**CENTRO MÉDICO DE BIENESTAR**



***https://github.com/AngelyCastellanos/ProyectoBBDD\_Angely.git***

**Nome Alumno/a:**

***Angely Gissell Castellanos Cala***

**Curso: *1º DAM* Materia:** ***Bases de Datos – Proyecto Final 24/25***

Contido

[1.](#_ibpkshqfyxaf) Introducción 2

[2.](#_gacn603r8bcm) Descripción del Problema / Requisitos 2

[3.](#_oh8p6x583d9t) Modelo Conceptual 2

[4.](#_xxfwe1irpupd) Modelo Relacional 2

[5.](#_hoo0st5t0n32) Proceso de Normalización 2

[6.](#_lryty5r2vd99) Script de Creación de la Base de Datos 2

[7.](#_vh855dl63p15) Carga de Datos Inicial 2

[8.](#_fr8kret7ohra) Funciones y Procedimientos Almacenados 2

[9.](#_e5982q4vj4jj) Triggers 2

[10.](#_lqbxxmucxh4s) Consultas SQL 2

[11.](#_8g7kxuget6i6) Casos de Prueba y Simulación 2

[12.](#_dg31ek701tz) Resultados y Verificación 2

[13.](#_bzui7v4ahut9) Capturas de Pantalla (opcional) 2

[14.](#_cswye3vdquya) Conclusiones y Mejoras Futuras 2

[15.](#_u2uujms4oen7) Enlace al Repositorio en GitHub 3

# Introducción

El proyecto consiste en el diseño y desarrollo de una base de datos para gestionar la información de un centro médico facilitando el manejo y orden de los datos que se vayan a gestionar.

# Descripción del Problema / Requisitos

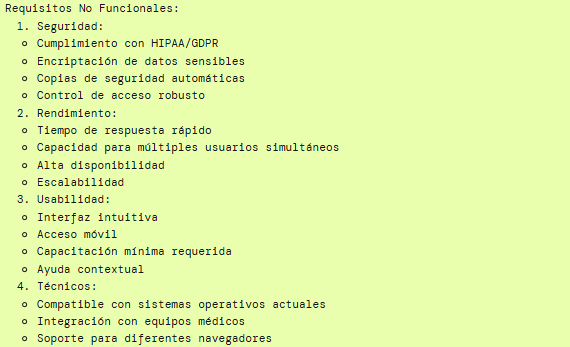
**Contexto**

Como dueño del "Centro Médico Bienestar", este dirige una clínica privada multiespecialidad que ofrece atención médica integral y personalizada. Cuenta con más de 15 consultorios equipados con tecnología de punta, un laboratorio clínico y servicios de imagenología.

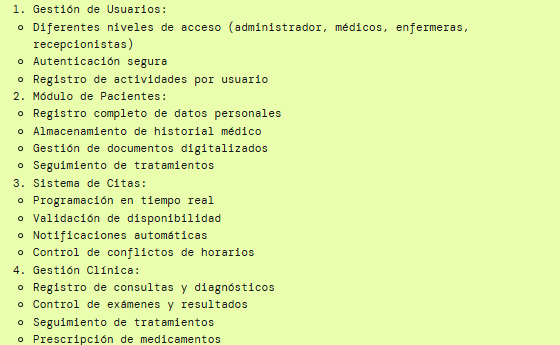
El centro opera de lunes a sábado, con horario extendido hasta las 8 p.m. entre semana. Cuenta con el personal permanente de 8 médicos especialistas y tiene convenios con más de 20 especialistas visitantes. El equipo humano incluye a 12 enfermeras, 4 técnicos de laboratorio y 6 administrativos.

Se distingue por ofrecer citas el mismo día para urgencias menores y por mantener tiempos de espera reducidos. Trabaja con las principales aseguradoras y ofrece planes de pago accesibles para pacientes sin seguro. La satisfacción del paciente y la excelencia médica son sus prioridades.

**Requisitos no funcionales**

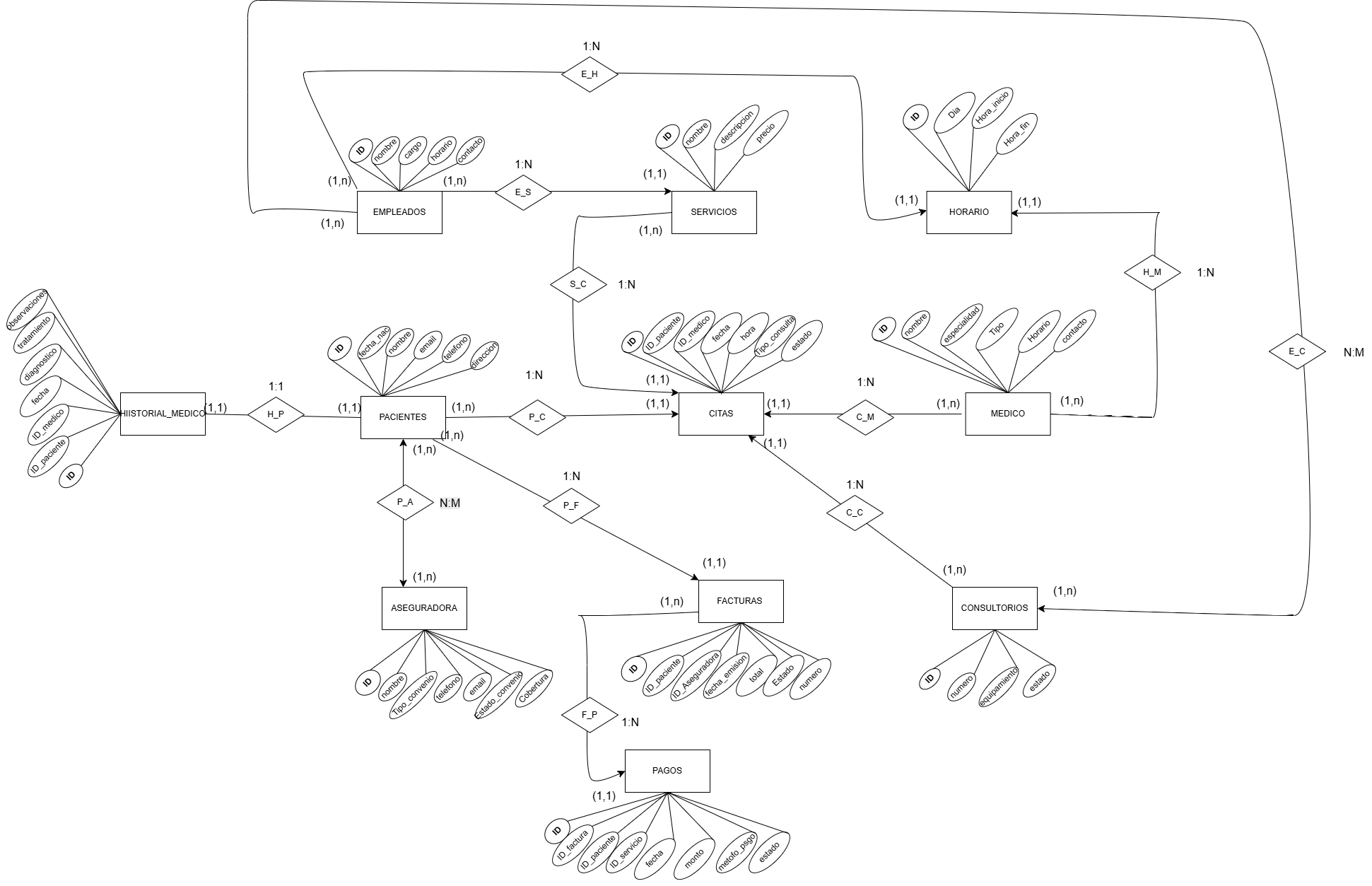


**Requisitos funcionales**

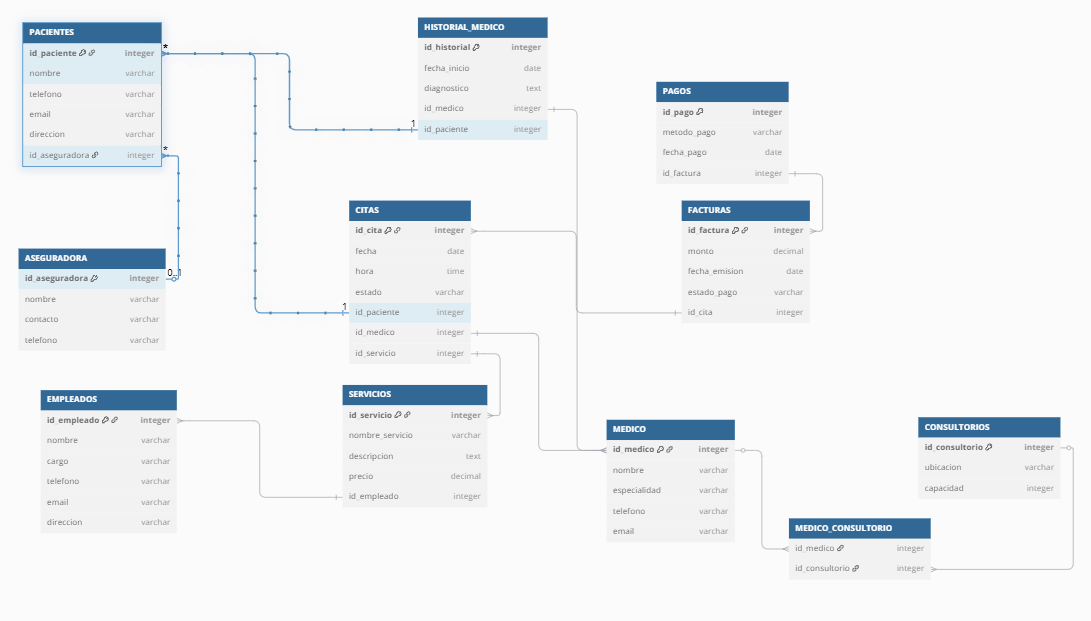


# Modelo Conceptual

Para el diseño del modelo conceptual, se identificó las entidades principales como Empleados, Servicio, Horario, Historial Médico, Citas, Pacientes, Médico, Aseguradora, Factura, Consultorio y Pagos, incluyendo los campos más relevantes de cada una.



# Modelo Relacional

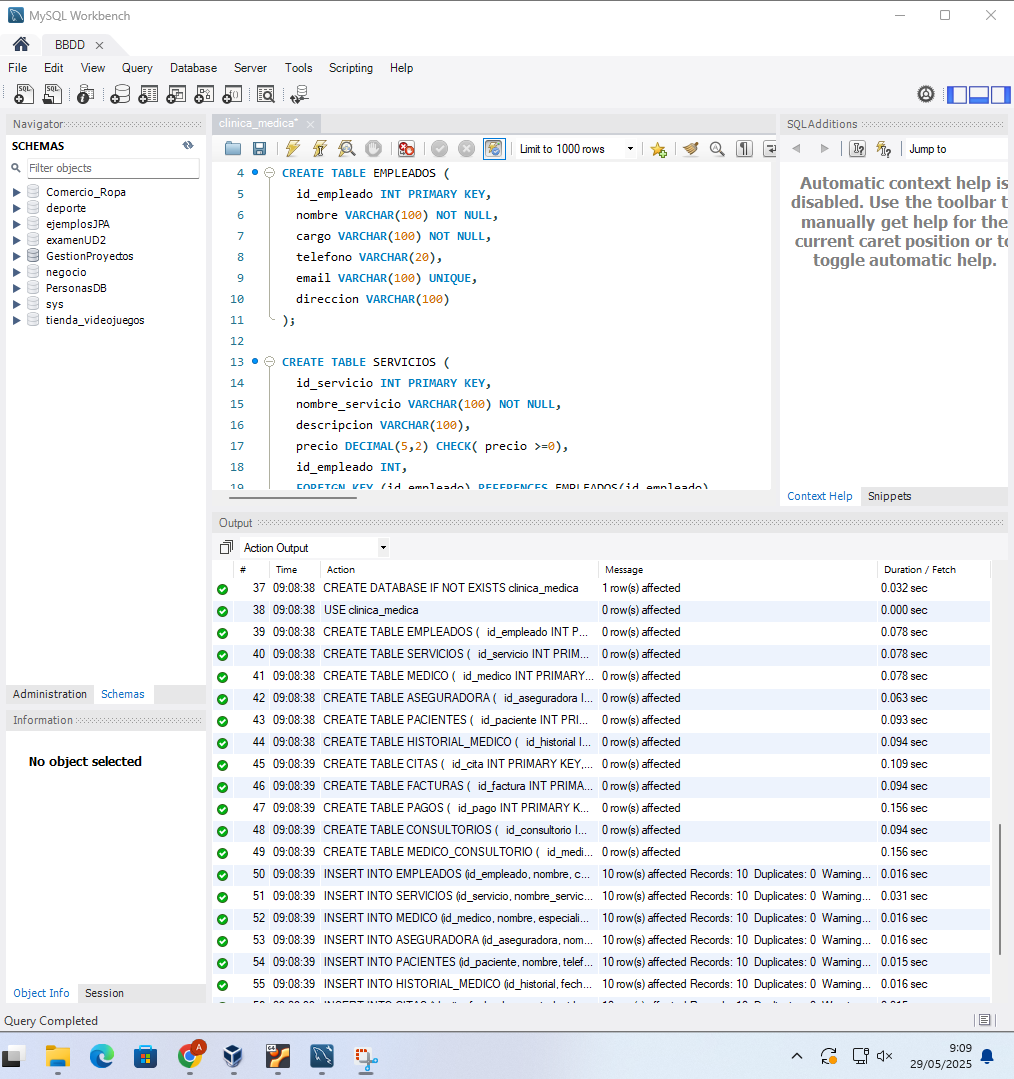


# Proceso de Normalización

Tras seguir el proceso de normalización de la tercera forma, todas las tablas han sido estructuradas de acuerdo con los estándares habituales, garantizando la ausencia de dependencias transitivas y asegurando así una organización consistente y eficiente de los datos.

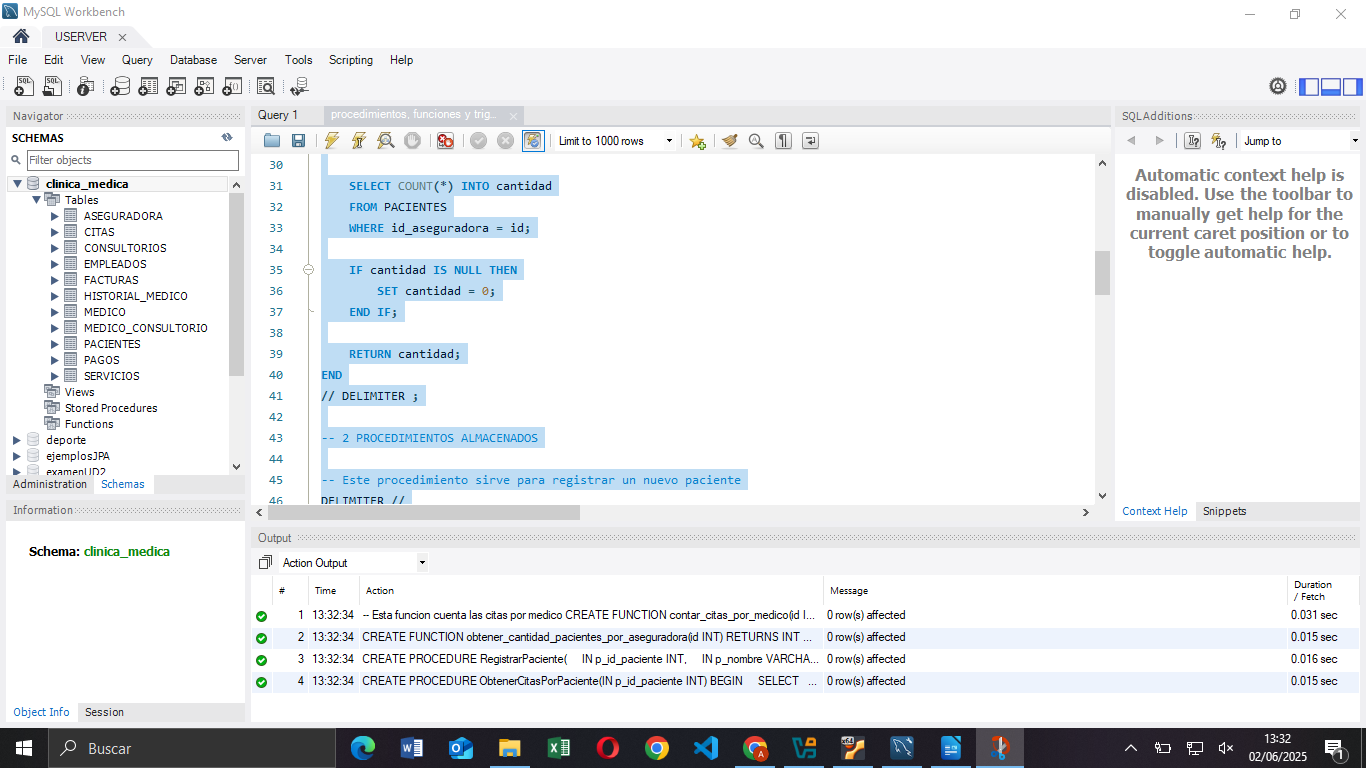
# Script de Creación de la Base de Datos

|  |
| --- |
| CREATE DATABASE IF NOT EXISTS clinica\_medica; USE clinica\_medica;  CREATE TABLE EMPLEADOS (  id\_empleado INT PRIMARY KEY,  nombre VARCHAR(100) NOT NULL,  cargo VARCHAR(100) NOT NULL,  telefono VARCHAR(20),  email VARCHAR(100) UNIQUE,  direccion VARCHAR(100) );  CREATE TABLE SERVICIOS (  id\_servicio INT PRIMARY KEY,  nombre\_servicio VARCHAR(100) NOT NULL,  descripcion VARCHAR(100),  precio DECIMAL(5,2) CHECK( precio >=0),  id\_empleado INT,  FOREIGN KEY (id\_empleado) REFERENCES EMPLEADOS(id\_empleado) );  CREATE TABLE MEDICO (  id\_medico INT PRIMARY KEY,  nombre VARCHAR(100) NOT NULL,  especialidad VARCHAR(100),  telefono VARCHAR(20),  email VARCHAR(100) UNIQUE );  CREATE TABLE ASEGURADORA (  id\_aseguradora INT PRIMARY KEY,  nombre VARCHAR(100) NOT NULL,  contacto VARCHAR(100),  telefono VARCHAR(20) );  CREATE TABLE PACIENTES (  id\_paciente INT PRIMARY KEY,  nombre VARCHAR(100) NOT NULL,  telefono VARCHAR(20),  email VARCHAR(100) UNIQUE,  direccion VARCHAR(100),  id\_aseguradora INT,  FOREIGN KEY (id\_aseguradora) REFERENCES ASEGURADORA(id\_aseguradora) );   CREATE TABLE HISTORIAL\_MEDICO (  id\_historial INT PRIMARY KEY,  fecha\_inicio DATETIME DEFAULT CURRENT\_TIMESTAMP,  diagnostico VARCHAR(100),  id\_medico INT,  id\_paciente INT,  FOREIGN KEY (id\_medico) REFERENCES MEDICO(id\_medico),  FOREIGN KEY (id\_paciente) REFERENCES PACIENTES(id\_paciente) );  CREATE TABLE CITAS (  id\_cita INT PRIMARY KEY,  fecha DATE NOT NULL,  hora TIME NOT NULL,  estado VARCHAR(100),  id\_paciente INT,  id\_medico INT,  id\_servicio INT,  FOREIGN KEY (id\_paciente) REFERENCES PACIENTES(id\_paciente),  FOREIGN KEY (id\_medico) REFERENCES MEDICO(id\_medico),  FOREIGN KEY (id\_servicio) REFERENCES SERVICIOS(id\_servicio) );  CREATE TABLE FACTURAS (  id\_factura INT PRIMARY KEY,  monto DECIMAL(5,2) CHECK (monto>=0),  fecha\_emision DATETIME DEFAULT CURRENT\_TIMESTAMP,  estado\_pago VARCHAR(100),  id\_cita INT,  FOREIGN KEY (id\_cita) REFERENCES CITAS(id\_cita) );  CREATE TABLE PAGOS (  id\_pago INT PRIMARY KEY,  metodo\_pago VARCHAR(100) NOT NULL,  fecha\_pago DATETIME DEFAULT CURRENT\_TIMESTAMP,  id\_factura INT,  FOREIGN KEY (id\_factura) REFERENCES FACTURAS(id\_factura) );  CREATE TABLE CONSULTORIOS (  id\_consultorio INT PRIMARY KEY,  ubicacion VARCHAR(100) NOT NULL,  capacidad INT CHECK (capacidad > 0) );  CREATE TABLE MEDICO\_CONSULTORIO (  id\_medico INT,  id\_consultorio INT,  PRIMARY KEY (id\_medico, id\_consultorio),  FOREIGN KEY (id\_medico) REFERENCES MEDICO(id\_medico),  FOREIGN KEY (id\_consultorio) REFERENCES CONSULTORIOS(id\_consultorio) );  INSERT INTO EMPLEADOS (id\_empleado, nombre, cargo, telefono, email, direccion) VALUES (1, 'Carlos Mendoza', 'Administrador', '3101234567', 'carlos.mendoza@clinicamedica.com', 'Calle 10 #23-45'), (2, 'Laura Pérez', 'Recepcionista', '3112345678', 'laura.perez@clinicamedica.com', 'Avenida 15 #45-67'), (3, 'Javier Torres', 'Técnico', '3123456789', 'javier.torres@clinicamedica.com', 'Carrera 8 #12-34'), (4, 'Sofía Ramírez', 'Asistente Médico', '3134567890', 'sofia.ramirez@clinicamedica.com', 'Diagonal 5 #78-90'), (5, 'Andrés López', 'Contador', '3145678901', 'andres.lopez@clinicamedica.com', 'Transversal 3 #56-78'), (6, 'Patricia Gómez', 'Secretaria', '3156789012', 'patricia.gomez@clinicamedica.com', 'Calle 12 #89-01'), (7, 'Diego Martínez', 'Mantenimiento', '3167890123', 'diego.martinez@clinicamedica.com', 'Carrera 14 #34-56'), (8, 'Valentina Díaz', 'Soporte Técnico', '3178901234', 'valentina.diaz@clinicamedica.com', 'Avenida 7 #65-43'), (9, 'Fernando Cruz', 'Gerente', '3189012345', 'fernando.cruz@clinicamedica.com', 'Calle 5 #21-87'), (10, 'Camila Rojas', 'Coordinadora', '3190123456', 'camila.rojas@clinicamedica.com', 'Carrera 2 #98-76');  INSERT INTO SERVICIOS (id\_servicio, nombre\_servicio, descripcion, precio, id\_empleado) VALUES (1, 'Consulta General', 'Primera consulta médica general', 50.00, 1), (2, 'Urgencias', 'Atención de emergencia las 24 horas', 100.00, 2), (3, 'Laboratorio Clínico', 'Análisis sanguíneos y pruebas clínicas', 40.00, 3), (4, 'Rayos X', 'Estudios radiográficos generales', 60.00, 4), (5, 'Ecografía', 'Exploración por ultrasonido abdominal', 80.00, 5), (6, 'Vacunación', 'Programa de inmunización básica', 30.00, 6), (7, 'Control Prenatal', 'Seguimiento médico durante el embarazo', 70.00, 7), (8, 'Fisioterapia', 'Tratamiento físico post-operatorio o lesiones', 90.00, 8), (9, 'Nutrición', 'Evaluación nutricional y plan alimenticio', 45.00, 9), (10, 'Psicología Clínica', 'Consulta psicológica individual', 55.00, 10);  INSERT INTO MEDICO (id\_medico, nombre, especialidad, telefono, email) VALUES (1, 'Dr. Juan Morales', 'Cardiología', '3102345678', 'juan.morales@medicoscl.com'), (2, 'Dra. Ana Fernández', 'Pediatría', '3113456789', 'ana.fernandez@medicoscl.com'), (3, 'Dr. Luis Castillo', 'Dermatología', '3124567890', 'luis.castillo@medicoscl.com'), (4, 'Dra. Elena Vargas', 'Neurología', '3135678901', 'elena.vargas@medicoscl.com'), (5, 'Dr. Pablo Ríos', 'Traumatología', '3146789012', 'pablo.rios@medicoscl.com'), (6, 'Dra. María Soto', 'Ginecología', '3157890123', 'maria.soto@medicoscl.com'), (7, 'Dr. Andrés Cárdenas', 'Oftalmología', '3168901234', 'andres.cardenas@medicoscl.com'), (8, 'Dra. Laura Ortega', 'Endocrinología', '3179012345', 'laura.ortega@medicoscl.com'), (9, 'Dr. Miguel Torres', 'Urología', '3180123456', 'miguel.torres@medicoscl.com'), (10, 'Dra. Patricia Rojas', 'Psiquiatría', '3191234567', 'patricia.rojas@medicoscl.com');  INSERT INTO ASEGURADORA (id\_aseguradora, nombre, contacto, telefono) VALUES (1, 'SaludTotal S.A.', 'Juan Camargo', '3201234567'), (2, 'VidaPlus Seguros', 'María León', '3212345678'), (3, 'Protección Salud', 'Carlos Duarte', '3223456789'), (4, 'Medicare Plus', 'Luisa Rueda', '3234567890'), (5, 'Seguros del Hogar', 'Daniela Mejía', '3245678901'), (6, 'Allianz Colombia', 'Fernando Ocampo', '3256789012'), (7, 'Bolívar Salud', 'Jorge Medina', '3267890123'), (8, 'AXA Salud S.A.', 'Carolina Roa', '3278901234'), (9, 'Mapfre Salud', 'Rafael Niño', '3289012345'), (10, 'Assist Card Salud', 'Paola Vega', '3290123456');  INSERT INTO PACIENTES (id\_paciente, nombre, telefono, email, direccion, id\_aseguradora) VALUES (1, 'Luis Jiménez', '3109876543', 'luis.jimenez@gmail.com', 'Calle 10 #23-45', 1), (2, 'María Fernández', '3118765432', 'maria.fernandez@hotmail.com', 'Avenida 15 #45-67', 2), (3, 'Carlos Rojas', '3127654321', 'carlos.rojas@yahoo.com', 'Carrera 8 #12-34', 3), (4, 'Ana Lucía Torres', '3136543210', 'anatorres@outlook.com', 'Diagonal 5 #78-90', 4), (5, 'Javier Mendoza', '3145432109', 'javier.mendoza@gmail.com', 'Transversal 3 #56-78', 5), (6, 'Sofía Pérez', '3154321098', 'sofia.perez@hotmail.com', 'Calle 12 #89-01', 6), (7, 'David Gómez', '3163210987', 'david.gomez@yahoo.com', 'Av. 7 #45-23', 7), (8, 'Lucía Ramírez', '3172109876', 'lucia.ramirez@outlook.com', 'Carrera 8 #12-34', 8), (9, 'Alejandro Castro', '3181098765', 'alejandro.castro@gmail.com', 'Calle 9 #34-56', 9), (10, 'Daniela Navarro', '3190987654', 'daniela.navarro@hotmail.com', 'Av. 10 #12-34', 10);  INSERT INTO HISTORIAL\_MEDICO (id\_historial, fecha\_inicio, diagnostico, id\_medico, id\_paciente) VALUES (1, '2024-01-10 09:30:00', 'Hipertensión arterial esencial', 1, 1), (2, '2024-02-15 10:15:00', 'Resfriado común leve', 2, 2), (3, '2024-03-01 11:00:00', 'Dermatitis alérgica', 3, 3), (4, '2024-04-05 14:20:00', 'Trastorno de ansiedad generalizada', 4, 4), (5, '2024-05-12 08:45:00', 'Fractura distal de radio derecho', 5, 5), (6, '2024-06-20 12:10:00', 'Embarazo en semana 12', 6, 6), (7, '2024-07-22 15:55:00', 'Miopía moderada bilateral', 7, 7), (8, '2024-08-30 16:40:00', 'Diabetes tipo 2 controlada', 8, 8), (9, '2024-09-10 09:15:00', 'Infección urinaria recurrente', 9, 9), (10, '2024-10-15 10:00:00', 'Depresión leve episódica', 10, 10);  INSERT INTO CITAS (id\_cita, fecha, hora, estado, id\_paciente, id\_medico, id\_servicio) VALUES (1, '2025-04-01', '09:00:00', 'Confirmada', 1, 1, 1), (2, '2025-04-01', '10:00:00', 'Pendiente', 2, 2, 2), (3, '2025-04-02', '11:00:00', 'Cancelada', 3, 3, 3), (4, '2025-04-02', '12:00:00', 'Confirmada', 4, 4, 4), (5, '2025-04-03', '13:00:00', 'Pendiente', 5, 5, 5), (6, '2025-04-03', '14:00:00', 'Confirmada', 6, 6, 6), (7, '2025-04-04', '15:00:00', 'Pendiente', 7, 7, 7), (8, '2025-04-04', '16:00:00', 'Confirmada', 8, 8, 8), (9, '2025-04-05', '17:00:00', 'Pendiente', 9, 9, 9), (10, '2025-04-05', '18:00:00', 'Cancelada', 10, 10, 10);  INSERT INTO FACTURAS (id\_factura, monto, fecha\_emision, estado\_pago, id\_cita) VALUES (1, 50.00, '2025-04-01 09:15:00', 'Pagado', 1), (2, 100.00, '2025-04-01 10:10:00', 'Pendiente', 2), (3, 40.00, '2025-04-02 11:15:00', 'Pagado', 3), (4, 60.00, '2025-04-02 12:20:00', 'Pendiente', 4), (5, 80.00, '2025-04-03 13:25:00', 'Pagado', 5), (6, 30.00, '2025-04-03 14:30:00', 'Pendiente', 6), (7, 70.00, '2025-04-04 15:35:00', 'Pagado', 7), (8, 90.00, '2025-04-04 16:40:00', 'Pendiente', 8), (9, 45.00, '2025-04-05 17:45:00', 'Pagado', 9), (10, 55.00, '2025-04-05 18:50:00', 'Pendiente', 10);  INSERT INTO PAGOS (id\_pago, metodo\_pago, fecha\_pago, id\_factura) VALUES (1, 'Tarjeta de Crédito', '2025-04-01 09:20:00', 1), (2, 'Efectivo', '2025-04-02 11:20:00', 3), (3, 'Transferencia Bancaria', '2025-04-03 13:30:00', 5), (4, 'PayPal', '2025-04-04 15:40:00', 7), (5, 'Tarjeta Débito', '2025-04-05 17:50:00', 9), (6, 'Tarjeta de Crédito', '2025-04-01 10:15:00', 2), (7, 'Efectivo', '2025-04-02 12:25:00', 4), (8, 'Transferencia Bancaria', '2025-04-03 14:35:00', 6), (9, 'PayPal', '2025-04-04 16:45:00', 8), (10, 'Tarjeta Débito', '2025-04-05 18:55:00', 10);  INSERT INTO CONSULTORIOS (id\_consultorio, ubicacion, capacidad) VALUES (1, 'Planta Baja - Consultorio 1', 2), (2, 'Primer Piso - Consultorio 2', 3), (3, 'Segundo Piso - Consultorio 3', 4), (4, 'Ala Norte - Consultorio 4', 2), (5, 'Ala Sur - Consultorio 5', 3), (6, 'Consultorio Externo 1 - Urgencias', 1), (7, 'Consultorio Externo 2 - Psicología', 1), (8, 'Área de Urgencias', 5), (9, 'Laboratorio Clínico', 4), (10, 'Sala de Rayos X', 2);  INSERT INTO MEDICO\_CONSULTORIO (id\_medico, id\_consultorio) VALUES (1, 1), (2, 2), (3, 3), (4, 4), (5, 5), (6, 6), (7, 7), (8, 8), (9, 9), (10, 10); |



# Funciones y Procedimientos Almacenados

|  |
| --- |
| -- 2 FUNCIONES ALMACENADAS DELIMITER // -- Esta funcion cuenta las citas por medico CREATE FUNCTION contar\_citas\_por\_medico(id INT) RETURNS INT DETERMINISTIC BEGIN  DECLARE total\_citas INT;   SELECT COUNT(\*) INTO total\_citas  FROM CITAS  WHERE id\_medico = id;   IF total\_citas IS NULL THEN  RETURN 0;  ELSE  RETURN total\_citas;  END IF;  END // DELIMITER ;  -- Esta funcion cuenta la cantidad de pacientes que tienen una aseguradora DELIMITER // CREATE FUNCTION obtener\_cantidad\_pacientes\_por\_aseguradora(id INT) RETURNS INT DETERMINISTIC BEGIN  DECLARE cantidad INT;   SELECT COUNT(\*) INTO cantidad  FROM PACIENTES  WHERE id\_aseguradora = id;   IF cantidad IS NULL THEN  SET cantidad = 0;  END IF;   RETURN cantidad; END // DELIMITER ;  -- 2 PROCEDIMIENTOS ALMACENADOS  -- Este procedimiento sirve para registrar un nuevo paciente DELIMITER // CREATE PROCEDURE RegistrarPaciente(  IN p\_id\_paciente INT,  IN p\_nombre VARCHAR(100),  IN p\_telefono VARCHAR(10),  IN p\_email VARCHAR(100),  IN p\_direccion VARCHAR(100),  IN p\_id\_aseguradora INT) BEGIN  INSERT INTO PACIENTES (  id\_paciente, nombre, telefono, email, direccion, id\_aseguradora) VALUES   (p\_id\_paciente, p\_nombre, p\_telefono, p\_email, p\_direccion, p\_id\_aseguradora); END // DELIMITER ;  -- Esta procedimiento lista todas las citas de un paciente DELIMITER // CREATE PROCEDURE ObtenerCitasPorPaciente(IN p\_id\_paciente INT) BEGIN  SELECT   c.id\_cita,   c.fecha,   c.hora,   c.estado,  m.nombre AS medico,  s.nombre\_servicio  FROM CITAS c  JOIN MEDICO m ON c.id\_medico = m.id\_medico  JOIN SERVICIOS s ON c.id\_servicio = s.id\_servicio  WHERE c.id\_paciente = p\_id\_paciente; END // DELIMITER ; |

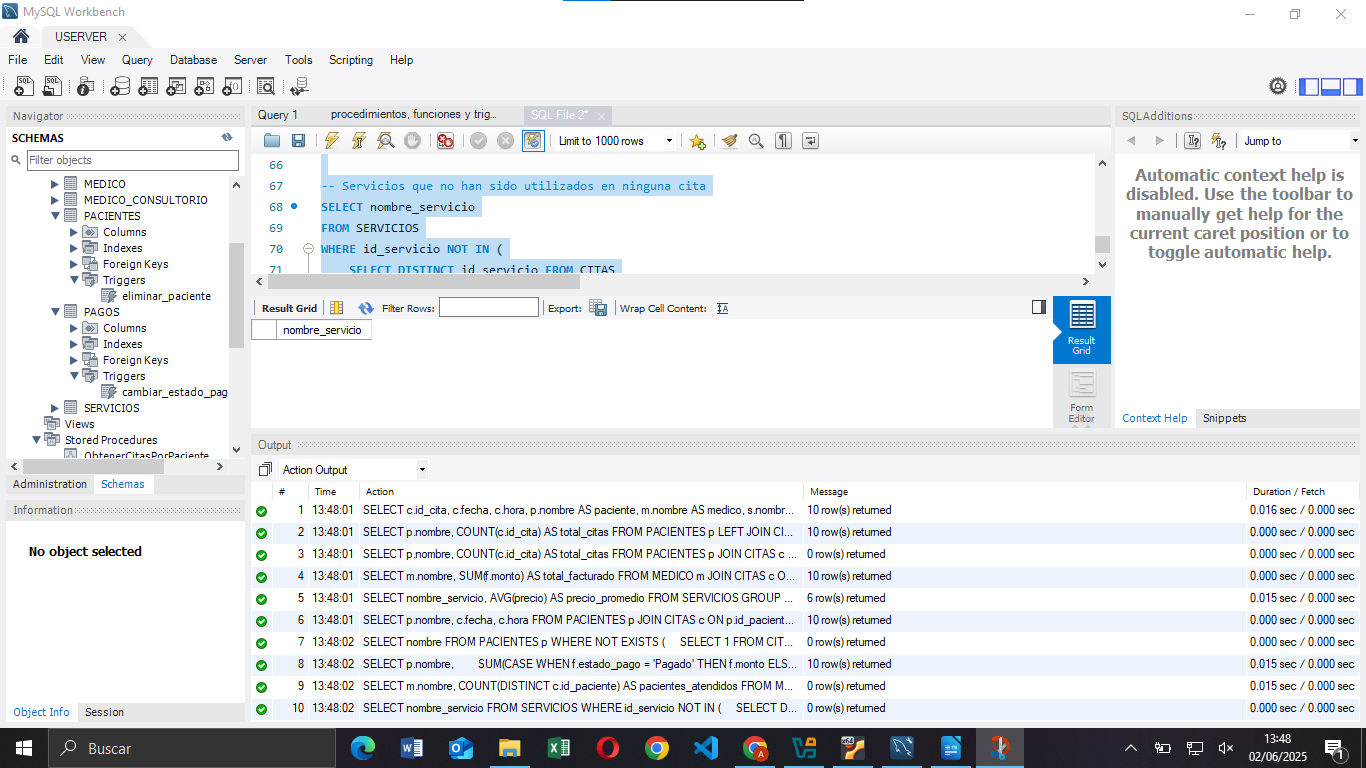


# Triggers

|  |
| --- |
| -- 2 TRIGGERS   -- Este trigger evita que se elimine un paciente con citas DELIMITER // CREATE TRIGGER eliminar\_paciente BEFORE DELETE ON PACIENTES FOR EACH ROW BEGIN  DECLARE total\_citas INT;  SELECT COUNT(\*) INTO total\_citas  FROM CITAS  WHERE id\_paciente = OLD.id\_paciente;   IF total\_citas > 0 THEN  SIGNAL SQLSTATE '45000'  SET MESSAGE\_TEXT = 'No se puede eliminar el paciente porque tiene citas.';  END IF; END // DELIMITER ;  -- Este trigger se encarga de actualizar el estado de una factura DELIMITER // CREATE TRIGGER cambiar\_estado\_pago AFTER INSERT ON PAGOS FOR EACH ROW BEGIN  UPDATE FACTURAS  SET estado\_pago = 'Pagado'  WHERE id\_factura = NEW.id\_factura; END  // DELIMITER ; |

# Consultas SQL

|  |
| --- |
| -- Lista todas las citas con datos del paciente, médico y servicio SELECT c.id\_cita, c.fecha, c.hora, p.nombre AS paciente, m.nombre AS medico, s.nombre\_servicio FROM CITAS c JOIN PACIENTES p ON c.id\_paciente = p.id\_paciente JOIN MEDICO m ON c.id\_medico = m.id\_medico JOIN SERVICIOS s ON c.id\_servicio = s.id\_servicio;  -- Cuenta cuántas citas tiene cada paciente SELECT p.nombre, COUNT(c.id\_cita) AS total\_citas FROM PACIENTES p LEFT JOIN CITAS c ON p.id\_paciente = c.id\_paciente GROUP BY p.id\_paciente, p.nombre;  -- Pacientes con más de 2 citas SELECT p.nombre, COUNT(c.id\_cita) AS total\_citas FROM PACIENTES p JOIN CITAS c ON p.id\_paciente = c.id\_paciente GROUP BY p.id\_paciente, p.nombre HAVING COUNT(c.id\_cita) > 2;  -- Total facturado por médico SELECT m.nombre, SUM(f.monto) AS total\_facturado FROM MEDICO m JOIN CITAS c ON m.id\_medico = c.id\_medico JOIN FACTURAS f ON c.id\_cita = f.id\_cita GROUP BY m.id\_medico, m.nombre;  -- Servicios con precio promedio mayor a 50 SELECT nombre\_servicio, AVG(precio) AS precio\_promedio FROM SERVICIOS GROUP BY nombre\_servicio HAVING AVG(precio) > 50;  -- Última cita de cada paciente SELECT p.nombre, c.fecha, c.hora FROM PACIENTES p JOIN CITAS c ON p.id\_paciente = c.id\_paciente WHERE c.fecha = (  SELECT MAX(fecha)   FROM CITAS   WHERE id\_paciente = p.id\_paciente );  -- Pacientes que no tienen citas SELECT nombre FROM PACIENTES p WHERE NOT EXISTS (  SELECT 1 FROM CITAS c WHERE c.id\_paciente = p.id\_paciente );  -- Total pagado y pendiente por paciente SELECT p.nombre,  SUM(CASE WHEN f.estado\_pago = 'Pagado' THEN f.monto ELSE 0 END) AS total\_pagado,  SUM(CASE WHEN f.estado\_pago = 'Pendiente' THEN f.monto ELSE 0 END) AS total\_pendiente FROM PACIENTES p JOIN CITAS c ON p.id\_paciente = c.id\_paciente JOIN FACTURAS f ON c.id\_cita = f.id\_cita GROUP BY p.id\_paciente, p.nombre;  -- Médicos que atienden más de 3 pacientes distintos SELECT m.nombre, COUNT(DISTINCT c.id\_paciente) AS pacientes\_atendidos FROM MEDICO m JOIN CITAS c ON m.id\_medico = c.id\_medico GROUP BY m.id\_medico, m.nombre HAVING pacientes\_atendidos > 3;  -- Servicios que no han sido utilizados en ninguna cita SELECT nombre\_servicio FROM SERVICIOS WHERE id\_servicio NOT IN (  SELECT DISTINCT id\_servicio FROM CITAS ); |

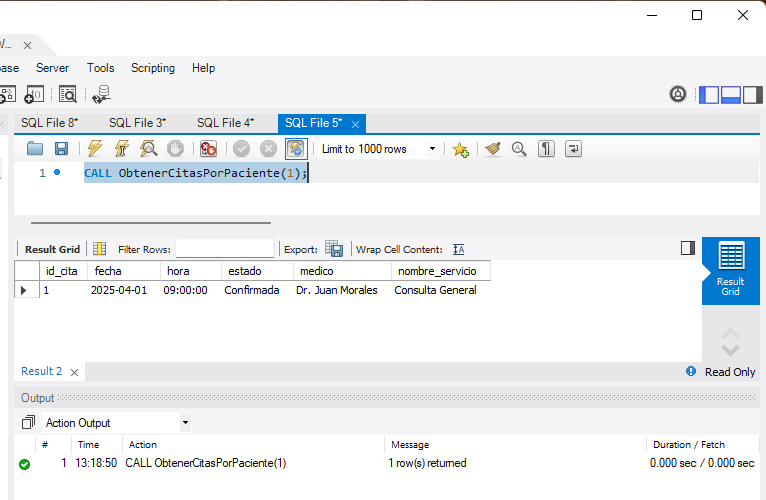


# Simulación y casos de prueba

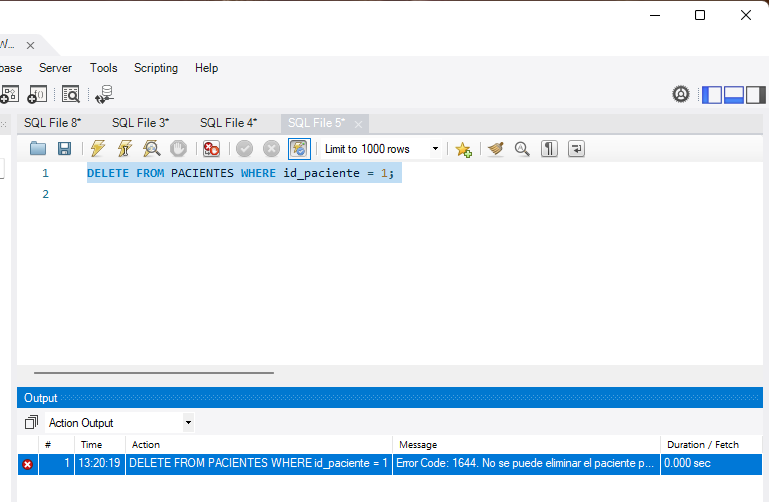
|  |
| --- |
| -- CASO 1: Registrar un nuevo paciente CALL RegistrarPaciente(23, 'Mateo Andrade', '3001122334', 'mateo.andrad@gmail.com', 'Calle 45 #12-34', 3);  -- CASO 2: Obtener citas de un paciente específico CALL ObtenerCitasPorPaciente(1);  -- CASO 3: Evitar eliminación de paciente con citas activas DELETE FROM PACIENTES WHERE id\_paciente = 1;  -- CASO 4: Insertar un pago INSERT INTO PAGOS (id\_pago, metodo\_pago, fecha\_pago, id\_factura)  VALUES (11, 'Efectivo', '2025-04-06 09:00:00', 2);  -- CASO 5: Usar función para contar citas de un médico SELECT contar\_citas\_por\_medico(1) AS total\_citas\_medico\_1;  -- CASO 6: Identificar servicios no utilizados SELECT nombre\_servicio  FROM SERVICIOS  WHERE id\_servicio NOT IN (  SELECT DISTINCT id\_servicio FROM CITAS );  -- CASO 7: Verificar total facturado por médico SELECT m.nombre, SUM(f.monto) AS total\_facturado FROM MEDICO m JOIN CITAS c ON m.id\_medico = c.id\_medico JOIN FACTURAS f ON c.id\_cita = f.id\_cita GROUP BY m.id\_medico, m.nombre; |

# Resultados y Verificación

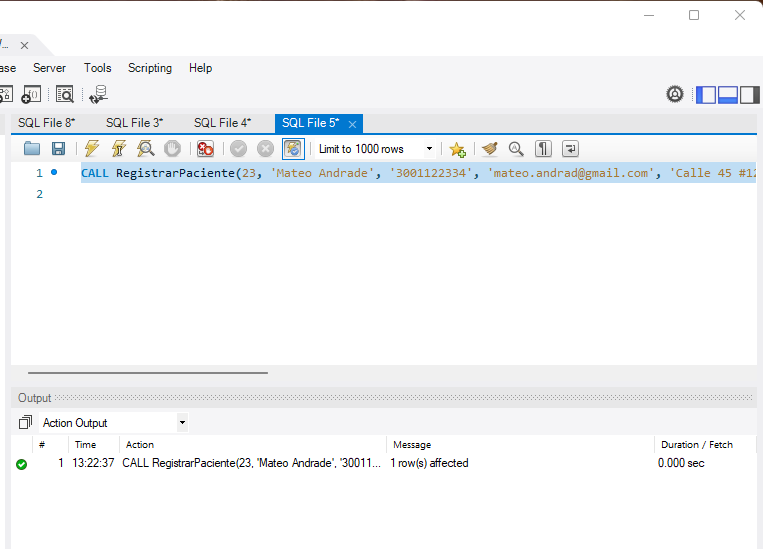
Caso 1: Obtener citas por paciente



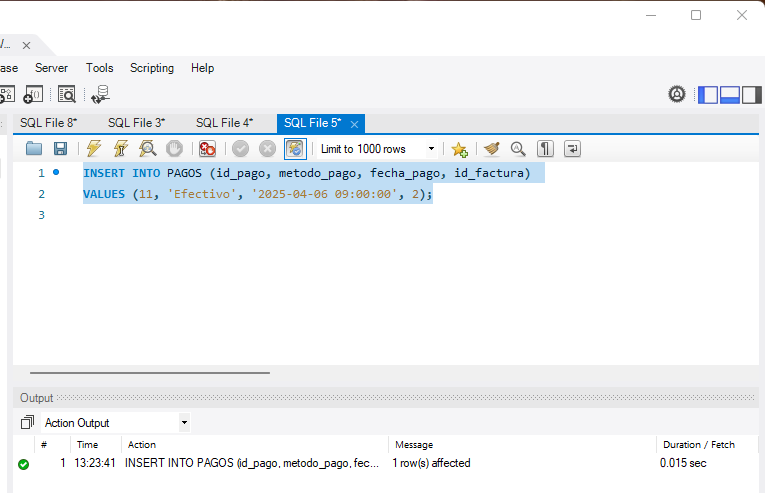
Caso 2: Evitar eliminación de paciente con citas activas



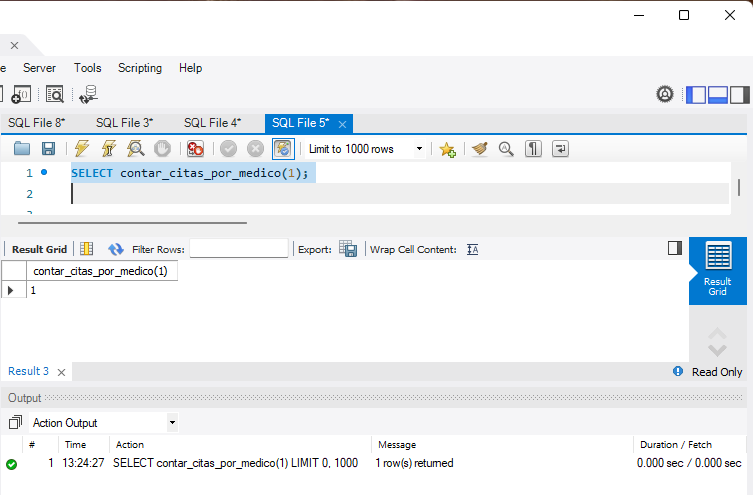
Caso 3: Registrar nuevo paciente



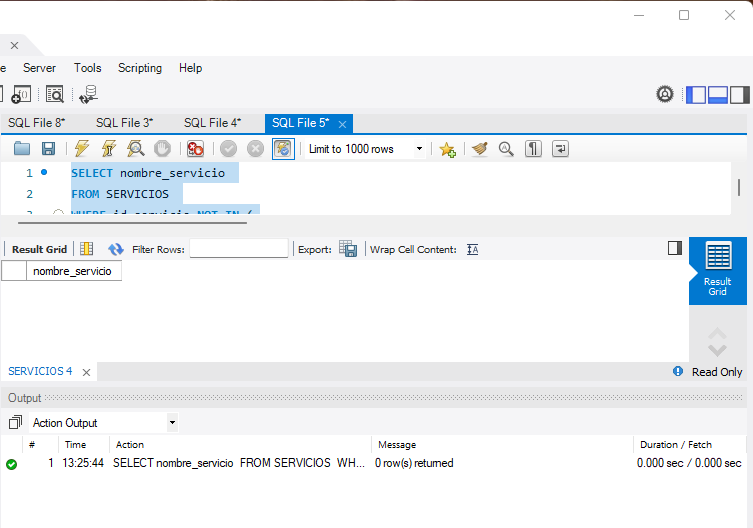
Caso 4: Actualización automática del estado de factura al registrar un pago



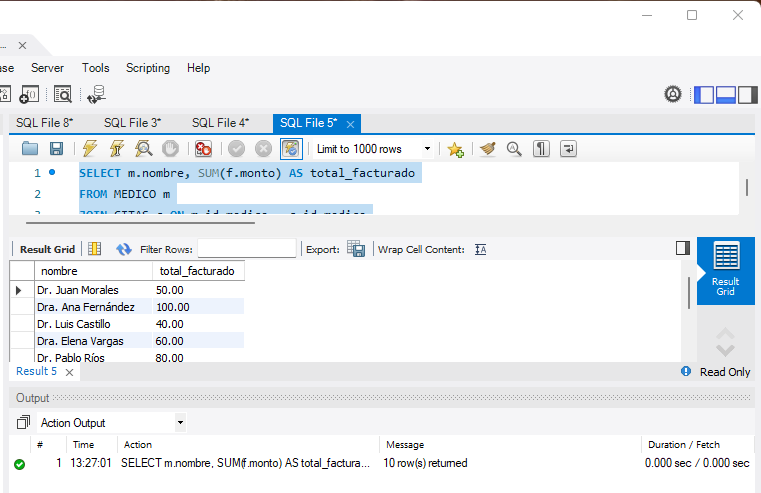
Caso 5: Usar función para contar citas de un médico



Caso 6: Identificar servicios no utilizados



Caso 7: Verificar total facturado por médico



# Conclusiones y Mejoras Futuras

El proyecto permitió crear una base de datos que organiza bien la información de un centro médico, se logró gestionar pacientes, médicos, citas, servicios y pagos de forma clara, también se añadieron funciones y procedimientos que ayudan a automatizar procesos.

Entre las mejoras que se podrían hacer más adelante están:

* Agregar más datos de prueba.
* Hacer consultas más complejas o detalladas.
* Añadir más validaciones simples, como evitar citas en horarios repetidos.
* Crear vistas que resuman información útil, como pacientes frecuentes o servicios más usados.

# Enlace al Repositorio en GitHub

https://github.com/AngelyCastellanos/ProyectoBBDD\_Angely.git