Dental Clinic Proyecto

Base de datos

<Rocío Baena Pérez>

<1ºDAW>

<IES Alixar>

19/03/2024

ÍNDICE

I. Introducción de la temática del proyecto

Este proyecto tiene como objetivo crear una base de datos que ayude a organizar y manejar mejor toda la información de una clínica dental. La idea es tener un sistema que permita guardar datos importantes sobre el personal, los pacientes, las citas y los tratamientos.

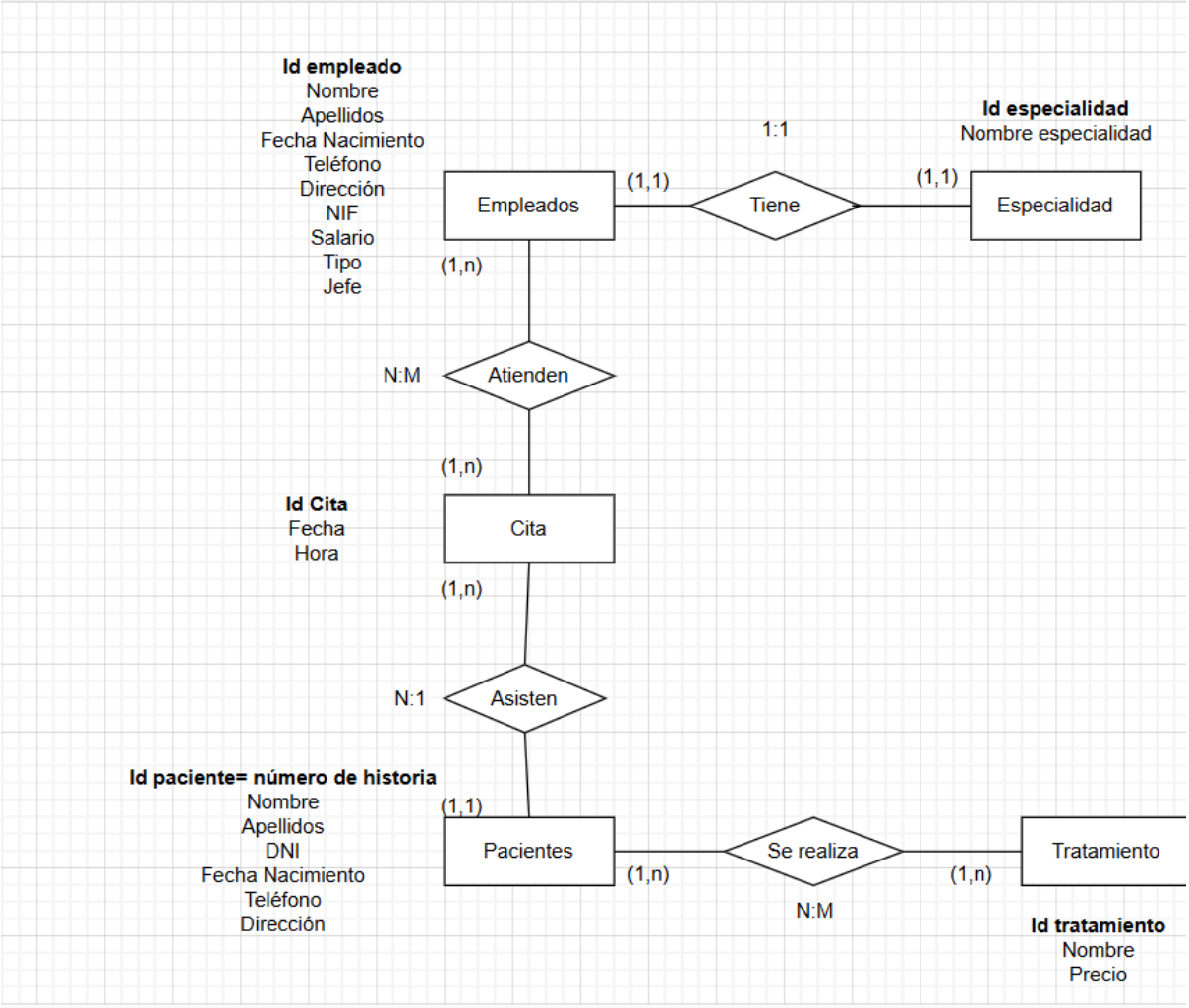
En la clínica trabajan odontólogos con diferentes especialidades, y uno de ellos es el jefe encargado de supervisar. Además, hay otros empleados como recepcionistas, comerciales y auxiliares que apoyan en el día a día.

Los pacientes tienen un número único de ficha-historial clínico donde se registra su información personal, tratamientos realizados y el empleado que los atendió. Un paciente puede ser atendido por uno o más odontólogos según las especialidades que necesiten.

Durante el desarrollo del proyecto, he seguido diferentes etapas, desde el diseño del modelo de datos hasta su implementación en un sistema de gestión de bases de datos. Además, se ha realizado la carga de datos, la creación de consultas para obtener información relevante y la realización de procedimientos,funciones, vistas y triggers.

Finalmente, la base de datos ha sido desplegada en una instancia de AWS, permitiendo su acceso remoto.

II. Modelo Entidad Relación



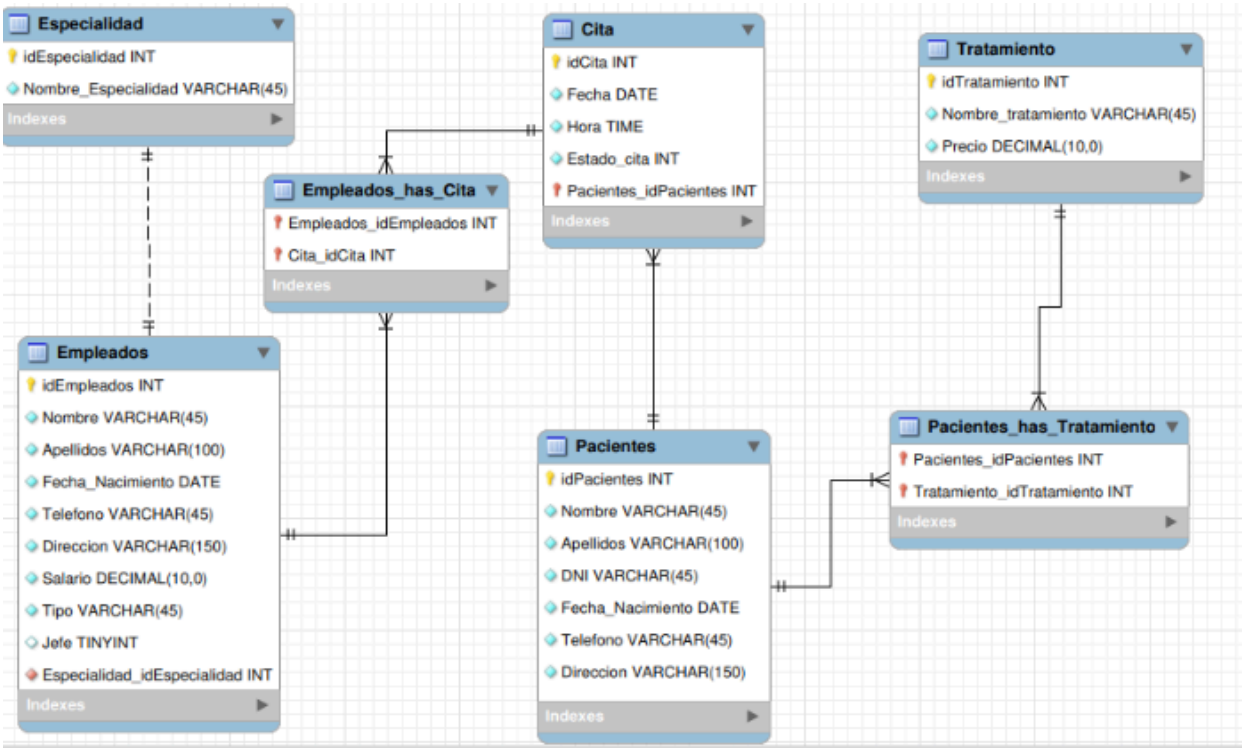
En mi modelo entidad relación podemos ver las siguientes entidades:

- Especialidad: Es la especialidad que se le asigna al empleado. Los datos que registra es id_especialidad y su nombre.
- Empleados: Son los trabajadores de la clínica, con datos como nombre, apellidos, NIF, salario y tipo de puesto.
- Pacientes: Son las personas que acuden a la clínica. Cada paciente tiene un número de historia clínica único y datos personales como DNI, dirección y teléfono.
- Citas: Se registran para gestionar las consultas de los pacientes. Cada cita tiene una fecha y hora, y puede ser atendida por uno o varios empleados.

- Tratamientos: Son los tratamientos que se realizan los pacientes en las citas. Cada tratamiento tiene un nombre y un precio.

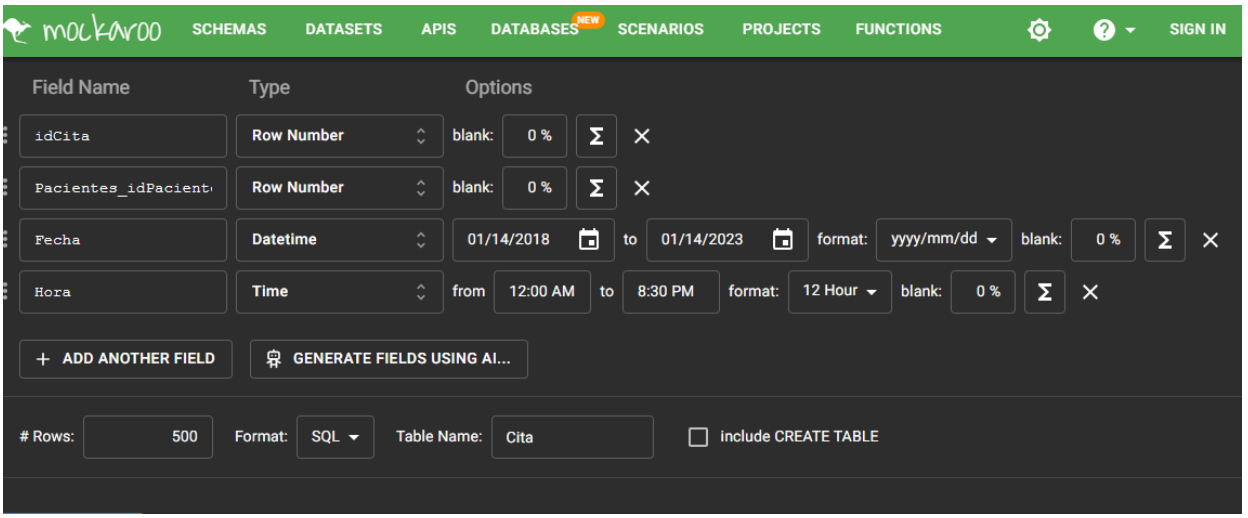
Como vemos tengo 5 entidades (Empleados, especialidad, cita, pacientes y tratamiento).
Como existen dos relaciones N:M al final se me generarán 7 tablas como veremos en el modelo relacional del siguiente apartado.

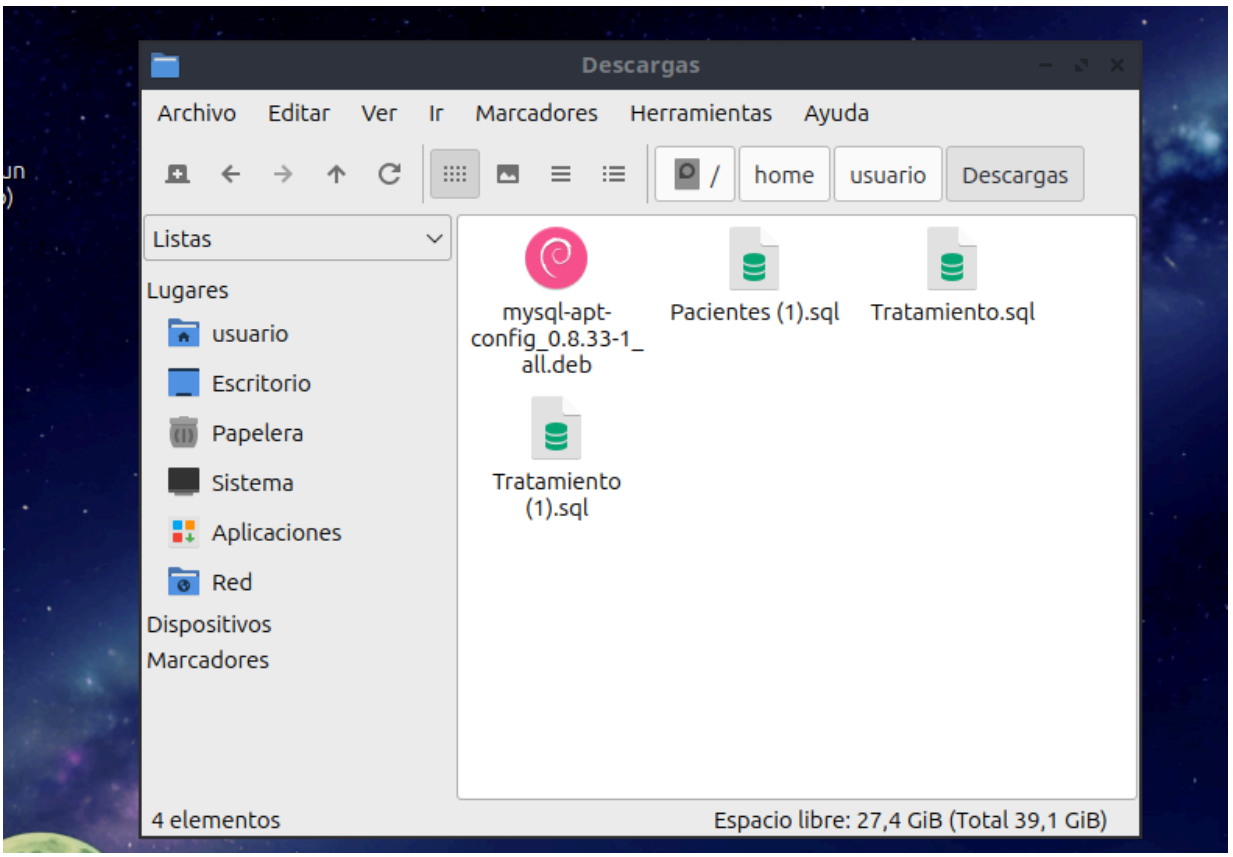
III. Modelo Relacional



IV. Carga masiva

Una vez generadas las tablas en MySql Workbench nos vamos a File en la barra de arriba, le damos a la opción /Export/Forward Engeneering SQL create script.
Ya con esto conseguimos nuestro script que lo pasamos al Dbeaver y generamos nuestras tablas con todos sus atributos para ir haciendo la carga de cada una de ellas.
La carga masiva la he realizado con la página [Mockaroo - Random Data Generator and API Mocking Tool | JSON …](). También tuve que ayudarme de otra ayuda debido a que ciertos datos no me lo generaba esta página, como pueden ser los tratamientos odontológicos o especialidades.





V. Consultas

- Pacientes que han recibido el tratamiento más caro.

```sql
-- Pacientes que han recibido el tratamiento más caro.
select t.Nombre_tratamiento, t.Precio, p.Nombre, p.Apellidos
from Tratamiento t
   inner join Pacientes_has_Tratamiento pt
   on t.idTratamiento = pt.Tratamiento_idTratamiento
   inner join Pacientes p
   on pt.Pacientes_idPacientes = p.idPacientes
where t.Precio =
    (select max(Precio)
    from Tratamiento);
```

select t.Nombre_tratamien | Enter a SQL expression to filter results (use Ctrl+Spac

	Nombre tratamiento	Precio	Nombre	Apellidos
1	Implantes All-on-4	4.500	Carlos	Martínez Fernández
2	Implantes All-on-4	4.500	Mónica	Torres Álvarez
3	Implantes All-on-4	4.500	Elena	Sánchez Díaz
4	Implantes All-on-4	4.500	José Antonio	Vázquez Rodríguez
5	Implantes All-on-4	4.500	Laura	Martínez López
6	Implantes All-on-4	4.500	Beatriz	Torres Jiménez
7	Implantes All-on-4	4.500	Antonio	Sánchez Álvarez

Grilla | Texto

	Nombre tratamiento	Precio	Nombre	Apellidos
8	Implantes All-on-4	4.500	Cristina	Torres González
9	Implantes All-on-4	4.500	María del Carmen	Torres Ramírez
10	Implantes All-on-4	4.500	Carlos	González Sánchez
11	Implantes All-on-4	4.500	María Teresa	Martínez Álvarez
12	Implantes All-on-4	4.500	Carlos Javier	Moreno González
13	Implantes All-on-4	4.500	Antonio	Sánchez Pérez
14	Implantes All-on-4	4.500	José Francisco	González Torres

- Total ingresos de tratamientos por empleados ordenados de mayor a menor ingreso.

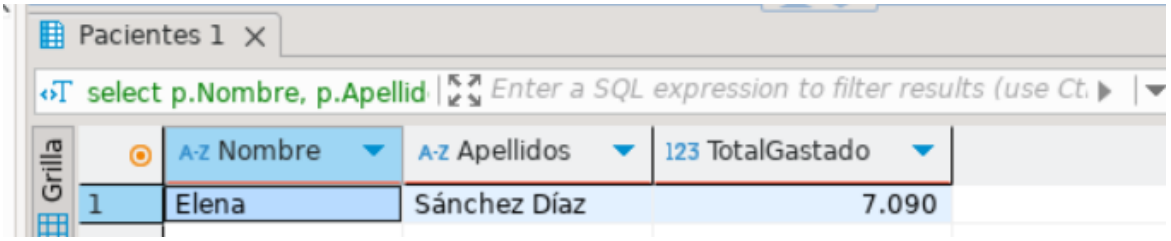
```sql
-- Total ingresos de tratamiento por empleados ordenados de mayor a menor ingreso.
select e.Nombre, e.Apellidos, sum(t.Precio) as Total_Ingresos
from Empleados e
   inner join Empleados_has_Cita ec on e.idEmpleados = ec.Empleados_idEmpleados
   inner join Cita c
   on ec.Cita_idCita = c.idCita
   inner join Pacientes_has_Tratamiento pt
   on c.Pacientes_idPacientes = pt.Pacientes_idPacientes
   inner join Tratamiento t
   on pt.Tratamiento_idTratamiento = t.idTratamiento
group by e.idEmpleados
order by Total_Ingresos desc;
```

Grilla | Texto

	Nombre	Apellidos	Total Ingresos
1	Lucía	Fernández Sánchez	218.590
2	Sergio	Hernández Moreno	165.140
3	Raquel	Martín Torres	162.040
4	Eduardo	Hernández Rodríguez	142.040
5	Sandra	Sánchez Pérez	137.580
6	Javier	Santos Romero	123.905
7	Ana	López Pérez	116.790

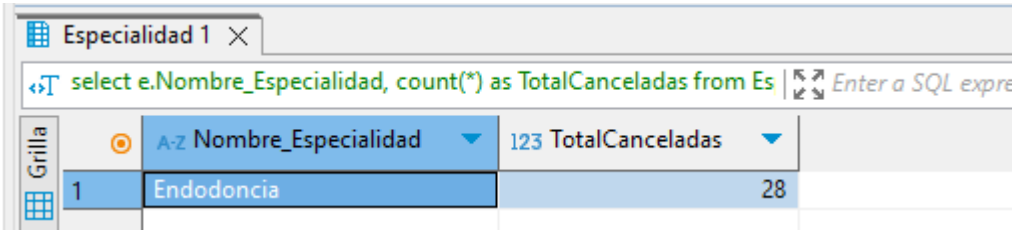
- Paciente que más ha gastado.

```sql
-- Paciente que más ha gastado
select p.Nombre, p.Apellidos, sum(t.Precio) as TotalGastado
from Pacientes p
   inner join Pacientes_has_Tratamiento pht
   on p.idPacientes = pht.Pacientes_idPacientes
   inner join Tratamiento t
   on pht.Tratamiento_idTratamiento = t.idTratamiento
group by p.idPacientes, p.Nombre, p.Apellidos
having sum(t.Precio) >= all (
   select sum(t2.Precio)
   from Pacientes_has_Tratamiento pht2
      inner join Tratamiento t2
      on pht2.Tratamiento_idTratamiento = t2.idTratamiento
   group by pht2.Pacientes_idPacientes);
```



- Especialidad con más citas canceladas

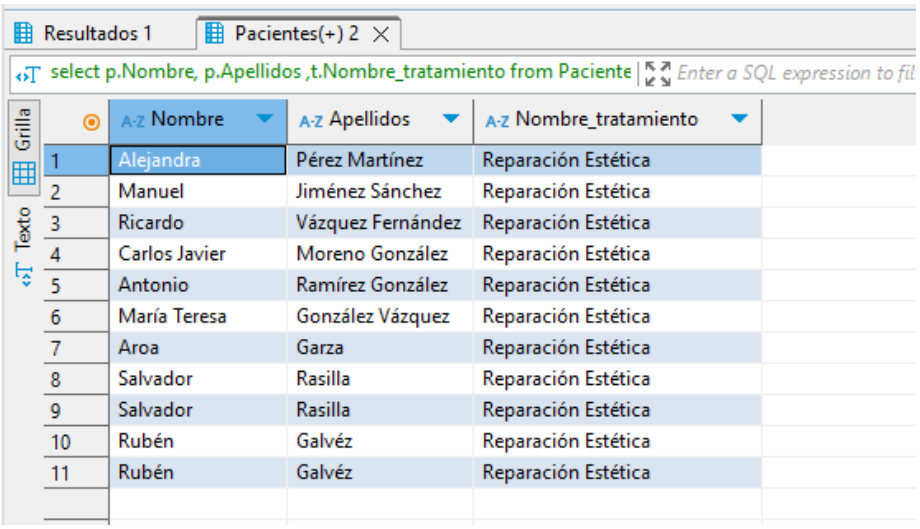
```sql
-- Especialidad con más citas canceladas.
select e.Nombre_Especialidad, count(*) as TotalCanceladas
from Especialidad e
    inner join Empleados e2
    on e.idEspecialidad = e2.Especialidad_idEspecialidad
    inner join Empleados_has_Cita ehc
    on e2.idEmpleados = ehc.Empleados_idEmpleados
    inner join Cita c
    on ehc.Cita_idCita = c.idCita
where  c.Estado_cita = 'Cancelada'
group by e.idEspecialidad, e.Nombre_Especialidad
having count(*) >= all (
        select count(*)
        from Cita c2
            inner join Empleados_has_Cita ehc2
            on c2.idCita = ehc2.Cita_idCita
            inner join Empleados e3
            on ehc2.Empleados_idEmpleados = e3.idEmpleados
            inner join Especialidad e4
            on e3.Especialidad_idEspecialidad = e4.idEspecialidad
        where c2.Estado_cita = 'Cancelada'
        group by e4.idEspecialidad);
```



- Pacientes que se han realizado el tratamiento más realizado en las fechas 1 de febrero 2025 al 28 febrero 2025.

```sql
-- Pacientes que se han realizado el tratamiento más realizado entre las
-- fechas 1 de febrero 2025 y 28 febrero 2025.
select p.Nombre, p.Apellidos ,t.Nombre_tratamiento
from Pacientes p
  inner join Cita c
  on p.idPacientes =c.Pacientes_idPacientes
  inner join Pacientes_has_Tratamiento pht
  on c.Pacientes_idPacientes =pht.Pacientes_idPacientes
  inner join Tratamiento t
  on pht.Tratamiento_idTratamiento =t.idTratamiento
where pht.Tratamiento_idTratamiento >=all (
select t.idTratamiento
```

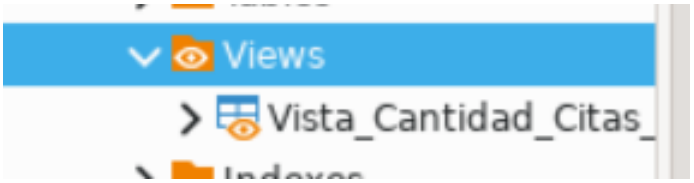
```sql
where pht.Tratamiento_idTratamiento >=all (
select t.idTratamiento
from Cita c
    inner join Pacientes p
    on c.Pacientes_idPacientes = p.idPacientes
    inner join Pacientes_has_Tratamiento pht
    on p.idPacientes =pht.Pacientes_idPacientes
    inner join Tratamiento t
    on pht.Tratamiento_idTratamiento = t.idTratamiento
where c.Fecha between '2025-02-01' and '2025-02-28'
group by t.idTratamiento , t.Nombre_tratamiento );
```



VI. View

- View cantidad de citas atendidas por empleados, ordenadas de mayor a menor cantidad.

```sql
-- View cantidad de citas atendidas por empleado,ordenadas de mayor a menor cantidad.
create view Vista_Cantidad_Citas_Empleados as
select e.Nombre, e.Apellidos, count(c.idCita) as Numero_Citas
from Empleados e
   inner join Empleados_has_Cita ec
   on e.idEmpleados = ec.Empleados_idEmpleados
   inner join Cita c
   on ec.Cita_idCita = c.idCita
group by e.idEmpleados
order by Numero_Citas desc;

select * from Vista_Cantidad_Citas_Empleados;
```

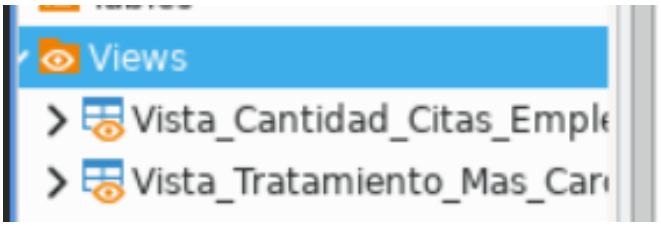


select * from Vista_Cantidad | Enter a SQL expression to filter results (use Ctrl+Spac

	Nombre	Apellidos	Numero Citas
1	Lucía	Fernández Sánchez	132
2	Eduardo	Hernández Rodríguez	113
3	Sandra	Sánchez Pérez	113
4	Raquel	Martín Torres	113
5	Sergio	Hernández Moreno	113
6	Ángel	Martínez Rodríguez	103
7	Manuel	Rodríguez Torres	103

- View pacientes que han recibido el tratamiento más caro.

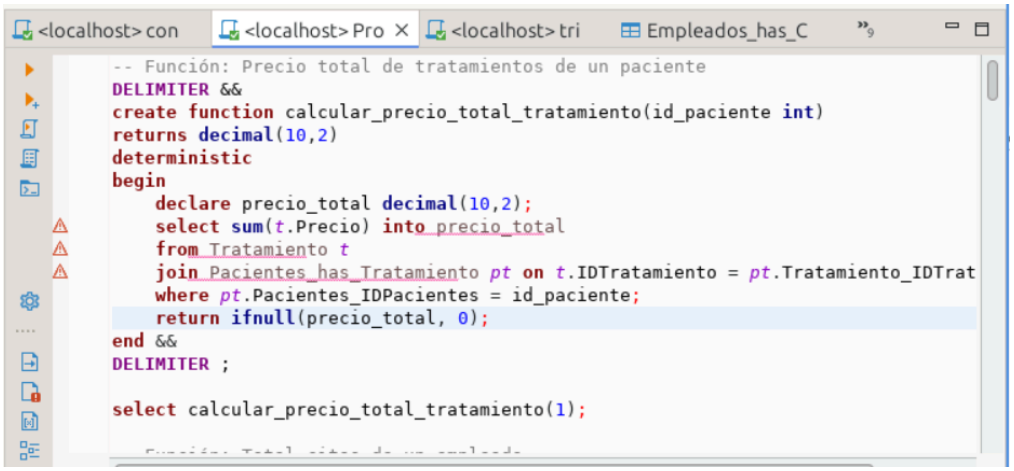
```
-- View pacientes que han recibido el tratamiento más caro.
create view Vista_Tratamiento_Mas_Caro as
select t.Nombre_tratamiento, t.Precio, p.Nombre, p.Apellidos
from Tratamiento t
   inner join Pacientes_has_Tratamiento pt
   on t.idTratamiento = pt.Tratamiento_idTratamiento
   inner join Pacientes p
   on pt.Pacientes_idPacientes = p.idPacientes
where t.Precio =
    (select max(Precio)
    from Tratamiento);

select * from Vista_Tratamiento_Mas_Caro;
```

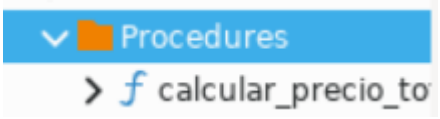


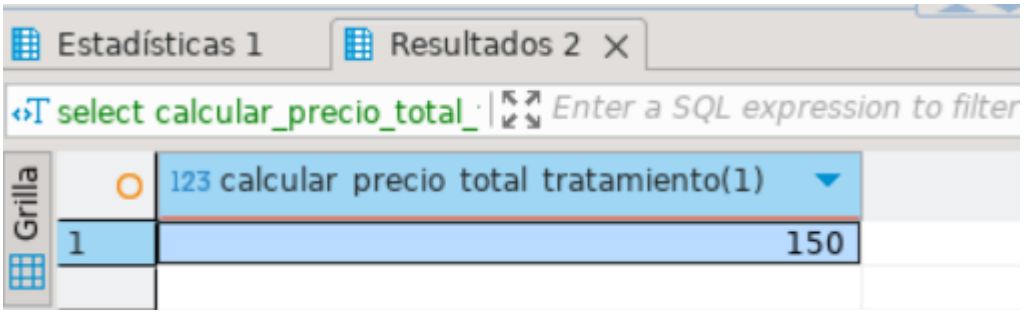
	Nombre tratamiento	Precio	Nombre	Apellidos
1	Implantes All-on-4	4.500	Carlos	Martínez Fernández
2	Implantes All-on-4	4.500	Mónica	Torres Álvarez
3	Implantes All-on-4	4.500	Elena	Sánchez Díaz
4	Implantes All-on-4	4.500	José Antonio	Vázquez Rodríguez
5	Implantes All-on-4	4.500	Laura	Martínez López
6	Implantes All-on-4	4.500	Beatriz	Torres Jiménez
7	Implantes All-on-4	4.500	Antonio	Sánchez Álvarez

VII. Funciones y procedimientos.

- Función Calcula precio total tratamientos por paciente.

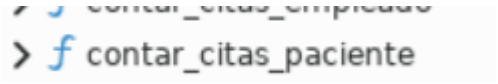


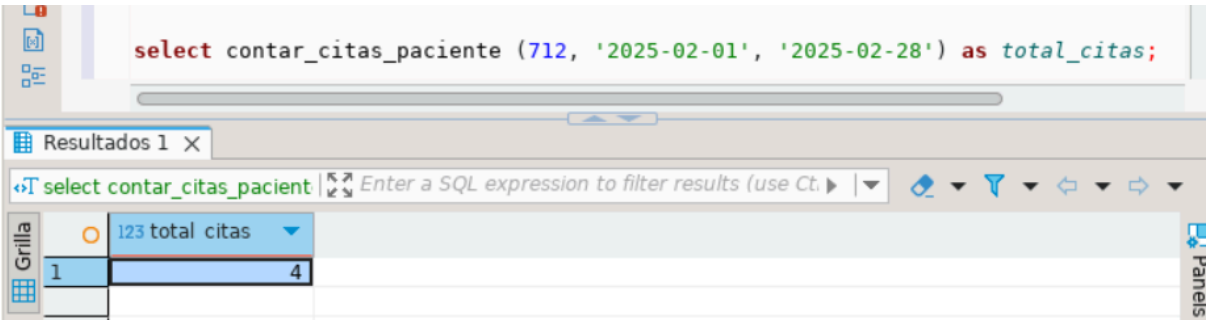




- Función :Cantidad de citas de un paciente en un periodo de tiempo.

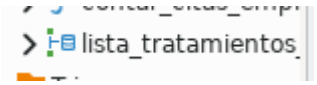
```
-- Función: Citas de un paciente en un periodo de tiempo
DELIMITER &&

create function contar_citas_paciente(id_paciente int, fecha_inicio date, fecha_
returns int
deterministic
begin
    declare total_citas INT default 0;

    select count(*) into total_citas
    from Cita c
    where c.Pacientes_idPacientes = id_paciente
    and c.Fecha between fecha_inicio and fecha_fin;

    return total_citas;
end &&

DELIMITER ;
```



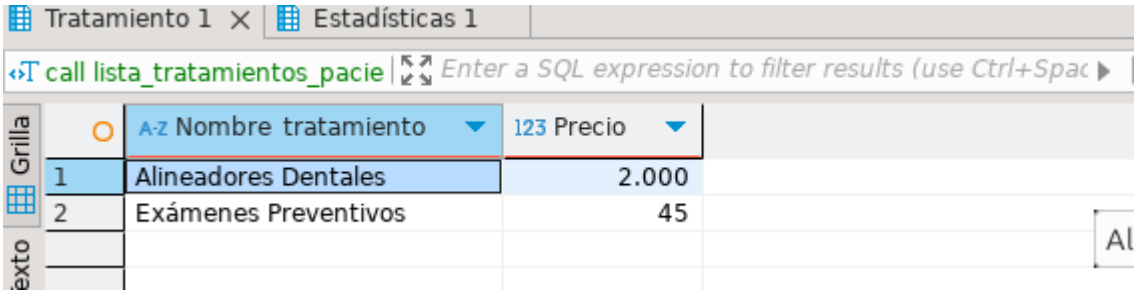


- Procedimiento: Lista de tratamientos con su precio y nombre, de un paciente.

```
-- Lista de tratamientos de los pacientes
DELIMITER &&

create procedure lista_tratamientos_paciente(in id_paciente int)
begin
    select t.Nombre_tratamiento, t.Precio
    from Tratamiento t
    inner join Pacientes_has_Tratamiento pt
    on t.idTratamiento = pt.Tratamiento_idTratamiento
    where pt.Pacientes_idPacientes = id_paciente;
end &&
DELIMITER ;
```

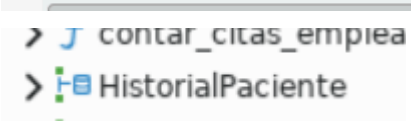


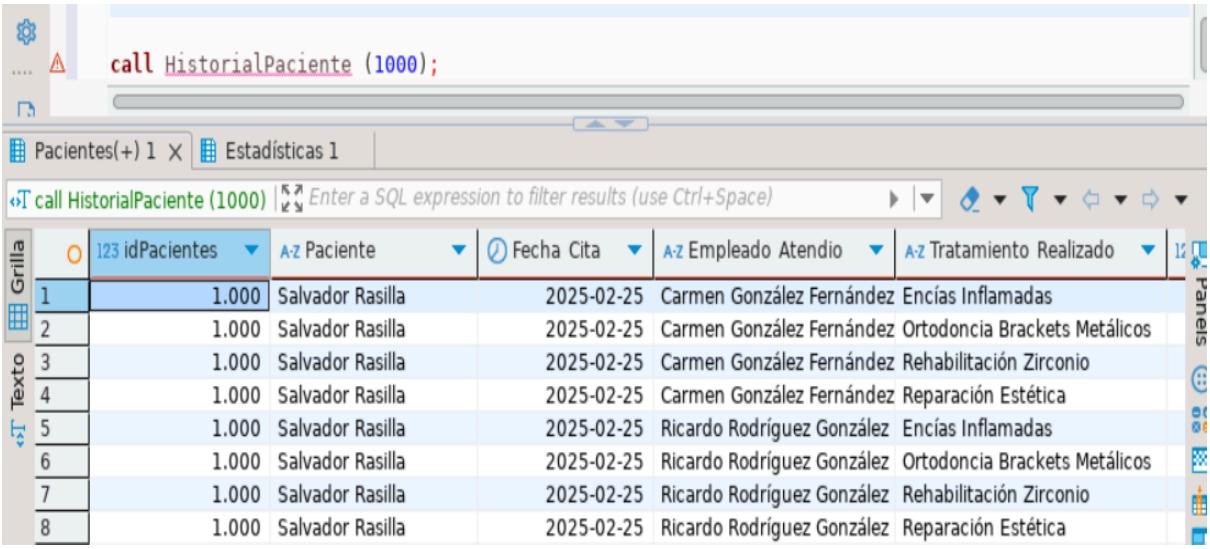

```
call lista_tratamientos_paciente (780);
```



- Procedimiento: Historial paciente

```sql
-- tratamiento,empleado que le atendio y gasto.
DELIMITER &&

create procedure HistorialPaciente(in paciente_id int)
begin
    select
    p.idPacientes,
    concat(p.Nombre, ' ', p.Apellidos) as Paciente,
    c.Fecha as Fecha_Cita,
    concat(e.Nombre, ' ', e.Apellidos) as Empleado_Atendio,
    t.Nombre_tratamiento as Tratamiento_Realizado,
    t.Precio as Gasto
from Pacientes p
   inner join Cita c
   on p.idPacientes = c.Pacientes_idPacientes
   inner join Empleados_has_Cita ec
   on c.idCita = ec.Cita_idCita
   inner join Empleados e
   on ec.Empleados_idEmpleados = e.idEmpleados
   inner join Pacientes_has_Tratamiento pt
   on p.idPacientes = pt.Pacientes_idPacientes
   inner join Tratamiento t
   on pt.Tratamiento_idTratamiento = t.idTratamiento
where p.idPacientes = paciente_id
order by c.Fecha desc;

end &&

DELIMITER ;
```





	idPacientes	Paciente	Fecha Cita	Empleado Atendio	Tratamiento Realizado
1	1.000	Salvador Rasilla	2025-02-25	Carmen González Fernández	Encías Inflamadas
2	1.000	Salvador Rasilla	2025-02-25	Carmen González Fernández	Ortodoncia Brackets Metálicos
3	1.000	Salvador Rasilla	2025-02-25	Carmen González Fernández	Rehabilitación Zirconio
4	1.000	Salvador Rasilla	2025-02-25	Carmen González Fernández	Reparación Estética
5	1.000	Salvador Rasilla	2025-02-25	Ricardo Rodríguez González	Encías Inflamadas
6	1.000	Salvador Rasilla	2025-02-25	Ricardo Rodríguez González	Ortodoncia Brackets Metálicos
7	1.000	Salvador Rasilla	2025-02-25	Ricardo Rodríguez González	Rehabilitación Zirconio
8	1.000	Salvador Rasilla	2025-02-25	Ricardo Rodríguez González	Reparación Estética

- Procedimiento con uso de la función: Precio total tratamientos de un paciente

```
-- Procedimiento con función anterior: Precio total tratamientos de un paciente.
DELIMITER &&

create procedure calcular_precio_total_tratamiento(in id_paciente int)
begin
    select p.Nombre, p.Apellidos, calcular_precio_total_tratamiento(id_paciente)
    from Pacientes p
    inner join Pacientes_has_Tratamiento pt
    on p.idPacientes = pt.Pacientes_idPacientes
    inner join Tratamiento t
    on pt.Tratamiento_idTratamiento = t.idTratamiento
where p.idPacientes = id_paciente;

end &&
DELIMITER ;
```

```
call calcular_precio_total_tratamientos (485);
```

Pacientes 1 | Estadísticas 1

call calcular_precio_total_tra | Enter a SQL expression to filter results (use

Grilla

	Nombre	Apellidos	Total precio
1	José Luis	Martínez Ramírez	2.400

VIII. Triggers

- Trigger: Actualizar gastos de un paciente al añadir un tratamiento.

```
-- Trigger: Actualizar el gasto de un paciente al añadir un tratamiento.
DELIMITER &&

create trigger mensaje_total_gasto
after insert on Pacientes_has_Tratamiento
for each row
begin
    declare total_gasto decimal(10,2);
    declare mensaje varchar(255);

    select sum(t.Precio)
    into total_gasto
    from Tratamiento t
       inner join Pacientes_has_Tratamiento pt
       on t.idTratamiento = pt.Tratamiento_idTratamiento
    where pt.Pacientes_idPacientes = new.Pacientes_idPacientes;


    set mensaje= concat('El paciente ha gastado un total de ', total_gasto, ' eu

    signal sqlstate '45000'
    set MESSAGE_TEXT = mensaje;
end &&

DELIMITER ;
```

Probamos con una consulta un paciente para ver su total actualmente.

```
-- Probamos
select p.idPacientes, p.Nombre, p.Apellidos, sum(t.Precio) as Total_Gastos
from Pacientes p
  inner join Pacientes_has_Tratamiento pt
  on p.idPacientes = pt.Pacientes_idPacientes
   inner join Tratamiento t
   on pt.Tratamiento_idTratamiento = t.idTratamiento
where p.idPacientes = 78
group by p.idPacientes, p.Nombre, p.Apellidos;
```

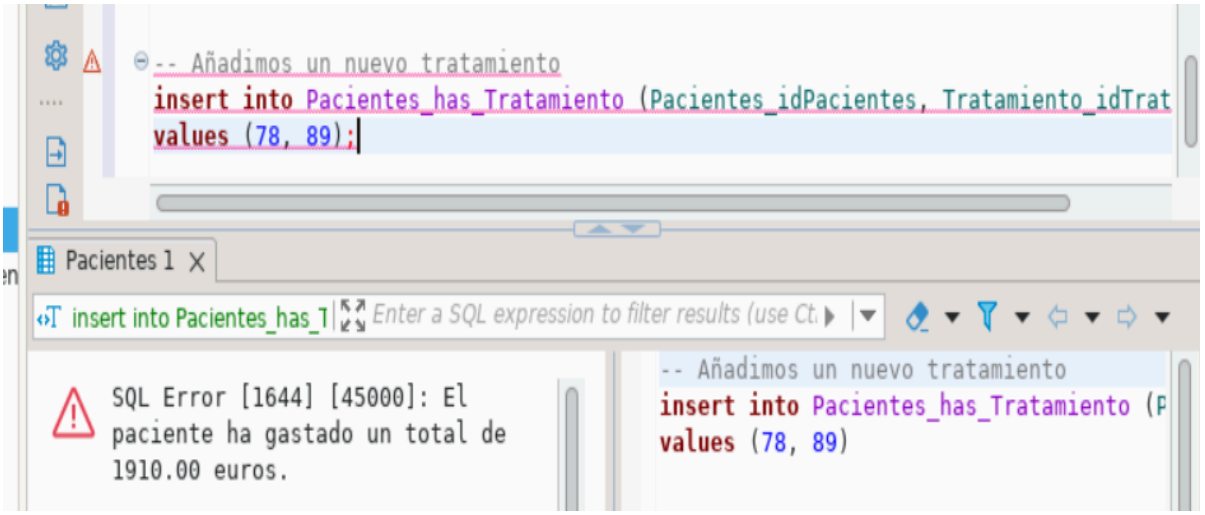
Pacientes 1

select p.idPacientes, p.Nor | Enter a SQL expression to filter results (use Ct

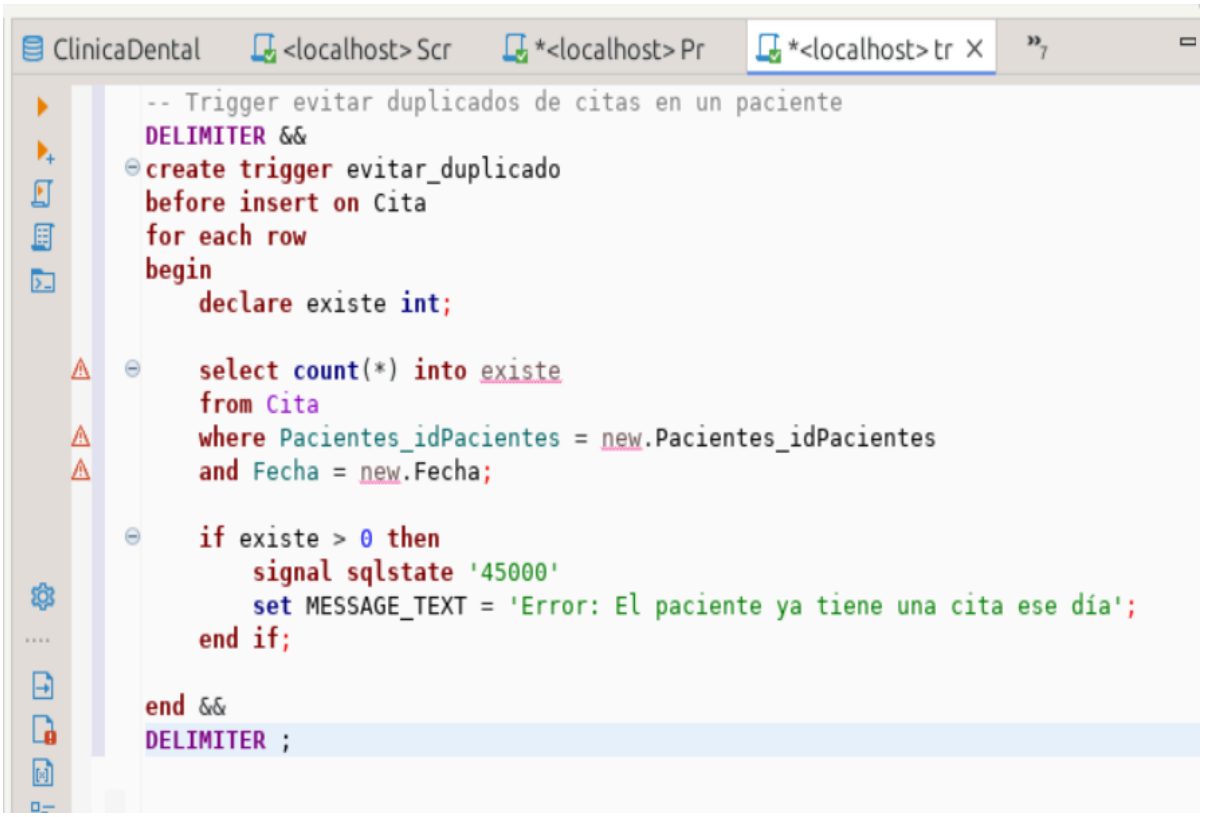
Grilla

	idPacientes	Nombre	Apellidos	Total Gastos
1	78	Susana	López Ruiz	1.460

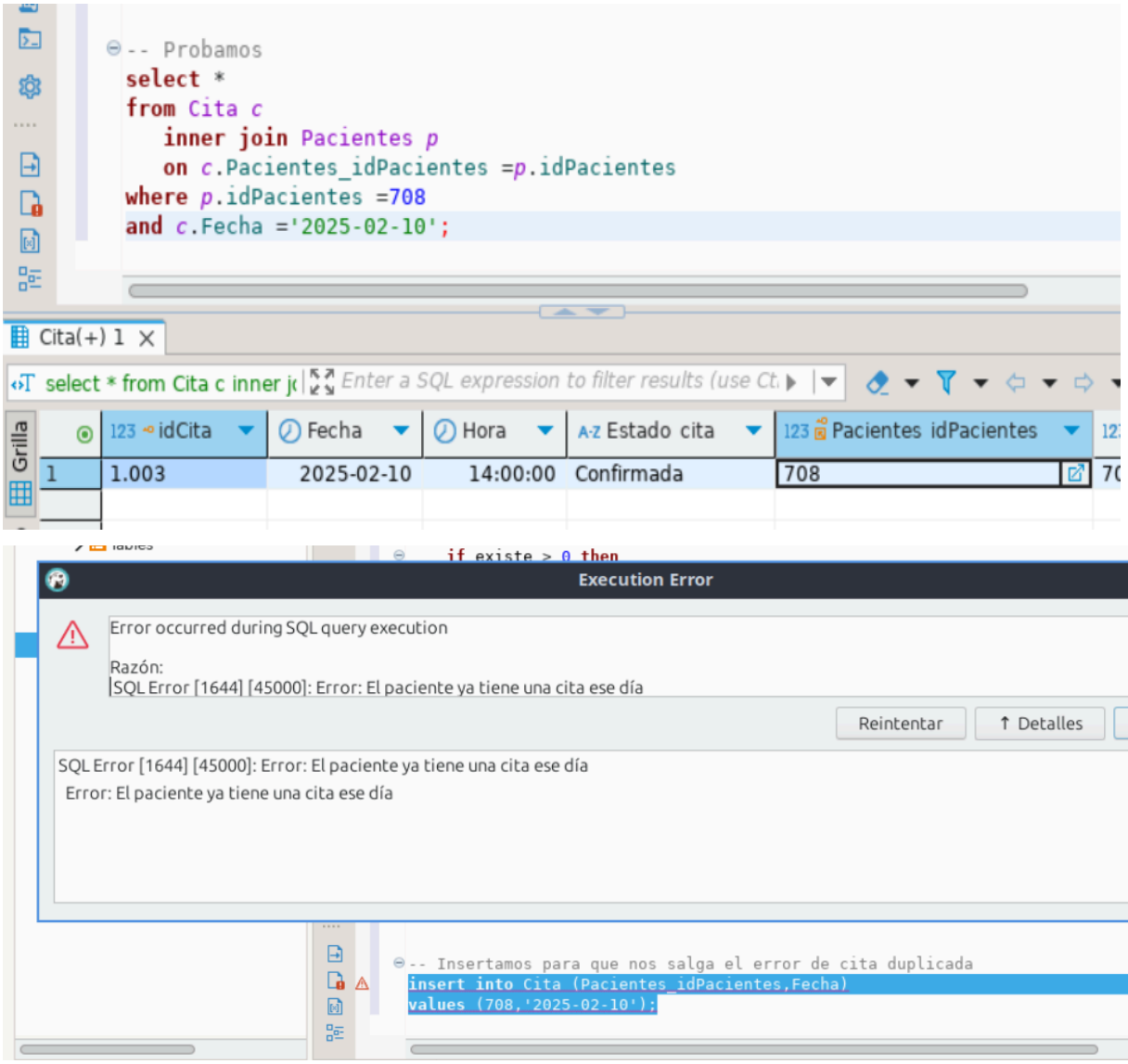
Insertamos un nuevo tratamiento al paciente y me salta el siguiente mensaje.



- Trigger para evitar duplicados de citas en un paciente.



Probamos que funcione.



IX. GitHub

En el siguiente repositorio de GitHub se encuentra todo el contenido del proyecto: [ProyectoClinicaDental]. Allí se pueden encontrar los siguientes archivos:

- **`miproyecto-schema.sql`**: Contiene el esquema de la base de datos.

- **`miproyecto-data.sql`**: Incluye los datos cargados en la base de datos.

- **`miproyecto-queries.sql`**: Recopila las consultas, vistas, funciones, procedimientos y triggers implementados.

- **`Proyecto_BaseDatos_RocioBaena.pdf`**: Documento con la información detallada del proyecto."

X. AWS

El proyecto ha sido desplegado en una instancia EC2 de AWS llamada MiproyectoDB-RB , donde se ha instalado MySQL Server y se ha cargado la base de datos. Para acceder a la base de datos desde DBeaver, se debe conectar a la siguiente dirección IP: `44.207.219.136`.

XI. Conclusión: Valoración personal de lo aprendido

Este proyecto ha sido un gran aprendizaje para mí. Al principio en la primera entrega me dió muchos quebraderos de cabeza sobre todo por la creación de datos para las tablas, pero ya en esta segunda entrega ha ido todo más rodado y agilizado.En definitiva trabajar desde cero mi propia base de datos me ha permitido entender mejor cómo funciona una base de datos y cómo organizar la información de forma eficiente. Diseñar el modelo y trabajar con MySQL ha sido un reto, pero también una buena forma de poner en práctica lo que he aprendido en clase.

Uno de los aspectos que más me ha gustado ha sido crear consultas para obtener datos útiles sobre la clínica dental. Al principio, algunas me parecían complicadas, pero con práctica y paciencia he conseguido que funcionen bien.

Los que más me ha costado en este proyecto diría que era la realización de triggers,funciones y procedimientos y he necesitado sobretodo en trigger ayuda.

Ver como mi propia base de datos funcionaba y podía practicar con ella me ha dado una gran satisfacción.

Además, subir el proyecto a GitHub y desplegarlo en AWS ha sido una experiencia nueva para mí. Aunque ha supuesto un esfuerzo extra, creo que es algo que me servirá mucho en el futuro.

En general, estoy contenta con el resultado y con todo lo que he aprendido durante este proceso. Ha sido un desafío, pero también una buena oportunidad para mejorar mis habilidades y ganar confianza trabajando con bases de datos, como digo, creo que haciendo yo mi propia base de datos desde cero me enseña mucho más y aprendo más al conocerla al detalle.