico: Dr. Luis Pérez - Fecha: 2025-07-02

ID: 3 - Paciente: Lucía Martínez - Médico: Dra. Carla Gómez - Fecha: 2025-07-02

ID: 4 - Paciente: Carlos Herrera - Médico: Dr. Jorge Lima - Fecha: 2025-07-03

ID: 5 - Paciente: Antony Cisneros - Médico: Dr. Jorge Lima - Fecha: 2025-07-04

ID: 6 - Paciente: Alan Logroño - Médico: Dr. Luis Pérez - Fecha: 2025-07-04

## CONCLUSION

En la ejecución de la interfaz gráfica de java, se puede observar diferentes funciones e vistas aplicadas. Como el trigger de Registrar Paciente, y la vista de Ver turnos activos.

```

private void registrarPaciente() { 1 usage
    try (Connection conn = DBConnection.getConnection()) {
        String nombre = txtNombre.getText();
        String cedula = txtCedula.getText();
        String fecha = txtFechaNacimiento.getText();

        CallableStatement stmt = conn.prepareCall( sql: "{CALL registrar_paciente(?, ?, ?)}");
        stmt.setString( parameterIndex: 1, nombre);
        stmt.setString( parameterIndex: 2, cedula);
        stmt.setString( parameterIndex: 3, fecha);
        stmt.execute();
        txtResultados.setText("Paciente registrado correctamente.");
    } catch (SQLException ex) {
        txtResultados.setText("Error: " + ex.getMessage());
    }
}

```

```

private void mostrarTurnos() { 1 usage
    try (Connection conn = DBConnection.getConnection()) {
        String query = "SELECT * FROM vista_turnos_activos";
        PreparedStatement stmt = conn.prepareStatement(query);
        ResultSet rs = stmt.executeQuery();
        StringBuilder sb = new StringBuilder();
        while (rs.next()) {
            sb.append("ID: ").append(rs.getInt( columnLabel: "id")).append(" - ")
              .append("Paciente: ").append(rs.getString( columnLabel: "paciente")).append(" - ")
              .append("Médico: ").append(rs.getString( columnLabel: "medico")).append(" - ")
              .append("Fecha: ").append(rs.getDate( columnLabel: "fecha")).append("\n");
        }
        txtResultados.setText(sb.toString());
    } catch (SQLException ex) {
        txtResultados.setText("Error: " + ex.getMessage());
    }
}

```

De esta manera, podemos simplificar mucho las funciones de MYSQL con una interfaz grafica que facilita las acciones, tan solo instalando una librería de MYSQL Connector.