ESCUELA POLITÉCNICA NACIONAL

BASE DE DATOS

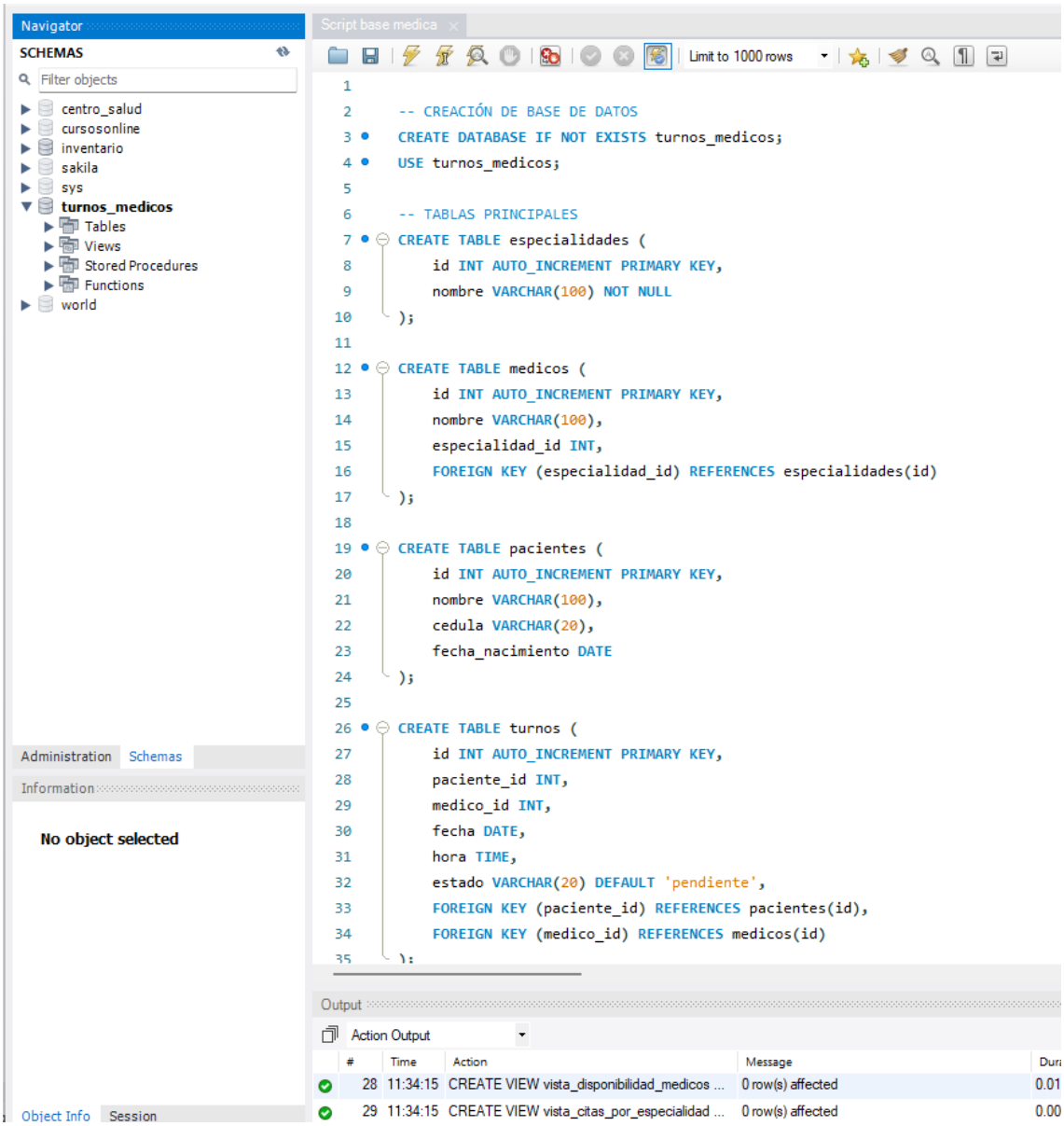
TALLER COMPLEMENTARIO

Nombre: Mercy Perugachi

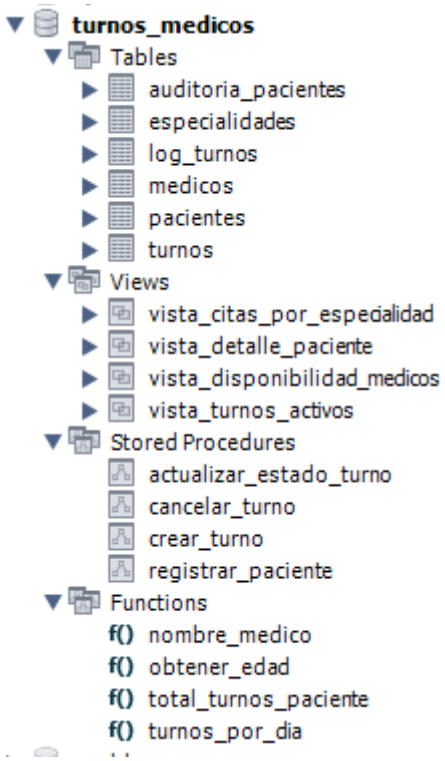
TALLER Sistema con Base de Datos y Java

Parte 1:

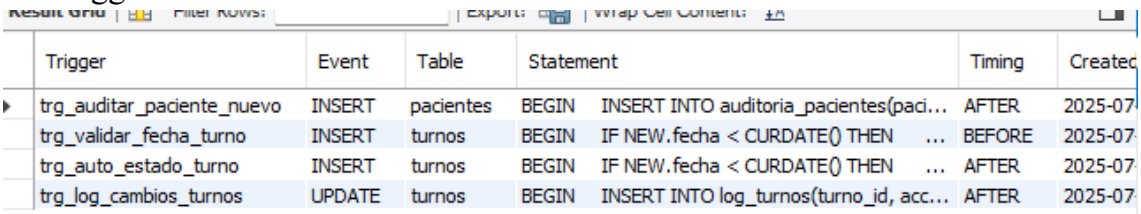
- Verificar que todos los datos coincidan con el taller 2 (Sistema de Gestión de Turnos Médicos).



- Tablas, views, procedimientos almacenadosy funciones



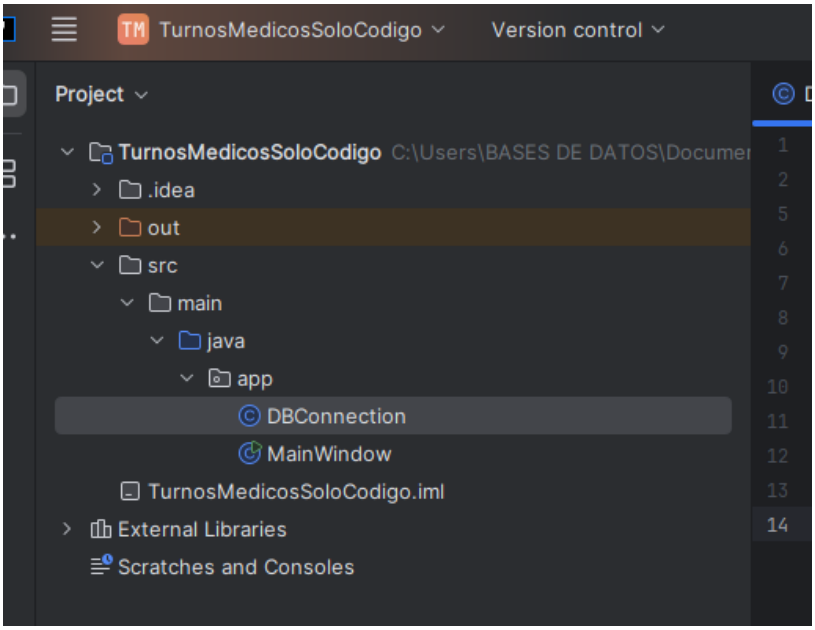
- Ver triggers



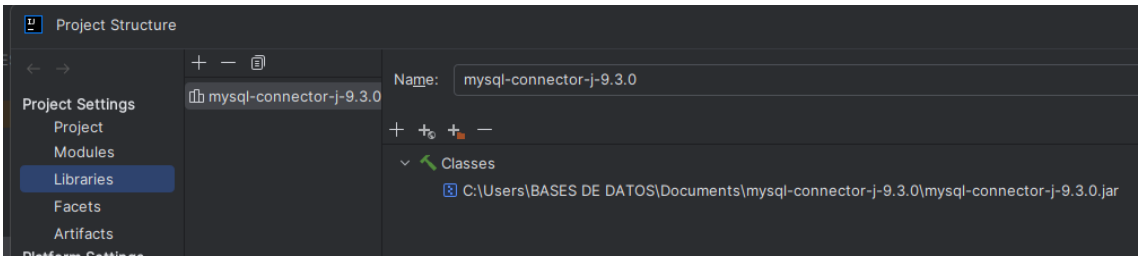
Trigger	Event	Table	Statement	Timing	Created
trg_auditar_paciente_nuevo	INSERT	pacientes	BEGIN INSERT INTO auditoria_pacientes(paci...	AFTER	2025-07
trg_validar_fecha_turno	INSERT	turnos	BEGIN IF NEW.fecha < CURDATE() THEN ...	BEFORE	2025-07
trg_auto_estado_turno	INSERT	turnos	BEGIN IF NEW.fecha < CURDATE() THEN ...	AFTER	2025-07
trg_log_cambios_turnos	UPDATE	turnos	BEGIN INSERT INTO log_turnos(turno_id, acc...	AFTER	2025-07

Parte 2:

- Abrir el proyecto y restaurar el código



- Descargar la conexión y conectar con IntelliJ

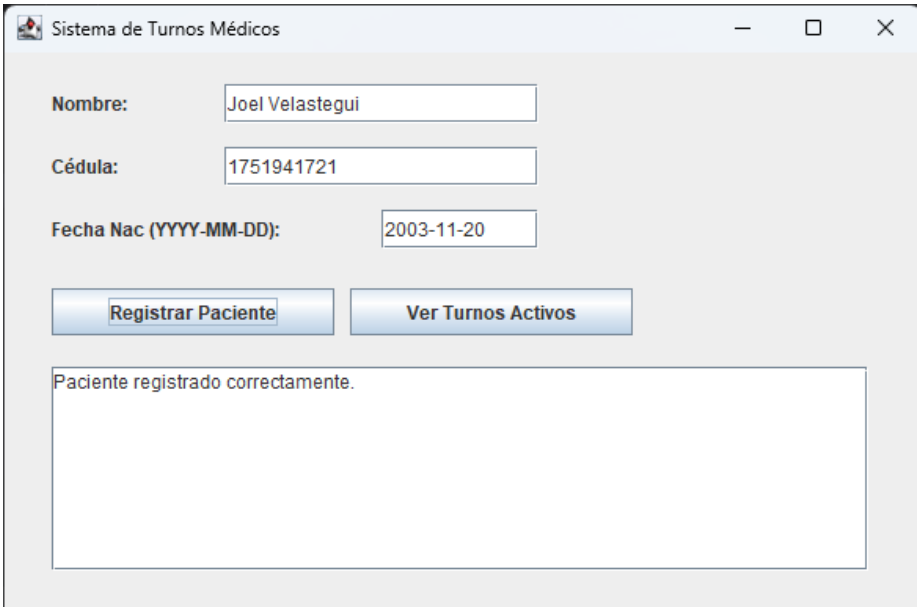


- Ejecutar el código Java y comprobar que la conexión con la base de datos funcione.

Antes:

id	nombre	cedula	fecha_nacimiento
1	María López	1102233445	1990-04-15
2	Pedro González	1103344556	1985-06-20
3	Lucía Martínez	1104455667	2002-09-10
4	Carlos Herrera	1105566778	1978-12-05
5	Ameri Velastegui	1751941707	2025-05-21
NULL	NULL	NULL	NULL

Agregar paciente:



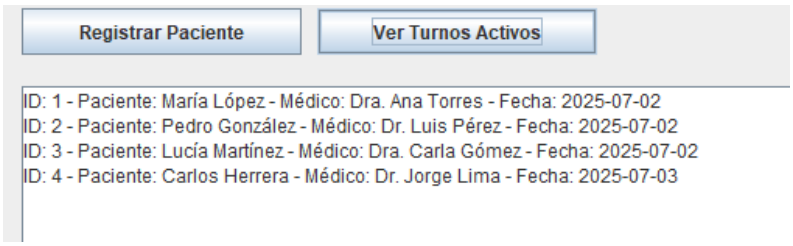
Después:

Result Grid | Filter Rows: | Edit:

id	nombre	cedula	fecha_nacimiento
1	María López	1102233445	1990-04-15
2	Pedro González	1103344556	1985-06-20
3	Lucía Martínez	1104455667	2002-09-10
4	Carlos Herrera	1105566778	1978-12-05
5	Ameri Velastegui	1751941707	2025-05-21
6	Joel Velastegui	1751941721	2003-11-20
NULL	NULL	NULL	NULL

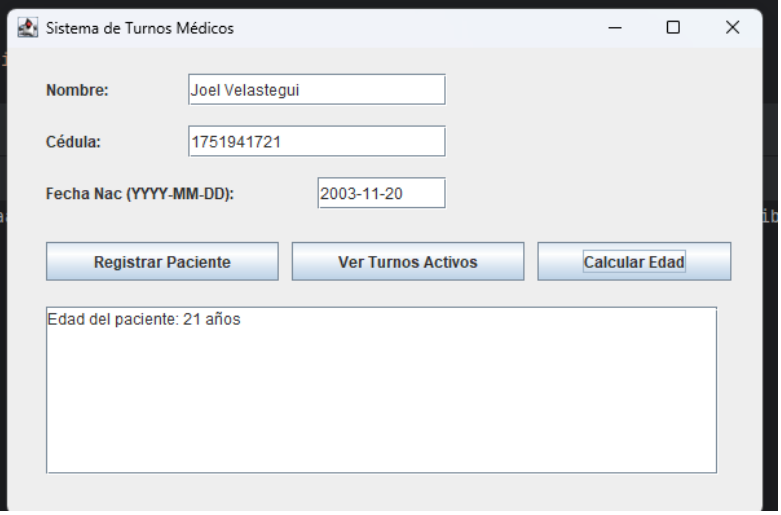
- Verificar que el código Java utilice correctamente los elementos de base de datos:
  Vistas

```
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());
    }
}
```



  Funciones

```java
private void calcularEdad() {  1 usage
    try (Connection conn = DBConnection.getConnection()) {
        String fechaNac = txtFechaNacimiento.getText();

        String query = "SELECT obtener_edad(?)";
        PreparedStatement stmt = conn.prepareStatement(query);
        stmt.setString( parameterIndex: 1, fechaNac);
        ResultSet rs = stmt.executeQuery();

        if (rs.next()) {
            int edad = rs.getInt( columnIndex: 1);
            txtResultados.setText("Edad del paciente: " + edad + " años");
        } else {
            txtResultados.setText("No se pudo calcular la edad.");
        }
    } catch (SQLException ex) {
        txtResultados.setText("Error al calcular edad: " + ex.getMessage());
    }
}
```



Procedimientos almacenados

```java
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());
    }
```

Triggers:

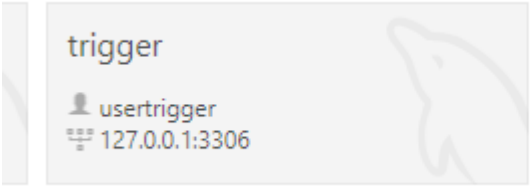
Cuando se ejecuta ese CALL, se inserta un paciente, y eso activa el trigger automáticamente.

```java
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());
    }
```

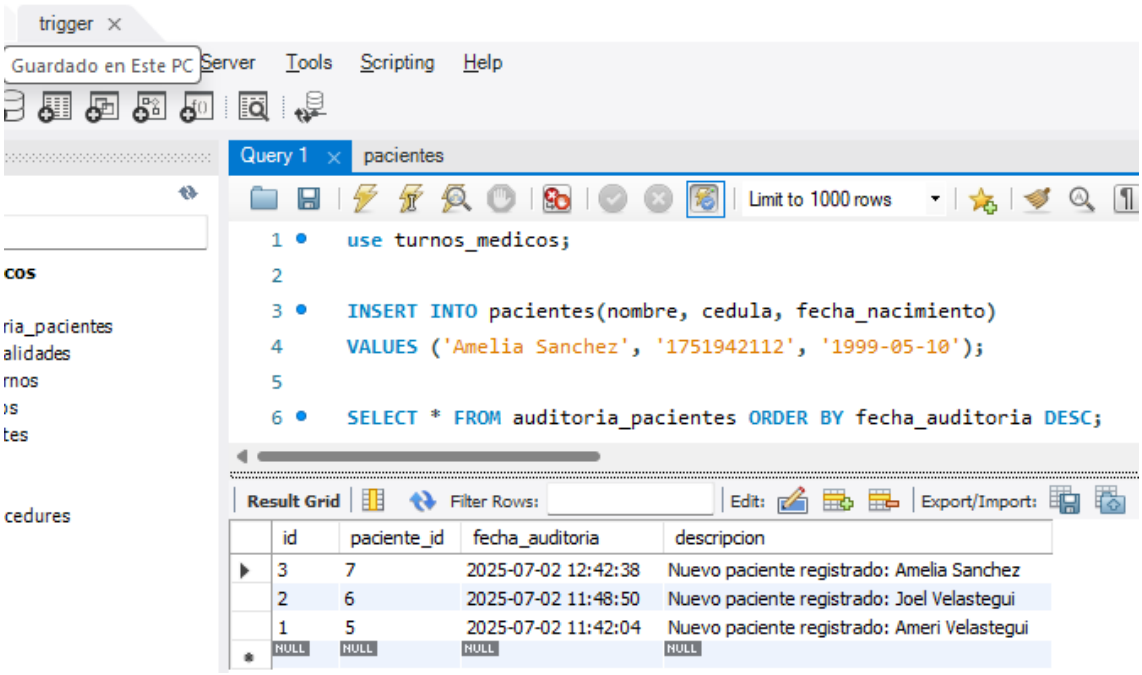
- Crear usuarios para probar las acciones del trigger

```sql
-- :::::::::::: CREAR USUARIO ::::::::::::::::::
CREATE USER 'usertrigger'@'localhost' identified by '123';

GRANT SELECT, INSERT, UPDATE ON turnos_medicos.pacientes TO 'usertrigger'@'localhost';
GRANT SELECT, INSERT, UPDATE ON turnos_medicos.auditoria_pacientes TO 'usertrigger'@'
GRANT SELECT, INSERT, UPDATE ON turnos_medicos.turnos TO 'usertrigger'@'localhost';
GRANT SELECT, INSERT ON turnos_medicos.log_turnos TO 'usertrigger'@'localhost';

GRANT USAGE ON turnos_medicos.* TO 'usertrigger'@'localhost';

GRANT SELECT, INSERT, UPDATE, EXECUTE
ON turnos_medicos.* TO 'usertrigger'@'localhost';



FLUSH PRIVILEGES;
```

Crear la conexión:



Ejecutar el código:



CONCLUSIÓN:

El taller de práctica de hoy es de gran ayuda para poder realizar la conexión entre una base de datos en MySQL a la aplicación IntelliJ, posterior a ello, también lograr que se suba a un repositorio con ayuda de GitHub. No tenía conocimiento sobre ello, pero ahora estoy segura de que me va servir en un futuro para mi crecimiento profesional. Todo esto es conjunto son herramientas con gran potencial, la cuestión es saber utilizarlo de forma correcta y tener los conocimientos adecuados.