



ESCUELA POLITÉCNICA NACIONAL

ESCUELA DE FORMACIÓN DE TECNÓLOGOS

Base de datos

ASIGNATURA:	Base de datos
PROFESOR:	Ing. Yadira Franco R
PERÍODO ACADÉMICO:	2024-A

Proyecto

Desarrollo de una Base de datos para una Clinica

ESTUDIANTE:
Loor Angelo,
Robalino Richard,
Tapias Marcos,
Villamil Angel

Link repositorio GITHUB: <https://github.com/VillamilA/ClinicaProyectoBaseDatos.git>

Link del Video: <https://youtu.be/uPcgBTiMdmU>

Proyecto de Base de Datos Grupal BD

Fecha de entrega : 4 de Agosto 2024

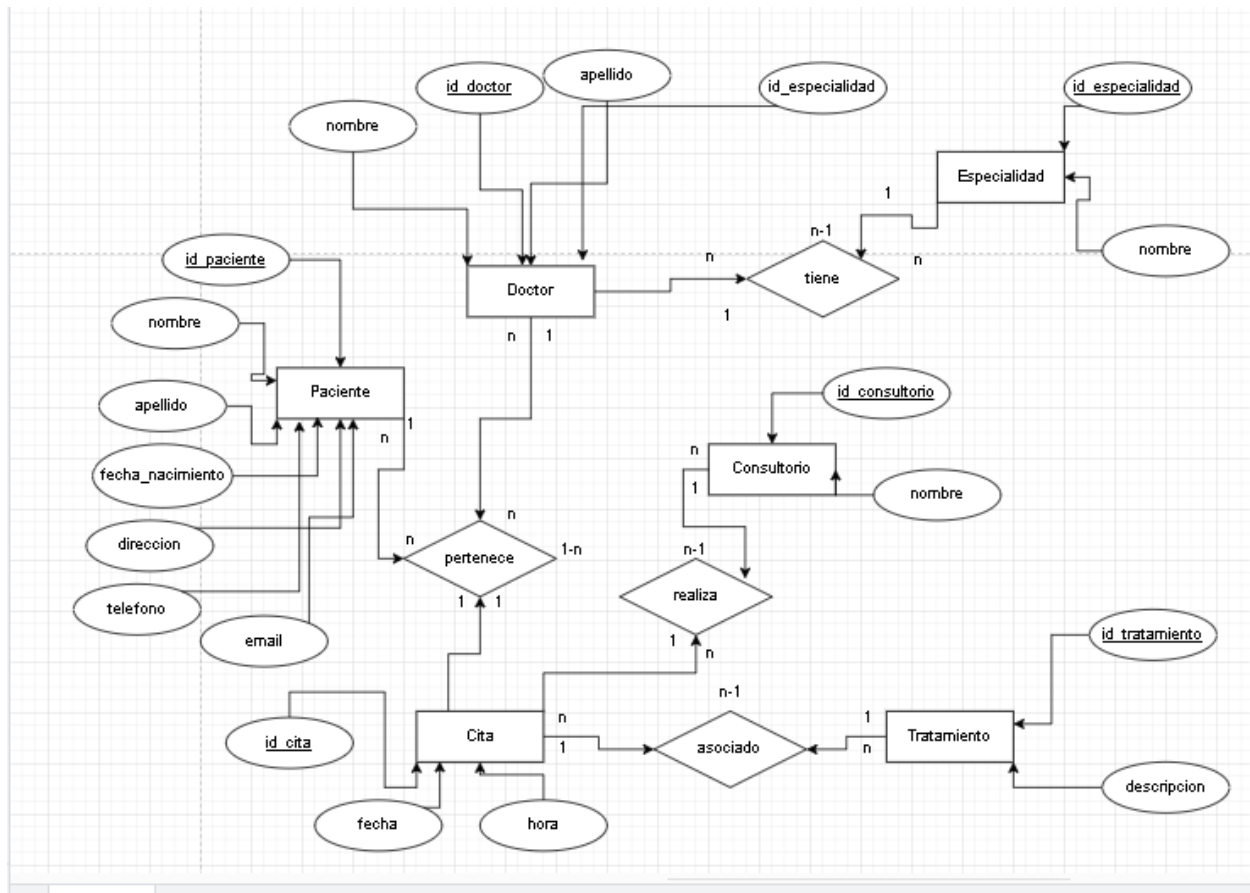
Instrucciones Generales

1. **Seleccionar una Temática:** Cada grupo debe seleccionar una de las siguientes temáticas para desarrollar su proyecto de base de datos. **En la parte inferior está indicado los temas a que grupo le corresponde.**
2. **Integrantes:** Los grupos deben estar formados por un número específico de integrantes según la temática seleccionada.
3. **Documentación y Entregables:**
 1. Modelado conceptual, lógico y físico. **(información sensible debe estar contraseñas con claves encriptadas), además se debe subir imagenes**
 2. Creación de scripts DDL y DML.
 3. Funciones cadena, calculados entre otros
 4. Consultas SQL y subconsultas.
 5. Creación y asignación de usuarios y privilegios.
 6. Implementación de vistas, procedimientos almacenados y triggers.
 7. Manejo de errores y pruebas de fallos.
 8. Generación de backups y procesos de importación/exportación.
 9. Aplicación de COMMIT y ROLLBACK.
 10. Joins
 11. Concurrencia
 12. Auditoría y seguridad.
 13. Procesos por lotes y funciones SQL.
 14. Monitoreo del consumo de recursos y registro de actividades de usuarios.

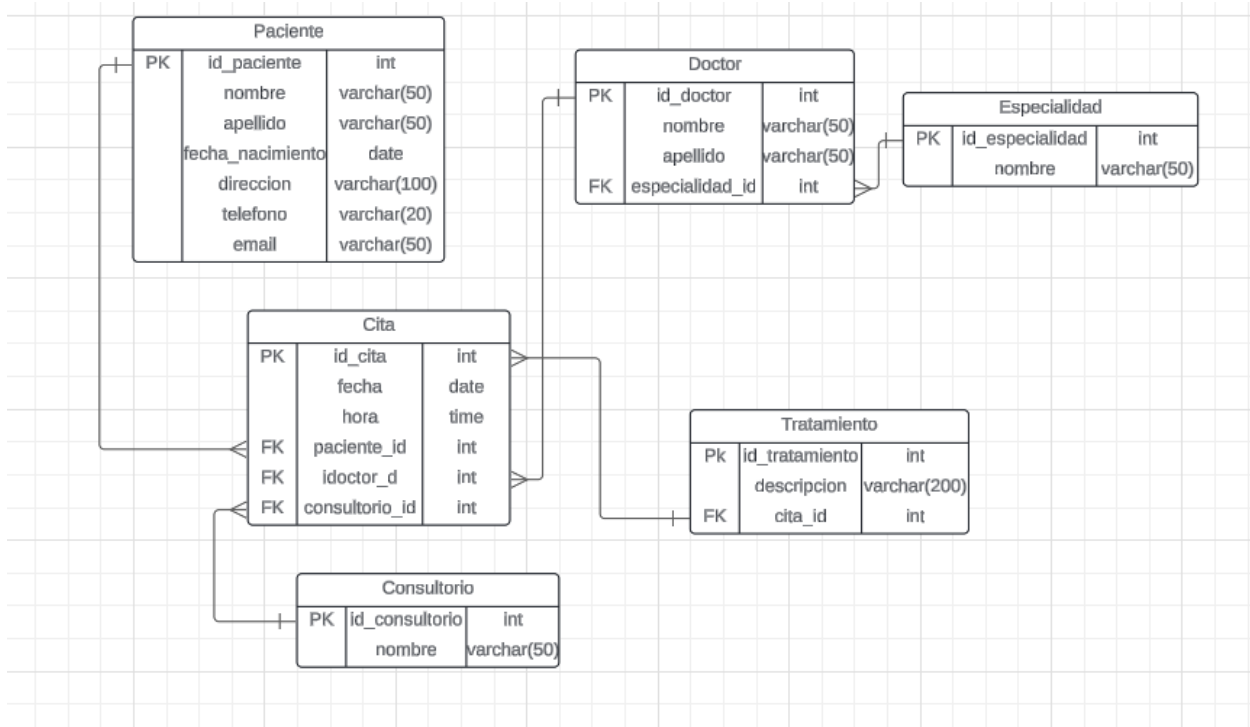
Componentes del Proyecto

1. Modelado conceptual, lógico y físico.

- **Modelado Conceptual:** Definición de entidades, relaciones y atributos; identificación de claves primarias y foráneas; normalización.



- **Modelado Lógico:** Transformación del modelo conceptual a tablas y columnas; aplicación de reglas de integridad; diseño de índices.



- **Modelado Físico:** Configuración de almacenamiento físico de datos; particionamiento de tablas; diseño de estructuras de almacenamiento eficientes.

```
create database clinica;
```

```
-- usar base de datos
```

```
use clinica;
```

```
-- crear tabla pacientes
```

```
create table pacientes (
```

```
    id_paciente int auto_increment,
```

```
    nombre varchar(50) not null,
```

```
    apellido varchar(50) not null,
```

```
    fecha_nacimiento date not null,
```

```
    direccion varchar(100) not null,
```

```
telefono varchar(20) not null,  
email varchar(50) not null,  
primary key (id_paciente)  
);
```

```
-- crear tabla doctores
```

```
create table doctores (  
    id_doctor int auto_increment,  
    nombre varchar(50) not null,  
    apellido varchar(50) not null,  
    especialidad_id int not null,  
    primary key (id_doctor),  
    foreign key (especialidad_id) references especialidades(id_especialidad)  
);
```

```
-- crear tabla especialidades
```

```
create table especialidades (  
    id_especialidad int auto_increment,  
    nombre varchar(50) not null,  
    primary key (id_especialidad)  
);
```

```
-- crear tabla consultorios
```

```
create table consultorios (  
    id_consultorio int auto_increment,  
    nombre varchar(50) not null,  
    primary key (id_consultorio)  
);
```

-- crear tabla citas

```
create table citas (  
    id_cita int auto_increment,  
    paciente_id int not null,  
    doctor_id int not null,  
    consultorio_id int not null,  
    fecha date not null,  
    hora time not null,  
    primary key (id_cita),  
    foreign key (paciente_id) references pacientes(id_paciente),  
    foreign key (doctor_id) references doctores(id_doctor),  
    foreign key (consultorio_id) references consultorios(id_consultorio)  
);
```

-- crear tabla tratamientos

```
create table tratamientos (  
    id_tratamiento int auto_increment,
```

```
cita_id int not null,  
  
descripcion varchar(200) not null,  
  
primary key (id_tratamiento),  
  
foreign key (cita_id) references citas(id_cita)  
);
```

2. Creación de scripts DDL y DML.

- **DDL (Data Definition Language):** Creación, modificación y eliminación de esquemas de base de datos; definición de tablas, índices, restricciones.

```
-- EJERCICIOS DDL  
-- para cambiar el nombre de la tabla doctores  
rename table doctores to medicos;  
-- para vaciar los datos de la tabla pacientes  
truncate table pacientes;  
-- para eliminar la tabla  
drop table pacientes;  
-- para modificar una tabla con alter  
alter table doctores add column fecha_contratacion date;  
-- en caso de querer eliminar la database clinica  
drop database clinica;  
-- para crear puede ser o tabla o la base de datos  
create database clinica;
```

- **DML (Data Manipulation Language):** Inserción, actualización y eliminación de datos; uso de transacciones.

```

-- EJERCICIOS DML
-- insertar un paciente
insert into pacientes (nombre, apellido, fecha_nacimiento, direccion, telefono, email) values
('ana', 'garcía', '1985-04-15', 'av. libertador 234', '555-1234', 'ana.garcia@example.com');
-- actualizar un paciente
update pacientes set direccion = 'av. nueva dirección 123' where id_paciente = 1;
-- eliminar un dato o varios
delete from pacientes where id_paciente = 1;
-- leer o seleccionar un dato o tabla
select * from pacientes where nombre = 'ana';

```

3. Funciones cadena, calculados entre otros.

- Funciones de cadena: CONCAT, SUBSTRING, CHARINDEX, REPLACE, etc.
 - CONCAT

```

383 -- Funciones de cadena
384 -- Concat
385 • SELECT CONCAT(nombre, ' ', apellido) AS nombre_completo FROM pacientes;

```

Result Grid | | Filter Rows: | Export: | Wrap Cell Content:

	nombre_completo
▶	Ana García
	Luis Pérez
	María Rodríguez
	Juan Martínez
	Carmen Hernández


```
386 • SELECT CONCAT('Doctor: ', nombre, ' ', apellido) AS detalle_doctor FROM doctores;
```

detalle_doctor
Doctor: carlos martínez
Doctor: gabriela mendoza
Doctor: jose ramírez
Doctor: maría fernández
Doctor: alejandro herrería
Doctor: paola mora
Doctor: nicolás suárez
Doctor: isabella león
Doctor: juan télez

```
387 • SELECT CONCAT('Consultorio: ', nombre) AS detalle_consultorio FROM consultorios;
```

detalle_consultorio
Consultorio: consultorio 1
Consultorio: consultorio 2
Consultorio: consultorio 3
Consultorio: consultorio 4
Consultorio: consultorio 5
Consultorio: consultorio 6
Consultorio: consultorio 7
Consultorio: consultorio 8
Consultorio: consultorio 9

▪ SUBSTRING

```
389 -- Substring
```

```
390 • SELECT SUBSTRING(nombre, 1, 3) AS iniciales_nombre FROM pacientes;
```

iniciales_nombre
Ana
Lui
Mar
Jua
Car
Jav
Lau
Ped
Sof

```
392 • SELECT SUBSTRING(email, LOCATE('@', email) + 1) AS dominio_email FROM pacientes;
```

dominio_email
example.com
example.com
example.com
example.com
example.com
example.com
example.com
example.com
example.com
example.com

```
393 • SELECT SUBSTRING(nombre, 1, 5) AS direccion_nombre FROM pacientes;
```

```
391 • SELECT SUBSTRING(direccion, 1, 10) AS direccion_corta FROM pacientes;
```

direccion_corta
Av. Libert
Calle Mayo
Plaza Cent
Calle del
Av. del No
Calle del
Calle de l
Calle del
Av. de la

Result 5 x

- CHARINDEX

```
394 -- Charindex
```

```
395 • SELECT LOCATE('Pérez', apellido) AS posicion_apellido FROM pacientes;
```

posicion_apellido
0
1
0
0
0
0

```
397 • SELECT LOCATE('@', email) AS posicion_arroba FROM pacientes;
```

<

Result Grid



Filter Rows:

Export:



Wrap Cell Content:



	posicion_arroba
▶	11
	11
	16
	14
	17
	13
	12
	14
	13
	15

Result 13 x

```
396 • SELECT LOCATE('Calle', direccion) AS posicion_direccion FROM pacientes;
```

<

Result Grid



Filter Rows:

Export:



Wrap Cell Content:



	posicion_direccion
▶	0
	1
	0
	1
	0
	1
	1
	1
	0
	1

- REPLACE

399 -- Replace

400 • `SELECT REPLACE(nombre, 'a', '@') AS nombre_modificado FROM pacientes;`

< [SQL Editor]

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

	nombre_modificado
▶	An@
	Luis
	M@rí@
	Ju@n
	C@rmen
	J@vier
	L@ur@
	Pedro
	Sofi@

Result 14 x

401 • `SELECT REPLACE(direccion, ' ', '_') AS direccion_modificada FROM pacientes;`

< [SQL Editor]

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

	direccion_modificada
▶	Av._Libertador_234
	Calle_Mayor_12
	Plaza_Central_5
	Calle_del_Sol_89
	Av._del_Norte_101
	Calle_del_Río_55
	Calle_de_la_Luna_78
	Calle_del_Mar_22
	Av._de_la_Paz_13

402 • `SELECT REPLACE(email, 'example', 'demo') AS email_modificado FROM pacientes;`

< [SQL Editor]

Result Grid | Filter Rows: | Export: | Wrap Cell Content:



	email_modificado
▶	ana.garcia@demo.com
	luis.perez@demo.com
	maria.rodriquez@demo.com
	juan.martinez@demo.com
	carmen.hernandez@demo.com
	javier.lopez@demo.com
	laura.gomez@demo.com
	pedro.vazquez@demo.com
	sofia.mendez@demo.com

- Funciones de fecha y hora: DATEADD, DATEDIFF, GETDATE, FORMAT, etc.
 - DATEADD

```
100
404 -- Funciones de fecha y hora
405 -- DATEADD
406 • SELECT DATE_ADD(fecha_nacimiento, INTERVAL 1 YEAR) AS fechaNaci FROM pacientes;
407 • SELECT DATE_ADD(fecha_nacimiento, INTERVAL 1 DAY) AS fechaNaci FROM pacientes;
```

<

Result Grid

Filter Rows: | Export:  | Wrap Cell Content: 

	fechaNaci
▶	1986-04-15
	1991-05-22
	1979-09-30
	1983-11-11
	1996-03-17
	1988-07-25
	1993-08-30

```
407 • SELECT DATE_ADD(fecha, INTERVAL 1 DAY) AS siguiente_dia FROM citas;
```

```
408 • SELECT DATE_ADD(hora, INTERVAL 1 HOUR) AS siguiente_hora FROM citas;
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content: [IA](#)

	siguiente_dia
▶	2024-08-02
	2024-08-02
	2024-08-02
	2024-08-02
	2024-08-02
	2024-08-02
	2024-08-02
	2024-08-02
	2024-08-02
	2024-08-02

```
408 • SELECT DATE_ADD(hora, INTERVAL 1 HOUR) AS siguiente_hora FROM citas;
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content: [IA](#)

	siguiente_hora
▶	10:00:00
	10:30:00
	11:00:00
	11:30:00
	12:00:00
	12:30:00
	13:00:00
	13:30:00
	14:00:00

- DATEDIFF

```
410 -- DATEDIFF
```

```
411 • SELECT DATEDIFF(CURDATE(), fecha_nacimiento) AS edad_dias FROM pacientes;
```

```
412 • SELECT DATEDIFF(CURDATE(), fecha_nacimiento) AS edad_dias FROM pacientes;
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content: [IA](#)

	edad_dias
▶	14346
	12483
	16735
	15232
	10723
	13515
	11652
	16266

```
412 • SELECT DATEDIFF(fecha, CURDATE()) AS dias_para_cita FROM citas;
```

```
413 • SELECT DATEDIFF(fecha, '2023-01-01') AS dias_desde_inicio_ano FROM citas;
```

Result Grid   Filter Rows: Export:  Wrap Cell Content: 

	dias_para_cita
▶	7
	7
	7
	7
	7
	7
	7
	7
	7

```
413 • SELECT DATEDIFF(fecha, '2023-01-01') AS dias_desde_inicio_ano FROM citas;
```

Result Grid   Filter Rows: Export:  Wrap Cell Content: 

	dias_desde_inicio_ano
▶	578
	578
	578
	578
	578
	578
	578
	578
	578

- GETDATE

```
415 -- GETDATE
416 • SELECT CURDATE() AS fecha_actual;
417 • SELECT NOW() AS fecha_hora_actual;
418 • SELECT CURTIME() AS hora_actual;
```

Result Grid | Filter Rows: | Exp

fecha_actual
2024-07-25

Result Grid | Filter Rows: | Exp

fecha_hora_actual
2024-07-25 20:57:51

```
418 • SELECT CURTIME() AS hora_actual;
```

Result Grid | Filter Rows: | Exp

hora_actual
20:58:27

- FORMAT

```
420 -- FORMAT
421 • SELECT DATE_FORMAT(fecha_nacimiento, '%d/%m/%Y') AS fecha_nacimiento_formateada FROM pacientes;
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content: |

fecha_nacimiento_formateada
15/04/1985
22/05/1990
30/09/1978
11/11/1982
17/03/1995
25/07/1987
30/08/1992
12/01/1980
04/10/1989

422 • `SELECT DATE_FORMAT(fecha, '%M %d, %Y') AS fecha_formateada FROM citas;`

Result Grid	
Filter Rows:	Export: Wrap Cell Content:
fecha_formateada	
▶ August 01, 2024	
August 01, 2024	
August 01, 2024	
August 01, 2024	
August 01, 2024	
August 01, 2024	
August 01, 2024	
August 01, 2024	
August 01, 2024	
August 01, 2024	

423 • `SELECT DATE_FORMAT(hora, '%h:%i %p') AS hora_formateada FROM citas;`

Result Grid	
Filter Rows:	Export: Wrap Cell Content:
hora_formateada	
▶ 09:00 AM	
09:30 AM	
10:00 AM	
10:30 AM	
11:00 AM	
11:30 AM	
12:00 PM	
12:30 PM	
01:00 PM	

- Funciones matemáticas: ROUND, CEILING, FLOOR, etc.
 - ROUND

```

425      -- Funciones matematicas
426      -- ROUND
427 • SELECT ROUND(AVG(LENGTH(nombre)), 2) AS promedio_longitud_nombre FROM pacientes;
428 • SELECT ROUND(LENGTH(direccion), -1) AS direccion_redondeada FROM pacientes;
429 • SELECT ROUND(LENGTH(email), 0) AS longitud_email FROM pacientes;

```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

	promedio_longitud_nombre
▶	6.30

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

	direccion_redondeada
▶	20
	10
	20
	20
	20
	20
	20
	20
	20

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

	longitud_email
▶	22
	22
	27
	25
	28
	24
	23
	25

- CEILING

```

431      -- CEILING
432 • SELECT CEILING(LENGTH(nombre)) AS longitud_nombre_redondeada_arriba FROM pacientes;
433 • SELECT CEILING(LENGTH(direccion) / 10) * 10 AS direccion_redondeada_arriba FROM pacientes;

```

Result Grid		Filter Rows:	Export:	Wrap Cell Content:
longitud_nombre_redondeada_arriba				
3				
4				
6				
4				
6				
6				
5				
5				
6				

```

433 • SELECT CEILING(LENGTH(direccion) / 10) * 10 AS direccion_redondeada_arriba FROM pacientes;
434 • SELECT CEILING(LENGTH(email) / 5) * 5 AS email_redondeado_arriba FROM pacientes;

```

Result Grid		Filter Rows:	Export:	Wrap Cell Content:
direccion_redondeada_arriba				
20				
20				
20				
20				
20				
20				
20				
20				
20				

```

434 • SELECT CEILING(LENGTH(email) / 5) * 5 AS email_redondeado_arriba FROM pacientes;
435 • SELECT CEILING(LENGTH(email) / 5) * 5 AS email_redondeado_arriba FROM pacientes;

```

Result Grid		Filter Rows:	Export:	Wrap Cell Content:
email_redondeado_arriba				
25				
25				
30				
25				
30				
25				
25				
25				
25				

- FLOOR

```

436      -- FLOOR
437 • SELECT FLOOR(LENGTH(nombre)) AS longitud_nombre_redondeada_abajo FROM pacientes;
438 • SELECT FLOOR(LENGTH(direccion) / 10) * 10 AS direccion_redondeada_abajo FROM pacientes;

```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

	longitud_nombre_redondeada_abajo
▶ 3	
4	
6	
4	
6	
6	
5	
5	
6	

```

438 • SELECT FLOOR(LENGTH(direccion) / 10) * 10 AS direccion_redondeada_abajo FROM pacientes;
439 • SELECT FLOOR(LENGTH(email) / 5) * 5 AS email_redondeado_abajo FROM pacientes;

```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

	direccion_redondeada_abajo
▶ 10	
10	
10	
10	
10	
10	
10	
10	
10	

```

439 • SELECT FLOOR(LENGTH(email) / 5) * 5 AS email_redondeado_abajo FROM pacientes;
440

```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

	email_redondeado_abajo
▶ 20	
20	
25	
25	
25	
20	
20	
25	
20	

4. Consultas SQL y subconsultas.

- Consultas básicas: SELECT, WHERE, ORDER BY, GROUP BY, HAVING.

SELECT

407 -- Consultas Basicas

408 • `select * from pacientes;`

Result Grid							
		Filter Rows:		Edit:		Export/Import:	
						Wrap Cell Content:	
	id_paciente	nombre	apellido	fecha_nacimiento	direccion	telefono	email
▶	1	Ana	García	1985-04-15	Av. Libertador 234	555-1234	ana.garcia@example.com
	2	Luis	Pérez	1990-05-22	Calle Mayor 12	555-5678	luis.perez@example.com
	3	María	Rodríguez	1978-09-30	Plaza Central 5	555-8765	maria.rodriguez@example.com
	4	Juan	Martínez	1982-11-11	Calle del Sol 89	555-4321	juan.martinez@example.com
	5	Carmen	Hernández	1995-03-17	Av. del Norte 101	555-3456	carmen.hernandez@example.com
	6	Javier	López	1987-07-25	Calle del Río 55	555-7890	javier.lopez@example.com
	7	Laura	Gómez	1992-08-30	Calle de la Luna 78	555-6543	laura.gomez@example.com
	8	Pedro	Vázquez	1980-01-12	Calle del Mar 22	555-3456	pedro.vazquez@example.com
	9	Sofía	Méndez	1989-10-04	Av. de la Paz 13	555-9876	sofia.mendez@example.com
	10	Carlos	Álvarez	1991-06-18	Calle de la Esperanza 27	555-1122	carlos.alvarez@example.com
	11	Lucía	Paniagua	1986-02-20	Calle de la Aleoría 39	555-3344	lucia.paniagua@example.com
pacientes 1 ×							

WHERE

409 -- Where

410 • `select * from pacientes where apellido = 'garcía';`

Result Grid							
		Filter Rows:		Edit:		Export/Import:	
						Wrap Cell Content:	
	id_paciente	nombre	apellido	fecha_nacimiento	direccion	telefono	email
▶	1	Ana	García	1985-04-15	Av. Libertador 234	555-1234	ana.garcia@example.com
	14	Fernando	García	1994-07-11	Calle del Ocaso 45	555-6655	fernando.garcia@example.com
	41	Ricardo	García	1981-06-06	Calle de la Victoria 43	555-8890	ricardo.garcia@example.com
*	NULL	NULL	NULL	NULL	NULL	NULL	NULL

ORDER BY

412 • `select * from pacientes order by fecha_nacimiento desc;`

Result Grid							
		Filter Rows:		Edit:		Export/Import:	
						Wrap Cell Content:	
	id_paciente	nombre	apellido	fecha_nacimiento	direccion	telefono	email
▶	34	Alejandra	Castano	1996-09-30	Calle de la Libertad 64	555-3344	alejandra.castano@example.com
	5	Carmen	Hernández	1995-03-17	Av. del Norte 101	555-3456	carmen.hernandez@example.com
	27	Sandra	Córdoba	1995-02-07	Av. del Sol 47	555-8765	sandra.cordoba@example.com
	14	Fernando	García	1994-07-11	Calle del Ocaso 45	555-6655	fernando.garcia@example.com
	30	Cecilia	Tovar	1994-04-04	Calle del Alba 74	555-7890	cecilia.tovar@example.com
	40	Valeria	Múñoz	1994-02-18	Calle de la Rosa 76	555-7789	valeria.munoz@example.com
	38	Mariana	Figueroa	1993-07-10	Calle de la Esperanza 99	555-3345	mariana.figueroa@example.com
	17	Natalia	Salazar	1993-05-06	Calle de la Primavera 50	555-8899	natalia.salazar@example.com
	47	Juliana	Peña	1993-02-26	Calle de los Pinos 18	555-7789	juliana.pena@example.com
	21	Diana	Torres	1992-10-15	Av. del Océano 88	555-6677	diana.torres@example.com
	7	Laura	Gómez	1992-08-30	Calle de la Luna 78	555-6543	laura.gomez@example.com
pacientes 3 ×							

GROUP BY

```

413      -- group by
414 •    select apellido, count(*) as cantidad from pacientes group by apellido;
415      -- having

```

Result Grid			Filter Rows:	Export:	Wrap Cell Content:
	apellido	cantidad			
▶	García	3			
	Pérez	1			
	Rodríguez	1			
	Martínez	1			
	Hernández	1			
	López	1			
	Gómez	2			
	Vázquez	1			
	Méndez	1			
	Álvarez	1			
	Paniagua	1			
	Cruz	1			

Result 5 ×

HAVING

```

415      -- having
416 •    select apellido, count(*) as cantidad from pacientes group by apellido having count(*) > 1;
417

```

Result Grid			Filter Rows:	Export:	Wrap Cell Content:
	apellido	cantidad			
▶	García	3			
	Gómez	2			
	Ramírez	2			
	Sánchez	2			

- Subconsultas: en SELECT, FROM, WHERE; subconsultas correlacionadas.

SELECT

C

```
419 • select nombre, (select count(*) from citas where citas.paciente_id = pacientes.id_paciente) as citas_cou  
420 from pacientes;
```

421

422

423

424

Result Grid |  Filter Rows: | Export:  | Wrap Cell Content: 

	nombre	citas_count
▶	Ana	1
	Luis	1
	María	1
	Juan	1
	Carmen	1
	Javier	1
	Laura	1
	Pedro	1
	Sofía	1
	Carlos	1
	Lucía	1
	Miguel	1

Result 7 x

SOBCONSULTA FROM

```
422 • select avg(citas_count) as promedio_citas from (  
423     select count(*) as citas_count from citas group by paciente_id  
424 ) as subconsulta;
```

425

426

Result Grid |  Filter Rows: | Export:  | Wrap Cell Content: 

	promedio_citas
▶	1.0000

SUBCONSULTA WHERE

```

425      -- WHERE
426 • select * from doctores where id_doctor in (
427      select doctor_id from citas where fecha = '2024-08-01'
428      );
429

```

id_doctor	nombre	apellido	especialidad_id
1	carlos	martínez	1
2	gabriela	mendoza	2
3	jose	ramírez	3
4	maría	fernández	4
5	alejandro	herrería	5
6	paola	mora	1
7	nicolás	suárez	2
8	isabella	león	3
9	juan	téllez	4
10	laura	rodríguez	5
NULL	NULL	NULL	NULL

Subconsulta correlacionada

```

430 • select nombre from pacientes p where exists (
431      select 1 from citas c where c.paciente_id = p.id_paciente and c.fecha = '2024-08-01'
432      );

```

nombre
Ana
Luis
María
Juan
Carmen
Javier
Laura
Pedro
Sofía
Carlos

5. Creación y asignación de usuarios y privilegios.

- Creación de usuarios: comandos CREATE USER, definición de roles.


```

64 • create user 'joe'@'localhost' identified with mysql_native_password by '123456';
65 • create user 'richard'@'localhost' identified with mysql_native_password by '654321';
66 • create user 'angel'@'localhost' identified with mysql_native_password by '246531';
67 • create user 'marco'@'localhost' identified with mysql_native_password by '135642';
68 • create role 'doctor';
69 • create role 'admin';

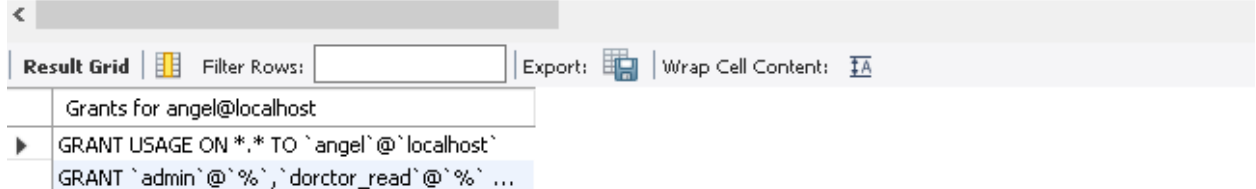
```

- Asignación de privilegios: GRANT, REVOKE, administración de roles.

```

94 • show grants for 'angel'@'localhost';
95 • revoke 'doctor_read_write' from 'angel'@'localhost';
96 • grant 'dorctor_read' to 'angel'@'localhost';
97
98

```



The screenshot shows a MySQL client window with a toolbar at the top containing 'Result Grid', 'Filter Rows', 'Export', and 'Wrap Cell Content'. Below the toolbar, the command 'show grants for 'angel'@'localhost';' has been executed. The output is displayed in a table with two rows: 'GRANT USAGE ON *.* TO `angel`@`localhost`' and 'GRANT `admin`@`%`,`dorctor_read`@`%` ...'.

```

75 -- Otorgar privilegios a roles
76 • grant select on clinica.* to 'dorctor_read';
77 • grant select, insert, update, delete on clinica.* to 'doctor_read_write';
78 • grant all privileges on clinica.* to 'admin';
79
80 -- Asignar roles a usuarios
81 • grant 'dorctor_read' to 'joe'@'localhost';
82 • grant 'doctor_read_write' to 'richard'@'localhost';
83 • grant 'doctor_read_write' to 'angel'@'localhost';
84 • grant 'admin' to 'marco'@'localhost';

```

- Políticas de seguridad y mejores prácticas.

```

97
98 • alter user 'joe'@'localhost' identified by '1234567';
99 • set global default_password_lifetime = 90;
100

```

- Ejercicios prácticos de creación y asignación de usuarios y roles.

```
92 • create user 'linda'@'localhost' identified with mysql_native_password by 'linda_password';
93 • grant 'doctor_read' to 'linda'@'localhost';
94 • show grants for 'linda'@'localhost';
95
96
97
```

6. Implementación de vistas, procedimientos almacenados y triggers.

Vistas.

- Vista para ver información básica de pacientes:

```

533 -- CREAT VIEWS
534 -- Vista para ver información básica de pacientes:
535 • create view vista_pacientes_basica as
536 select
537     id_paciente,
538     nombre,
539     apellido,
540     fecha_nacimiento,
541     email
542 from
543     pacientes;
544 • select * from vista_pacientes_basica;
545

```

Result Grid  Filter Rows: <input type="text"/> Export:  Wrap Cell Content: 					
	id_paciente	nombre	apellido	fecha_nacimiento	email
▶	1	Ana	García	1985-04-15	ana.garcia@example.com
	2	Luis	Pérez	1990-05-22	luis.perez@example.com
	3	María	Rodríguez	1978-09-30	maria.rodriguez@example.com
	4	Juan	Martínez	1982-11-11	juan.martinez@example.com
	5	Carmen	Hernández	1995-03-17	carmen.hernandez@example.com
	6	Javier	López	1987-07-25	javier.lopez@example.com
	7	Laura	Gómez	1992-08-30	laura.gomez@example.com
	8	Pedro	Vázquez	1980-01-12	pedro.vazquez@example.com
	9	Sofía	Méndez	1989-10-04	sofia.mendez@example.com
	10	Carlos	Álvarez	1991-06-18	carlos.alvarez@example.com
	11	Lucía	Paniagua	1986-02-20	lucia.paniagua@example.com
	12	Miguel	Cruz	1975-12-05	miguel.cruz@example.com

vista_pacientes_basica 3 x

- -- Vista para ver citas programadas por consultorio:

```

546      -- Vista para ver citas programadas por consultorio:
547 •   create view vista_citas_por_consultorio as
548       select
549         co.nombre as consultorio,
550         c.id_cita,
551         p.nombre as nombre_paciente,
552         d.nombre as nombre_doctor,
553         c.fecha,
554         c.hora
555       from
556         citas c
557       join
558         pacientes p on c.paciente_id = p.id_paciente
559       join
560         doctores d on c.doctor_id = d.id_doctor
561       join
562         consultorios co on c.consultorio_id = co.id_consultorio;
563 •   select * from vista_citas_por_consultorio;
564

```

	consultorio	id_cita	nombre_paciente	nombre_doctor	fecha	hora
▶	consultorio 1	1	Ana	carlos	2024-08-01	09:00:00
	consultorio 2	2	Luis	gabriela	2024-08-01	09:30:00
	consultorio 3	3	María	jose	2024-08-01	10:00:00
	consultorio 4	4	Juan	maría	2024-08-01	10:30:00
	consultorio 5	5	Carmen	alejandro	2024-08-01	11:00:00
	consultorio 6	6	Javier	paola	2024-08-01	11:30:00
	consultorio 7	7	Laura	nicolás	2024-08-01	12:00:00
	consultorio 8	8	Pedro	isabella	2024-08-01	12:30:00
	consultorio 9	9	Sofía	juan	2024-08-01	13:00:00
	consultorio 10	10	Carlos	laura	2024-08-01	13:30:00
	consultorio 11	11	Lucía	francisco	2024-08-02	09:00:00
	consultorio 12	12	Miguel	camila	2024-08-02	09:30:00

vista_citas_por_consultorio 4 ×

- Vista para ver especialidades y sus doctores:

```

565      -- Vista para ver especialidades y sus doctores:
566 •   create view vista_especialidades_doctores as
567   select
568       e.nombre as especialidad,
569       d.nombre as nombre_doctor,
570       d.apellido as apellido_doctor
571   from
572       especialidades e
573   join
574       doctores d on e.id_especialidad = d.especialidad_id;
575 •   select * from vista_especialidades_doctores;
576

```

Result Grid Filter Rows: <input type="text"/> Export: Wrap Cell Content:			
	especialidad	nombre_doctor	apellido_doctor
▶	medicina general	carlos	martínez
	cardiología	gabriela	mendoza
	neurología	jose	ramírez
	ortopedia	maría	fernández
	ginecología	alejandro	herrería
	medicina general	paola	mora
	cardiología	nicolás	suárez
	neurología	isabella	león
	ortopedia	juan	téllez
	ginecología	laura	rodríguez
	medicina general	francisco	quintero
	cardiología	camila	mendoza
	neurología	sergio	cardenas
	ortopedia	valeria	gómez
	ginecología	david	pinto
	medicina general	juliana	rojas
	cardiología	eduardo	suárez
	neurología	mónica	salazar
	ortopedia	felipe	molina
	ginecología	catalina	rodríguez
	medicina general	marco	arango
	cardiología	andrea	velasquez
	neurología	manuel	peña

vista especialidades doctores 6 ×

Procedimientos Almacenados

Ej:

Se visualiza los datos previos a los cambios

```
592 • select * from pacientes;
593      -- aplicacion
```

Result Grid							
Filter Rows:							
Edit:							
Export/Import:							
Wrap Cell Content:							
	id_paciente	nombre	apellido	fecha_nacimiento	direccion	telefono	email
	4	Juan	Martínez	1982-11-11	Calle del Sol 89	555-4321	juan.martinez@example.com
	5	Carmen	Hernández	1995-03-17	Av. del Norte 101	555-3456	carmen.hernandez@example.com
	6	Javier	López	1987-07-25	Calle del Río 55	555-7890	javier.lopez@example.com
	7	Laura	Gómez	1992-08-30	Calle de la Luna 78	555-6543	laura.gomez@example.com
	8	Pedro	Vázquez	1980-01-12	Calle del Mar 22	555-3456	pedro.vazquez@example.com
	9	Sofía	Méndez	1989-10-04	Av. de la Paz 13	555-9876	sofia.mendez@example.com
	10	Carlos	Álvarez	1991-06-18	Calle de la Esperanza 27	555-1122	carlos.alvarez@example.com
	11	Lucía	Paniagua	1986-02-20	Calle de la Alegría 39	555-3344	lucia.paniagua@example.com
	12	Miguel	Cruz	1975-12-05	Av. de los Pinos 66	555-2233	miguel.cruz@example.com
	13	Valeria	Díaz	1983-09-14	Calle del Viento 87	555-7788	valeria.diaz@example.com
	14	Fernando	García	1994-07-11	Calle del Ocaso 45	555-6655	fernando.garcia@example.com
	15	Isabel	Ramírez	1979-11-28	Av. de la Libertad 32	555-5566	isabel.ramirez@example.com
	16	Andrés	Ríos	1988-04-09	Calle del Río 93	555-4433	andres.rios@example.com
	17	Natalia	Salazar	1993-05-06	Calle de la Primavera 50	555-8899	natalia.salazar@example.com
	18	Ricardo	Mora	1981-08-23	Av. de la Cultura 19	555-2234	ricardo.mora@example.com
	19	Patricia	Uribe	1990-12-01	Calle de los Olivos 74	555-3345	patricia.uribe@example.com
	20	Alejandro	Sánchez	1984-03-14	Calle del Alba 33	555-5566	alejandro.sanchez@example.com
	21	Diana	Torres	1992-10-15	Av. del Océano 88	555-6677	diana.torres@example.com
	22	Héctor	Suárez	1980-07-02	Calle de la Pradera 29	555-7789	hector.suarez@example.com
	23	Paola	Cárdenas	1991-01-18	Calle del Jardín 57	555-8890	paola.cardenas@example.com
	24	Jorge	Molina	1986-06-30	Av. de los Rosales 62	555-9900	jorge.molina@example.com
	25	Mónica	Vega	1979-03-09	Calle de la Conquista 84	555-1234	monica.vega@example.com
	26	Esteban	Márquez	1988-05-14	Calle de la Fe 91	555-5678	esteban.marquez@example.com
	27	Sandra	Córdoba	1995-02-07	Av. del Sol 47	555-8765	sandra.cordoba@example.com

pacientes 7 ×

-- 1 Procedimiento para actualizar la dirección de un paciente:

```

577 -- PROCEDIMIENTOS ALMACENADOS
578 -- 1Procedimiento para actualizar la dirección de un paciente:
579 delimiter //
580
581 • create procedure actualizar_direccion_paciente(
582     in p_id_paciente int,
583     in p_direccion varchar(100)
584 )
585 • begin
586     update pacientes
587     set direccion = p_direccion
588     where id_paciente = p_id_paciente;
589 end //
590
591 delimiter ;
592 • select * from pacientes;

```

Result Grid			
Filter Rows:			
Export:			
Wrap Cell Content:			
	especialidad	nombre_doctor	apellido_doctor
▶	medicina general	carlos	martínez
	cardiología	gabriela	mendoza
	neurología	jose	ramírez
	ortopedia	maría	fernández
	ginecología	alejandro	herrería
	medicina general	paola	mora
	cardiología	nicolás	suárez
	neurología	isabella	león
	ortopedia	juan	téllez
	ginecología	laura	rodríguez
	medicina general	francisco	quintero
	cardiología	camila	mendoza
	neurología	sergio	cardenas
	ortopedia	valeria	gómez
	ginecología	david	pinto
	medicina general	juliana	rojas
	cardiología	eduardo	suárez
	neurología	mónica	salazar
	ortopedia	felipe	molina

```

594 • call actualizar_direccion_paciente(1, 'Av Aucas e Inagpirca La Florida');
595 • select * from pacientes where id_paciente = 1;
596

```



Result Grid							
Filter Rows:							
Edit:							
Export/Import:							
Wrap Cell Content:							
	id_paciente	nombre	apellido	fecha_nacimiento	direccion	telefono	email
▶	1	Ana	García	1985-04-15	Av Aucas e Inagpirca La Florida	555-1234	ana.garcia@example.com
	2	Luis	Pérez	1990-05-22	Calle Mayor 12	555-5678	luis.perez@example.com
	3	María	Rodríguez	1978-09-30	Plaza Central 5	555-8765	maria.rodriguez@example.com
	4	Juan	Martínez	1982-11-11	Calle del Sol 89	555-4321	juan.martinez@example.com
	5	Carmen	Hernández	1995-03-17	Av. del Norte 101	555-3456	carmen.hernandez@example.com
	6	Javier	López	1987-07-25	Calle del Río 55	555-7890	javier.lopez@example.com

Obtener info del paciente

```
1 • create procedure obtener_info_paciente(  
2     in p_id_paciente int  
3 )  
4 begin  
5     select  
6         nombre,  
7         apellido,  
8         fecha_nacimiento,  
9         direccion,  
10        telefono,  
11        email  
12    from  
13        pacientes  
14    where  
15        id_paciente = p_id_paciente;  
16 end //
```

En esta ocasion ingresamos la ID 1 pero podemos ingresar la ID según deseamos consular la información

```
632     delimiter ;  
633 • call obtener_info_paciente(1);  
634
```

Result Grid						
Filter Rows: <input type="text"/>						
Export:  Wrap Cell Content: 						
	nombre	apellido	fecha_nacimiento	direccion	telefono	email
▶	Ana	García	1985-04-15	Av Aucas e Inagpirca La Florida	555-1234	ana.garcia@example.com

Trigger

Primero creamos una tabla para guardar la auditoria que es realizada por los Trigger

```
635 -- TRIGGER
636 -- CREAMOS UNA TABLA PARA REGISTRAR LAS AUDITORIA DE LOS TRIGGERS
637 create table auditoria_triggers (
638     id_auditoria int auto_increment primary key,
639     nombre_trigger varchar(100),
640     tabla_afectada varchar(100),
641     tipo_operacion varchar(10),
642     fecha_ejecucion datetime,
643     detalles varchar(255)
644 );
```

- Trigger para registrar auditoría en la tabla de pacientes al insertar un nuevo paciente:

```
-- Trigger para registrar auditoría en la tabla de pacientes al insertar un nuevo paciente:
delimiter //
create trigger tr_auditoria_insert_paciente
after insert on pacientes
for each row
begin
    insert into auditoria_triggers (nombre_trigger, tabla_afectada, tipo_operacion, fecha_ejecucion, detalles)
    values ('tr_auditoria_insert_paciente', 'pacientes', 'INSERT', now(), concat('Nuevo paciente insertado: ', new.nombre, ' ', new.apellido));
end //
delimiter ;
```









Ejemplo de inserción y verificación:

```
657 • insert into pacientes (nombre, apellido, fecha_nacimiento, direccion, telefono, email)
658 | values ('Juan', 'Perez', '1985-05-20', 'Calle Falsa 123', '555-1234', 'juan.perez@example.com');
659
660 • select * from auditoria_triggers;
```

Result Grid						
Filter Rows: <input type="text"/>						
Edit: Export/Import: Wrap Cell Content:						
	id_auditoria	nombre_trigger	tabla_afectada	tipo_operacion	fecha_ejecucion	detalles
1	1	tr_auditoria_insert_paciente	pacientes	INSERT	2024-07-29 21:59:58	Nuevo paciente insertado: Juan Perez
*	NULL	NULL	NULL	NULL	NULL	NULL

- Actualizacion de la fecha de una cita


```
673 update citas set fecha = '2024-08-02' where id_cita = 1;
674 -- verificamos el cambio en la tabla
675 • select * from auditoria_triggers;
676
```

Result Grid		 Filter Rows: <input type="text"/>	Edit: 		 Export/Import: 	 Wrap Cell Content: 
	id_auditoria	nombre_trigger	tabla_afectada	tipo_operacion	fecha_ejecucion	detalles
▶	1	tr_auditoria_insert_paciente	pacientes	INSERT	2024-07-29 21:59:58	Nuevo paciente insertado: Juan Perez
	2	tr_auditoria_update_cita	citas	UPDATE	2024-07-29 22:02:27	Cita actualizada: ID Cita = 1
✱	NULL	NULL	NULL	NULL	NULL	NULL

- ```
677 -- Trigger para registrar auditoría en la tabla de doctores al eliminar un doctor:
678 delimiter //
679 • create trigger tr_auditoria_delete_doctor
680 after delete on doctores
681 for each row
682 begin
683 insert into auditoria_triggers (nombre_trigger, tabla_afectada, tipo_operacion, fecha_ejecucion, detalles)
684 values ('tr_auditoria_delete_doctor', 'doctores', 'DELETE', now(), concat('Doctor eliminado: ID Doctor = ', old.id_doctor));
685 end //
686 delimiter ;
687
```

```
687 • -- Verificacion de los datos y eliminacion
688 delete from doctores where id_doctor = 1;
689 • select * from auditoria_triggers;
690
```

Result Grid

Filter Rows:

| Edit:

| Export/Import:

| Wrap Cell Content:

|   | id_auditoria | nombre_trigger               | tabla_afectada | tipo_operacion | fecha_ejecucion     | detalles                             |
|---|--------------|------------------------------|----------------|----------------|---------------------|--------------------------------------|
| ▶ | 1            | tr_auditoria_insert_paciente | pacientes      | INSERT         | 2024-07-29 21:59:58 | Nuevo paciente insertado: Juan Perez |
|   | 2            | tr_auditoria_update_cita     | citas          | UPDATE         | 2024-07-29 22:02:27 | Cita actualizada: ID Cita = 1        |
|   | 3            | tr_auditoria_delete_doctor   | doctores       | DELETE         | 2024-07-29 22:12:30 | Doctor eliminado: ID Doctor = 1      |
| ✱ | NULL         | NULL                         | NULL           | NULL           | NULL                | NULL                                 |

## 7. Manejo de errores y pruebas de fallos.



### Manejo de errores y transacciones

Transacciones: utiliza transacciones para agrupar varias operaciones SQL en una única unidad de trabajo. Si alguna operación falla, puedes revertir toda la transacción para mantener la consistencia de la base de datos.

Manejo de errores: en MySQL, puedes utilizar DECLARE ... HANDLER para manejar errores específicos y tomar correctivas.

#### Ejemplo 1: inserción de datos con transacciones

```
22 -- Iniciar una transacción
23 • START TRANSACTION;
24
25 -- Intentar insertar datos en la tabla de departamentos usando INSERT IGNORE
26 • INSERT IGNORE INTO Departamentos (id, nombre) VALUES
27 (1, 'Recursos Humanos'),
28 (2, 'TI'),
29 (3, 'Ventas');
30
31
32
33
34
35
36
37
38
39 -- Confirmar la transacción
40 • COMMIT;
41
42 -- Mensaje de éxito
43 • SELECT 'Datos insertados correctamente.' AS mensaje_exito;
44
```

|                                                                                                                                                                                                      |                                   |
|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|-----------------------------------|
| <                                                                                                                                                                                                    |                                   |
| Result Grid                                                                                                                                                                                          | Filter Rows: <input type="text"/> |
| Export:  Wrap Cell Content:  |                                   |
| mensaje_exito                                                                                                                                                                                        |                                   |
| ▶                                                                                                                                                                                                    | Datos insertados correctamente.   |

## Ejemplo 2: verificación e inserción condicional

Para el manejo de errores de forma más explícita, puedes usar INSERT IGNORE y verificar los resultados después de cada inserción.



```
-- Intentar insertar datos en la tabla de departamentos usando INSERT IGNORE
INSERT IGNORE INTO Departamentos (id, nombre) VALUES (1, 'Recursos Humanos');
SET error = error + IF(ROW_COUNT() = 0, 1, 0);

INSERT IGNORE INTO Departamentos (id, nombre) VALUES (2, 'TI');
SET error = error + IF(ROW_COUNT() = 0, 1, 0);



INSERT IGNORE INTO Departamentos (id, nombre) VALUES (3, 'Ventas');
SET error = error + IF(ROW_COUNT() = 0, 1, 0);

104 -- Manejo de errores y confirmación/rollback de la transacción
105 IF error > 0 THEN
106 ROLLBACK;
107 SELECT 'Error al insertar datos. Transacción revertida.' AS mensaje_error;
108 ELSE
109 COMMIT;
110 SELECT 'Datos insertados correctamente.' AS mensaje_exit;
111 END IF;
```

```
116 • -- Llamar al procedimiento almacenado para insertar los datos
117 CALL InsertarDatos();
118
```

|                                                                                                                                                                                                      |                                   |
|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|-----------------------------------|
| <                                                                                                                                                                                                    |                                   |
| Result Grid                                                                                                                                                                                          | Filter Rows: <input type="text"/> |
| Export:  Wrap Cell Content:  |                                   |
| mensaje_exit                                                                                                                                                                                         |                                   |
| ▶                                                                                                                                                                                                    | Datos insertados correctamente.   |

```
116 • -- Llamar al procedimiento almacenado para insertar los datos
117 CALL InsertarDatos();
118
```

|                                                                                                                                                                                                      |                                                 |
|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|-------------------------------------------------|
| <                                                                                                                                                                                                    |                                                 |
| Result Grid                                                                                                                                                                                          | Filter Rows: <input type="text"/>               |
| Export:  Wrap Cell Content:  |                                                 |
| mensaje_error                                                                                                                                                                                        |                                                 |
| ▶                                                                                                                                                                                                    | Error al insertar datos. Transacción revertida. |

### Ejemplo 3: Limpieza de datos existentes antes de inserciones

Otra forma de evitar errores de duplicación es limpiar los datos existentes antes de realizar nuevas inserciones. Esto se puede hacer usando los comandos TRUNCATE o DELETE.

```
122 • SET FOREIGN_KEY_CHECKS = 0;
123
124 -- Limpiar datos existentes
125 • TRUNCATE TABLE Empleados;
126 • TRUNCATE TABLE Departamentos;
127
128 -- Activar verificaciones de claves foráneas
129 • SET FOREIGN_KEY_CHECKS = 1;

134 -- Intentar insertar datos en la tabla de departamentos
135 • INSERT INTO Departamentos (id, nombre) VALUES
136 (1, 'Recursos Humanos'),
137 (2, 'TI'),
138 (3, 'Ventas');
139
140 -- Intentar insertar datos en la tabla de empleados
141 • INSERT INTO Empleados (id, nombre, departamento_id, salario) VALUES
142 (1, 'Juan Pérez', 1, 50000.00),
143 (2, 'María López', 2, 60000.00),
144 (3, 'Carlos Ruiz', 3, 55000.00),
145 (4, 'Ana Gómez', 2, 62000.00),
146 (5, 'Luis Martínez', NULL, 48000.00);
147

148 -- Confirmar la transacción
149 • COMMIT;
150
151 -- Mensaje de éxito
152 • SELECT 'Datos insertados correctamente.' AS mensaje_exito;
```

#### Notas Importantes:

- Variables de error: En estos ejemplos, se utiliza una variable @error para rastrear si ha ocurrido un error en alguna de las operaciones. Si hay un error, se realiza un rollback de la transacción; de lo contrario, se confirma (commit).

- @@ERROR: En MySQL, no hay una variable @@ERROR como en otros sistemas de bases de datos. Para capturar errores, en realidad necesitas verificar cada declaración en el contexto de una aplicación o un script de cliente que ejecute estas sentencias SQL.
- FOREIGN KEY: Asegúrate de que las claves foráneas no causen problemas de integridad referencial al insertar o eliminar datos.

## **8. Generación de backups y procesos de importación/exportación.**

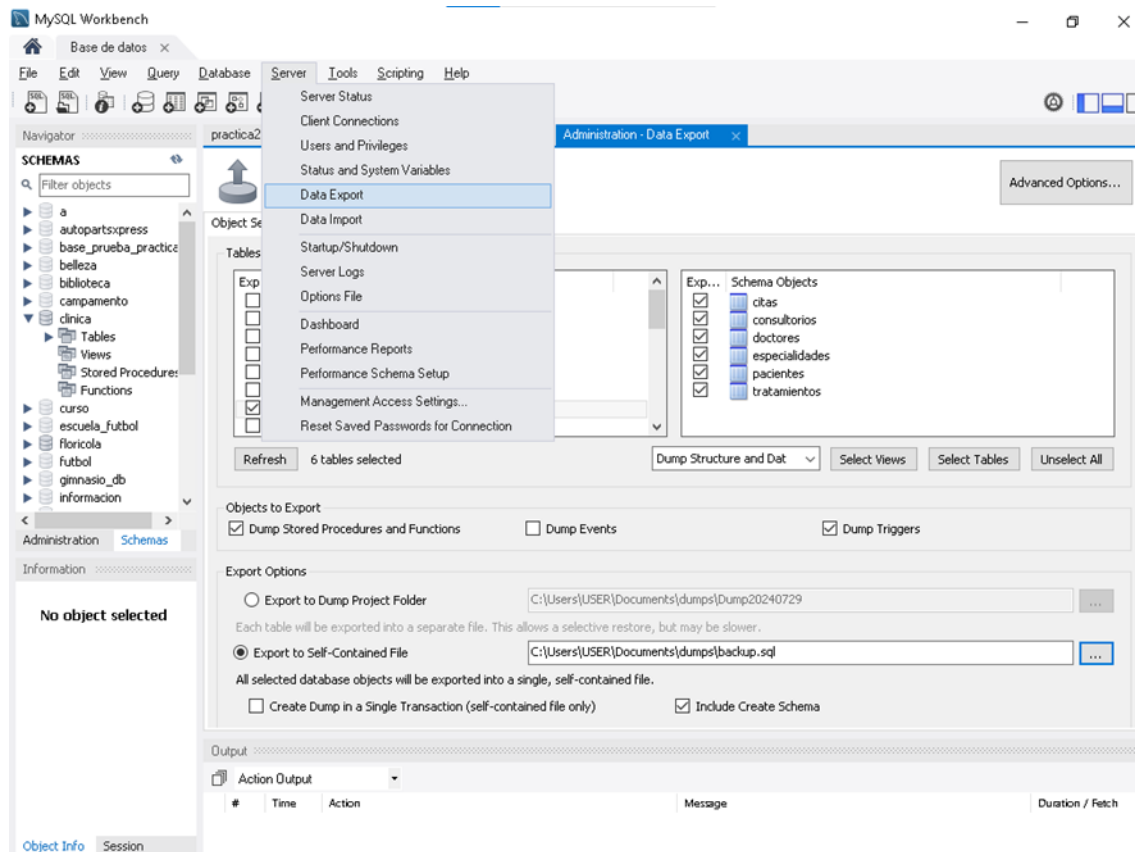
### **Los Backups**

Son copias de seguridad de la base de datos que se pueden usar para restaurar datos en caso de pérdida o corrupción.

### **Importación/Exportación**

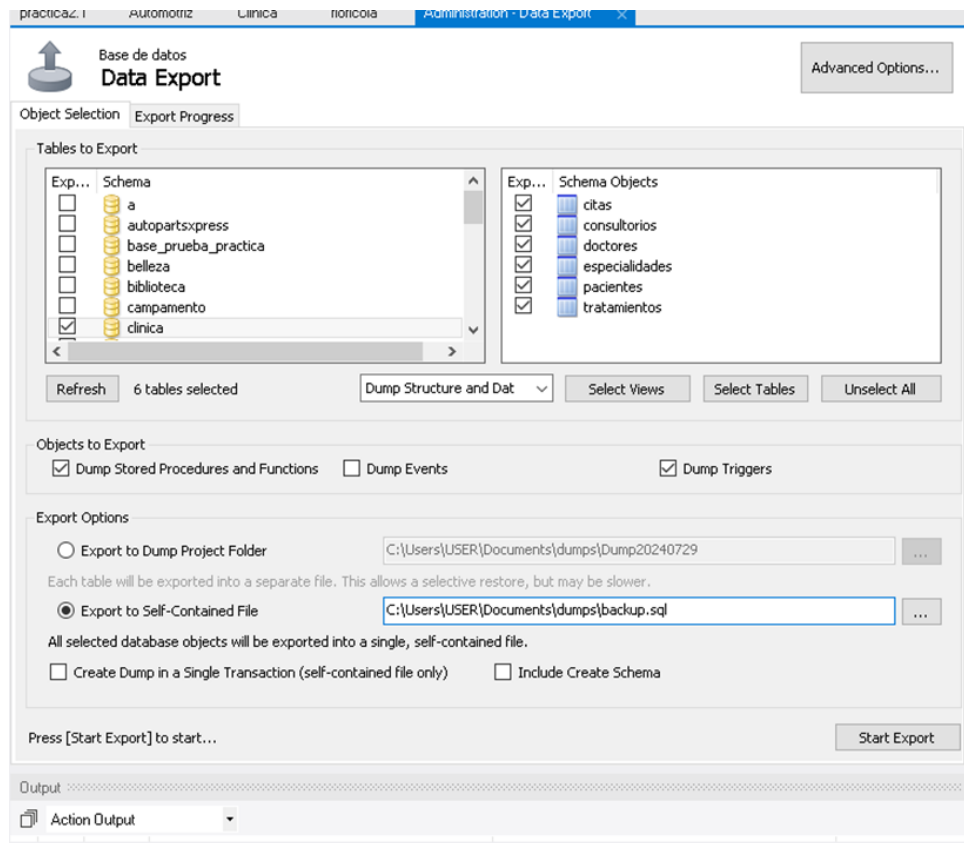
La exportación es el proceso de extraer datos de una base de datos y guardarlos en un archivo, mientras que la importación es el proceso de cargar datos desde un archivo a la base de datos.

- 1) Se ingresa en server para poder hacer el export de la base de datos y para poder exportar de manera manual las cosas las tablas los triggers,etc.

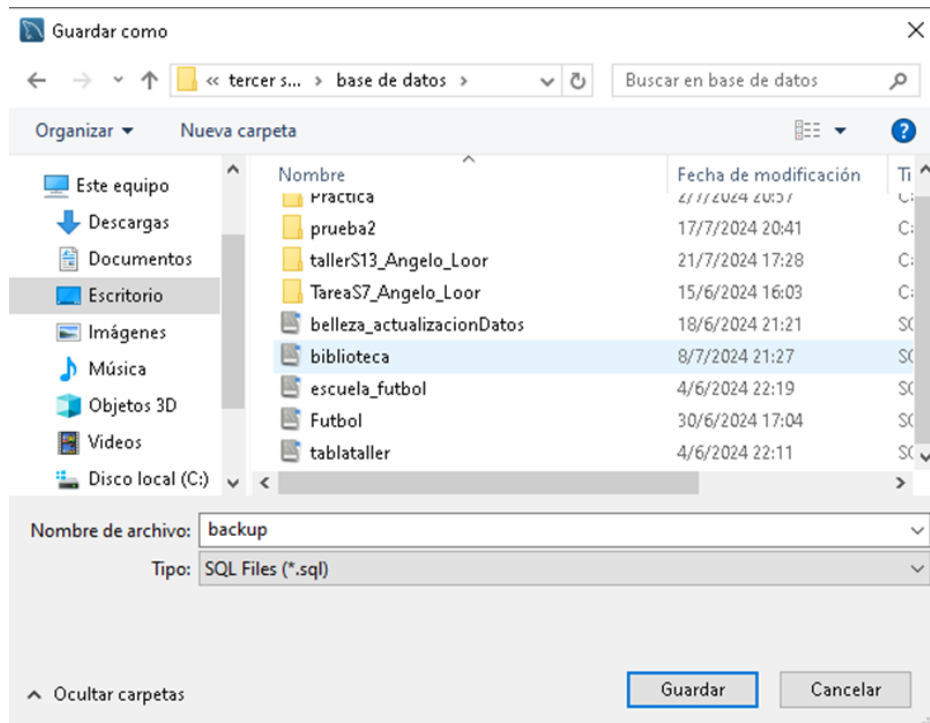


- 2) Seleccionamos todas las configuraciones que queremos guardar en la exportación y en la parte de export to selft podemos averiguar la ruta en la que queremos guardar.

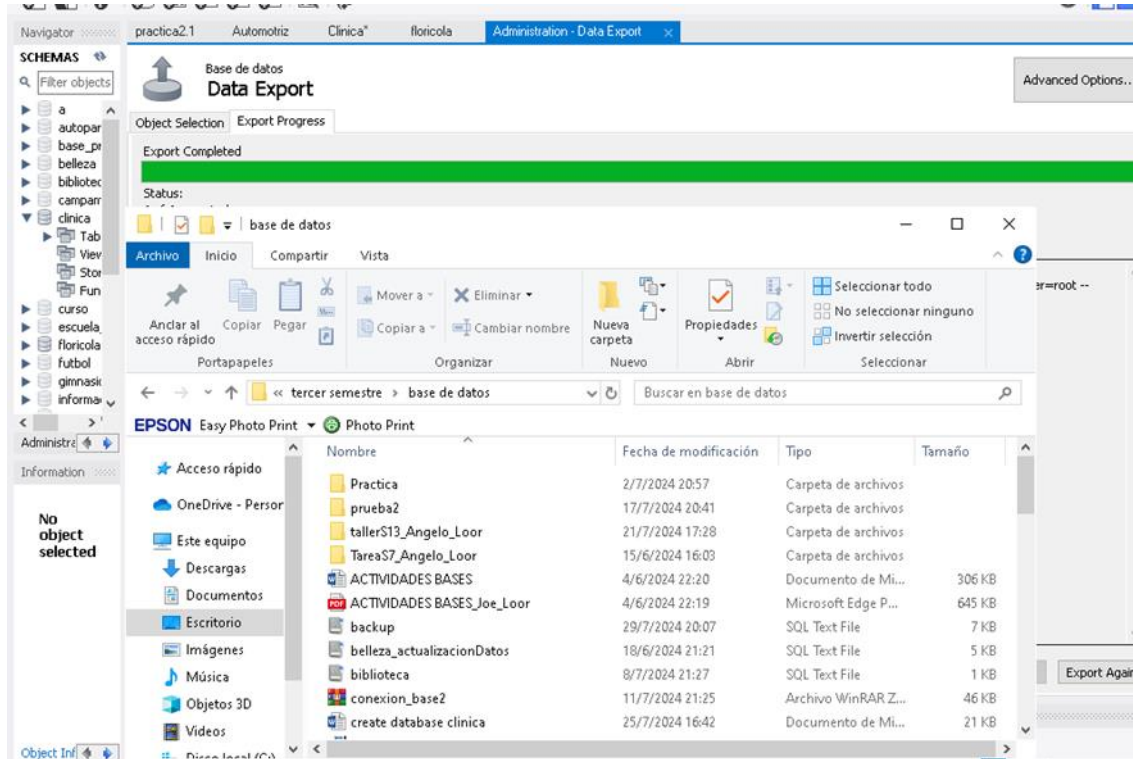




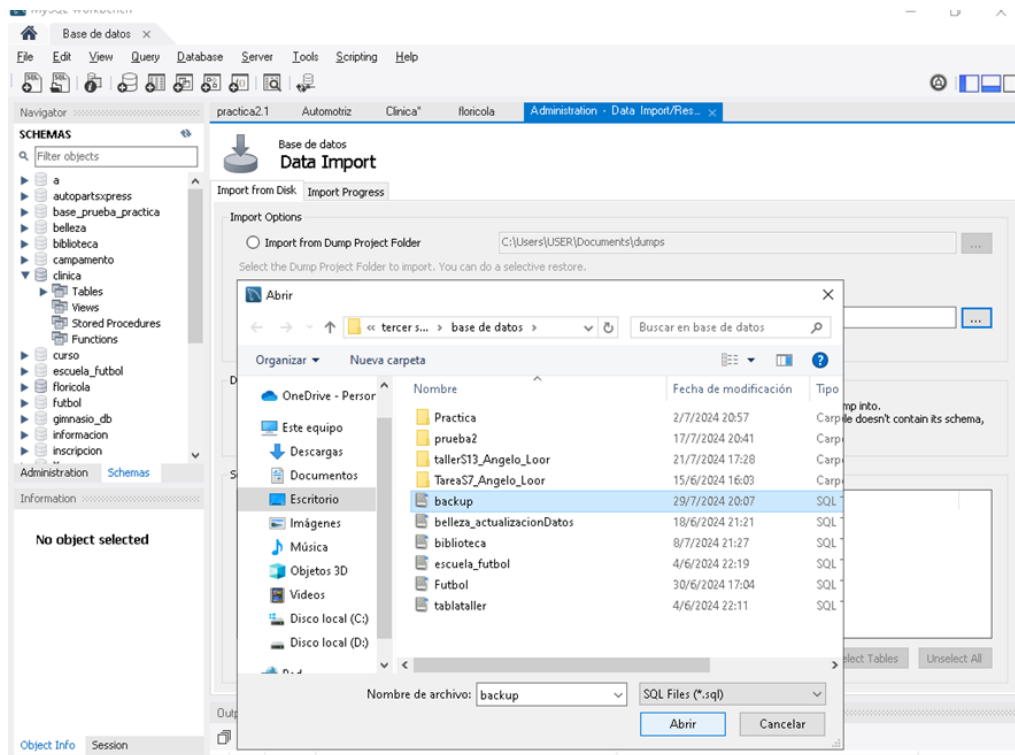
3) Seleccionamos la ruta en la que queremos exportar y su respectivo nombre que le demos.



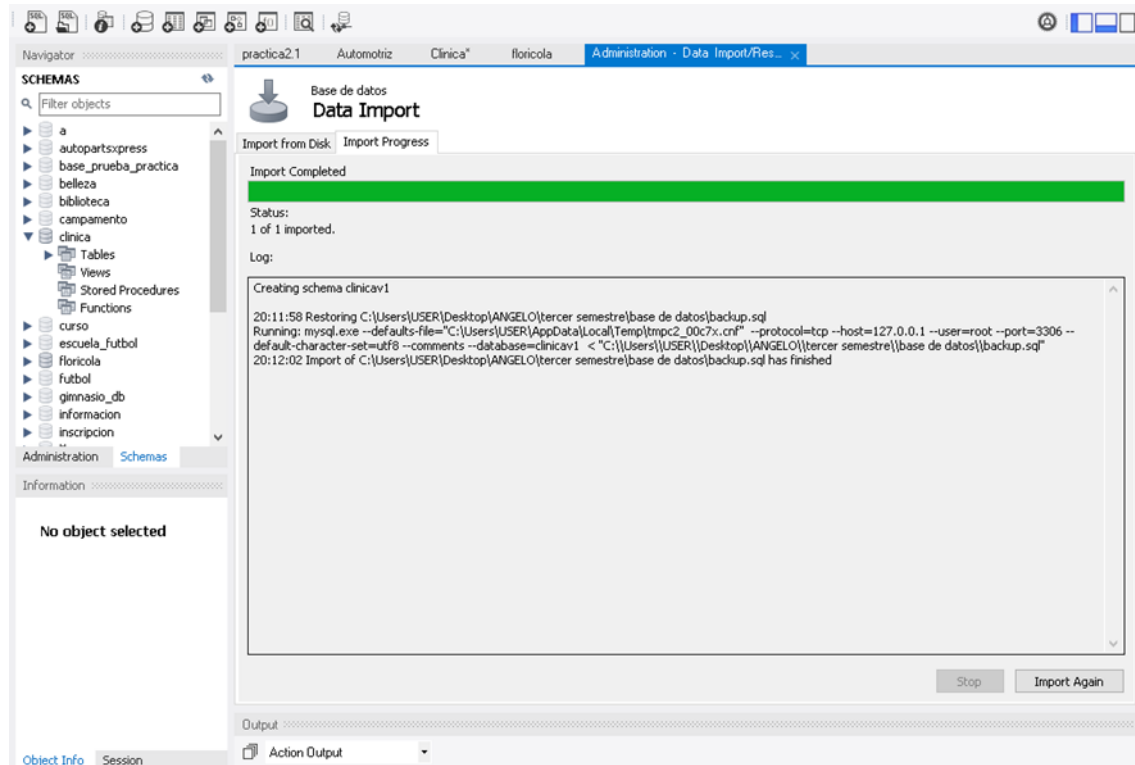
4) Verificamos si se exporto nuestra copia.



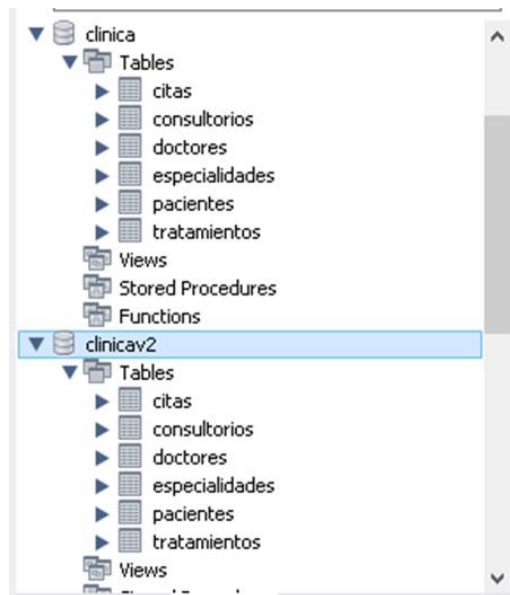
5) Importamos el backup que creamos.



6) Importamos con los cambios y seleccionamos new para poder nombrar la base de datos.



7) Verificamos el backup de la base de datos.



## 9. Aplicación de COMMIT y ROLLBACK.

En estos ejemplos, COMMIT asegura que los cambios realizados se guarden en la base de datos.

### COMMIT

```
-- COMMIT
-- Confirmar inserción de nuevo paciente:
start transaction;
insert into pacientes (nombre, apellido, fecha_nacimiento, direccion, telefono, email)
values ('luis', 'salazar', '1985-06-10', 'calle nueva 12', '555-0001', 'luis.salazar@example.com');
commit;

-- Confirmar actualización de datos del doctor:
start transaction;
update doctores
set nombre = 'mario', apellido = 'fernández'
where id_doctor = 10;
commit;

-- Confirmar eliminación de una cita:
start transaction;
delete from citas
where id_cita = 5;
commit;
```

ROLLBACK: deshace cualquier cambio realizado durante la transacción si algo sale mal o si se decide no proceder.

```

-- ROLLBACK
-- Revertir inserción de nuevo paciente:
• start transaction;
• insert into pacientes (nombre, apellido, fecha_nacimiento, direccion, telefono, email)
 values ('ana', 'garcía', '1990-07-15', 'avenida central 15', '555-5555', 'ana.garcia@example.com');
• rollback;

-- Revertir actualización de datos del doctor:
• start transaction;
• update doctores
 set nombre = 'mariana', apellido = 'sánchez'
 where id_doctor = 8;
• rollback;

-- Revertir eliminación de una cita:
• start transaction;
• delete from citas
 where id_cita = 12;
• rollback;

```

## 10. Joins

### INNER JOIN

Consulta para obtener la información completa de las citas, incluyendo los datos del paciente, el doctor y el consultorio para cada cita:

```

•select
 citas.id_cita,
 pacientes.nombre as paciente_nombre,
 pacientes.apellido as paciente_apellido,
 doctores.nombre as doctor_nombre,
 doctores.apellido as doctor_apellido,
 consultorios.nombre as consultorio_nombre,
 citas.fecha,
 citas.hora
from
 citas
inner join pacientes on citas.paciente_id = pacientes.id_paciente
inner join doctores on citas.doctor_id = doctores.id_doctor
inner join consultorios on citas.consultorio_id = consultorios.id_consultorio;

```

| id_cita | paciente_nombre | paciente_apellido | doctor_nombre | doctor_apellido | consultorio_nombre |
|---------|-----------------|-------------------|---------------|-----------------|--------------------|
| 1       | Ana             | García            | carlos        | martínez        | consultorio 1      |
| 2       | Luis            | Pérez             | gabriela      | mendoza         | consultorio 2      |
| 3       | María           | Rodríguez         | jose          | ramírez         | consultorio 3      |
| 4       | Juan            | Martínez          | maría         | fernández       | consultorio 4      |
| 5       | Carmen          | Hernández         | alejandro     | herrería        | consultorio 5      |
| 6       | Javier          | López             | paola         | mora            | consultorio 6      |
| 7       | Laura           | Gómez             | nicolás       | suárez          | consultorio 7      |
| 8       | Pedro           | Vázquez           | isabella      | león            | consultorio 8      |
| 9       | Sofía           | Méndez            | juan          | téllez          | consultorio 9      |
| 10      | Carlos          | Álvarez           | laura         | rodríguez       | consultorio 10     |
| 11      | Lucía           | Paniagua          | francisco     | quintero        | consultorio 11     |

Consulta para obtener la información de cada cita junto con la descripción del tratamiento asociado a esa cita (si existe):

```
-- Consulta para obtener la información de cada cita junto con la descripción del
-- tratamiento asociado a esa cita (si existe):
```

```
•select
 citas.id_cita,
 pacientes.nombre as paciente_nombre,
 pacientes.apellido as paciente_apellido,
 doctores.nombre as doctor_nombre,
 doctores.apellido as doctor_apellido,
 consultorios.nombre as consultorio_nombre,
 citas.fecha,
 citas.hora,
 tratamientos.descripcion as tratamiento_descripcion
from
 citas
inner join pacientes on citas.paciente_id = pacientes.id_paciente
inner join doctores on citas.doctor_id = doctores.id_doctor
inner join consultorios on citas.consultorio_id = consultorios.id_consultorio
left join tratamientos on citas.id_cita = tratamientos.cita_id;
```

| id_cita | paciente_nombre | paciente_apellido | doctor_nombre | doctor_apellido | consultorio_nombre |
|---------|-----------------|-------------------|---------------|-----------------|--------------------|
| 1       | Ana             | García            | carlos        | martínez        | consultorio 1      |
| 2       | Luis            | Pérez             | gabriela      | mendoza         | consultorio 2      |
| 3       | María           | Rodríguez         | jose          | ramírez         | consultorio 3      |
| 4       | Juan            | Martínez          | maría         | fernández       | consultorio 4      |
| 5       | Carmen          | Hernández         | alejandro     | herrería        | consultorio 5      |
| 6       | Javier          | López             | paola         | mora            | consultorio 6      |
| 7       | Laura           | Gómez             | nicolás       | suárez          | consultorio 7      |
| 8       | Pedro           | Vázquez           | isabella      | león            | consultorio 8      |
| 9       | Sofía           | Méndez            | juan          | téllez          | consultorio 9      |
| 10      | Carlos          | Álvarez           | laura         | rodríguez       | consultorio 10     |
| 11      | Lucía           | Paniagua          | francisco     | quintero        | consultorio 11     |

## LEFT JOIN

Consulta para obtener todos los pacientes y las citas que han tenido, incluyendo aquellos pacientes que no tienen citas:

```
sql> -- Consulta para obtener todos los pacientes y las citas que han tenido,
-- incluyendo aquellos pacientes que no tienen citas:
```

```
•select
```

```
 pacientes.nombre as paciente_nombre,
 pacientes.apellido as paciente_apellido,
 citas.id_cita,
 citas.fecha,
 citas.hora
```

```
from
```

```
 pacientes
```

```
left join citas on pacientes.id_paciente = citas.paciente_id;
```

| paciente_nombre | paciente_apellido | id_cita | fecha      | hora     |  |
|-----------------|-------------------|---------|------------|----------|--|
| Ana             | García            | 1       | 31/07/2024 | 9:00:00  |  |
| Luis            | Pérez             | 2       | 31/07/2024 | 9:30:00  |  |
| María           | Rodríguez         | 3       | 31/07/2024 | 10:00:00 |  |
| Juan            | Martínez          | 4       | 31/07/2024 | 10:30:00 |  |
| Carmen          | Hernández         | 5       | 31/07/2024 | 11:00:00 |  |
| Javier          | López             | 6       | 31/07/2024 | 11:30:00 |  |
| Laura           | Gómez             | 7       | 31/07/2024 | 12:00:00 |  |
| Pedro           | Vázquez           | 8       | 31/07/2024 | 12:30:00 |  |
| Sofía           | Méndez            | 9       | 31/07/2024 | 13:00:00 |  |
| Carlos          | Álvarez           | 10      | 31/07/2024 | 13:30:00 |  |
| Lucía           | Paniagua          | 11      | 01/08/2024 | 9:00:00  |  |
| ...             | ...               | ...     | ...        | ...      |  |

Consulta para obtener una lista de todos los pacientes junto con la información de las citas que han tenido, si es que han tenido alguna:



```
> -- Consulta para obtener una lista de todos los pacientes junto con la
-- información de las citas que han tenido, si es que han tenido alguna:
•select
 pacientes.id_paciente,
 pacientes.nombre as paciente_nombre,
 pacientes.apellido as paciente_apellido,
 citas.id_cita,
 citas.fecha,
 citas.hora
from
 pacientes
left join citas on pacientes.id_paciente = citas.paciente_id;
```

| id_paciente | paciente_nombre | paciente_apellido | id_cita | fecha      | hora     |
|-------------|-----------------|-------------------|---------|------------|----------|
| 1           | Ana             | García            | 1       | 31/07/2024 | 9:00:00  |
| 2           | Luis            | Pérez             | 2       | 31/07/2024 | 9:30:00  |
| 3           | María           | Rodríguez         | 3       | 31/07/2024 | 10:00:00 |
| 4           | Juan            | Martínez          | 4       | 31/07/2024 | 10:30:00 |
| 5           | Carmen          | Hernández         | 5       | 31/07/2024 | 11:00:00 |
| 6           | Javier          | López             | 6       | 31/07/2024 | 11:30:00 |
| 7           | Laura           | Gómez             | 7       | 31/07/2024 | 12:00:00 |
| 8           | Pedro           | Vázquez           | 8       | 31/07/2024 | 12:30:00 |
| 9           | Sofía           | Méndez            | 9       | 31/07/2024 | 13:00:00 |
| 10          | Carlos          | Álvarez           | 10      | 31/07/2024 | 13:30:00 |
| 11          | Lucía           | Paniagua          | 11      | 01/08/2024 | 9:00:00  |
| --          | --              | --                | --      | --         | --       |

## RIGHT JOIN

Consulta para obtener todos los doctores y las citas que tienen, incluyendo aquellos doctores que no han tenido citas:

```
ql> -- Consulta para obtener todos los doctores y las citas que tienen,
-- incluyendo aquellos doctores que no han tenido citas:
```

```
•select
 doctores.nombre as doctor_nombre,
 doctores.apellido as doctor_apellido,
 citas.id_cita,
 citas.fecha,
 citas.hora
from
 doctores
right join citas on doctores.id_doctor = citas.doctor_id;
```

| doctor_nombre | doctor_apellido | id_cita | fecha      | hora     |  |
|---------------|-----------------|---------|------------|----------|--|
| carlos        | martínez        | 1       | 31/07/2024 | 9:00:00  |  |
| gabriela      | mendoza         | 2       | 31/07/2024 | 9:30:00  |  |
| jose          | ramírez         | 3       | 31/07/2024 | 10:00:00 |  |
| maría         | fernández       | 4       | 31/07/2024 | 10:30:00 |  |
| alejandro     | herrería        | 5       | 31/07/2024 | 11:00:00 |  |
| paola         | mora            | 6       | 31/07/2024 | 11:30:00 |  |
| nicolás       | suárez          | 7       | 31/07/2024 | 12:00:00 |  |
| isabella      | león            | 8       | 31/07/2024 | 12:30:00 |  |
| juan          | téllez          | 9       | 31/07/2024 | 13:00:00 |  |
| laura         | rodríguez       | 10      | 31/07/2024 | 13:30:00 |  |
| francisco     | quintero        | 11      | 01/08/2024 | 9:00:00  |  |
| ..            | .               | ..      | ..         | ..       |  |

Consulta para obtener una lista de todas las citas junto con la información del doctor que realizó cada cita, incluyendo aquellos doctores que no tienen citas:

```

mysql> -- Consulta para obtener una lista de todas las citas junto con la información del
-- doctor que realizó cada cita, incluyendo aquellos doctores que no tienen citas:
•select
 citas.id_cita,
 citas.fecha,
 citas.hora,
 doctores.nombre as doctor_nombre,
 doctores.apellido as doctor_apellido
from
 citas
right join doctores on citas.doctor_id = doctores.id_doctor;

```

| id_cita | fecha      | hora     | doctor_nombre | doctor_apellido |
|---------|------------|----------|---------------|-----------------|
| 1       | 31/07/2024 | 9:00:00  | carlos        | martínez        |
| 2       | 31/07/2024 | 9:30:00  | gabriela      | mendoza         |
| 3       | 31/07/2024 | 10:00:00 | jose          | ramírez         |
| 4       | 31/07/2024 | 10:30:00 | maría         | fernández       |
| 5       | 31/07/2024 | 11:00:00 | alejandro     | herrería        |
| 6       | 31/07/2024 | 11:30:00 | paola         | mora            |
| 7       | 31/07/2024 | 12:00:00 | nicolás       | suárez          |
| 8       | 31/07/2024 | 12:30:00 | isabella      | león            |
| 9       | 31/07/2024 | 13:00:00 | juan          | téllez          |
| 10      | 31/07/2024 | 13:30:00 | laura         | rodríguez       |
| 11      | 01/08/2024 | 9:00:00  | francisco     | quintero        |
| --      | --         | --       | --            | --              |

## FULL OUTER JOIN

En MySQL, no existe `FULL OUTER JOIN` directamente, por lo que se utiliza una combinación de `LEFT JOIN` y `RIGHT JOIN`

Consulta para obtener la información completa de citas y pacientes, incluyendo aquellos registros en ambas tablas que no tienen coincidencias

```

•(select
 pacientes.nombre as paciente_nombre,
 pacientes.apellido as paciente_apellido,
 citas.id_cita,
 citas.fecha,
 citas.hora
from
 pacientes
left join citas on pacientes.id_paciente = citas.paciente_id)
union
(select
 pacientes.nombre as paciente_nombre,
 pacientes.apellido as paciente_apellido,
 citas.id_cita,
 citas.fecha,
 citas.hora
from
 citas
right join pacientes on citas.paciente_id = pacientes.id_paciente);

```

| paciente_nombre | paciente_apellido | id_cita | fecha      | hora     |
|-----------------|-------------------|---------|------------|----------|
| Ana             | García            | 1       | 31/07/2024 | 9:00:00  |
| Luis            | Pérez             | 2       | 31/07/2024 | 9:30:00  |
| María           | Rodríguez         | 3       | 31/07/2024 | 10:00:00 |
| Juan            | Martínez          | 4       | 31/07/2024 | 10:30:00 |
| Carmen          | Hernández         | 5       | 31/07/2024 | 11:00:00 |
| Javier          | López             | 6       | 31/07/2024 | 11:30:00 |
| Laura           | Gómez             | 7       | 31/07/2024 | 12:00:00 |
| Pedro           | Vázquez           | 8       | 31/07/2024 | 12:30:00 |
| Sofía           | Méndez            | 9       | 31/07/2024 | 13:00:00 |
| Carlos          | Álvarez           | 10      | 31/07/2024 | 13:30:00 |
| Lucía           | Paniagua          | 11      | 01/08/2024 | 9:00:00  |
| ...             | ...               | ...     | ...        | ...      |

## CROSS JOIN

Consulta para obtener una combinación de todas las especialidades con todos los doctores (Nota: `CROSS JOIN` puede producir una gran cantidad de resultados):

```

sql> -- Consulta para obtener una combinación de todas las especialidades con todos los
-- doctores (Nota: 'CROSS JOIN' puede producir una gran cantidad de resultados):
•select
 especialidades.nombre as especialidad_nombre,
 doctores.nombre as doctor_nombre,
 doctores.apellido as doctor_apellido
from
 especialidades
cross join doctores;

```

| especialidad_nombre | doctor_nombre | doctor_apellido |
|---------------------|---------------|-----------------|
| medicina general    | jennifer      | carrillo        |
| medicina general    | sergio        | hernández       |
| medicina general    | carolina      | beltrán         |
| medicina general    | felipe        | medina          |
| medicina general    | mariana       | contreras       |
| medicina general    | santiago      | valencia        |
| medicina general    | verónica      | pardo           |
| medicina general    | manuel        | moreno          |
| medicina general    | isabel        | patifio         |
| medicina general    | alejandro     | quintero        |
| medicina general    | paula         | moreno          |

Consulta para obtener una combinación de todos los pacientes con todos los doctores.

```

pl> -- Consulta para obtener una combinación de todos los pacientes con todos los doctores.
•select
 pacientes.nombre as paciente_nombre,
 pacientes.apellido as paciente_apellido,
 doctores.nombre as doctor_nombre,
 doctores.apellido as doctor_apellido
from
 pacientes
cross join doctores;

```

| paciente_nombre | paciente_apellido | doctor_nombre | doctor_apellido |
|-----------------|-------------------|---------------|-----------------|
| Hugo            | Núñez             | carlos        | martínez        |
| Marcela         | Ramírez           | carlos        | martínez        |
| Mario           | Jiménez           | carlos        | martínez        |
| Juliana         | Peña              | carlos        | martínez        |
| Catalina        | Ruiz              | carlos        | martínez        |
| Daniel          | Rendón            | carlos        | martínez        |
| Lina            | Campos            | carlos        | martínez        |
| Francisco       | Sánchez           | carlos        | martínez        |
| Beatriz         | Lozano            | carlos        | martínez        |
| Ricardo         | García            | carlos        | martínez        |
| Valeria         | Múñoz             | carlos        | martínez        |

## Joins Complejos: Múltiples Tablas y Subconsultas

### Consulta Compleja con Múltiples Tablas

Consulta para obtener el listado de tratamientos, junto con la información de la cita, paciente y doctor asociados:

```

• select
 tratamientos.id_tratamiento,
 tratamientos.descripcion as tratamiento_descripcion,
 citas.id_cita,
 pacientes.nombre as paciente_nombre,
 pacientes.apellido as paciente_apellido,
 doctores.nombre as doctor_nombre,
 doctores.apellido as doctor_apellido
from
 tratamientos
inner join citas on tratamientos.cita_id = citas.id_cita
inner join pacientes on citas.paciente_id = pacientes.id_paciente
inner join doctores on citas.doctor_id = doctores.id_doctor;

```

| id_tratamiento | tratamiento_descripcion | id_cita | paciente_nombre | paciente_apellido | doctor_nombre |
|----------------|-------------------------|---------|-----------------|-------------------|---------------|
| 1              | examen físico completo  | 1       | Ana             | García            | Carlos        |
| 2              | evaluación cardiológica | 2       | Luis            | Pérez             | García        |
| 3              | consultoría neurológica | 3       | María           | Rodríguez         | Joel          |
| 4              | revisión ortopédica     | 4       | Juan            | Martínez          | María         |
| 5              | examen ginecológico     | 5       | Carmen          | Hernández         | Alfonso       |
| 6              | consulta pediátrica     | 6       | Javier          | López             | Pablo         |
| 7              | chequeo dermatológico   | 7       | Laura           | Gómez             | Nicolás       |
| 8              | evaluación urológica    | 8       | Pedro           | Vázquez           | Isabel        |
| 9              | limpieza dental         | 9       | Sofía           | Méndez            | Juan          |
| 10             | consulta psiquiátrica   | 10      | Carlos          | Álvarez           | Laureano      |
| 11             | cirugía de apendicitis  | 11      | Lucía           | Paniagua          | Francisco     |

## Subconsulta con Joins

Consulta para obtener una lista de pacientes que han tenido citas recientes (en los últimos 30 días) junto con los tratamientos que recibieron en esas citas

```
> -- Consulta para obtener una lista de pacientes que han tenido citas recientes (en los últimos 30 días)
-- junto con los tratamientos que recibieron en esas citas

•select
 p.id_paciente,
 p.nombre as paciente_nombre,
 p.apellido as paciente_apellido,
 t.descripcion as tratamiento_descripcion,
 c.fecha,
 c.hora
from
 pacientes p
inner join citas c on p.id_paciente = c.paciente_id
inner join tratamientos t on c.id_cita = t.cita_id
where
 c.fecha >= (select date_sub(current_date(), interval 30 day));
```

| id_paciente | paciente_nombre | paciente_apellido | tratamiento_descripcion | fecha      | hora |
|-------------|-----------------|-------------------|-------------------------|------------|------|
| 1           | Ana             | García            | examen físico completo  | 31/07/2024 | 9:   |
| 2           | Luis            | Pérez             | evaluación cardiológica | 31/07/2024 | 9:   |
| 3           | María           | Rodríguez         | consultoría neurológica | 31/07/2024 | 10   |
| 4           | Juan            | Martínez          | revisión ortopédica     | 31/07/2024 | 10   |
| 5           | Carmen          | Hernández         | examen ginecológico     | 31/07/2024 | 11   |
| 6           | Javier          | López             | consulta pediátrica     | 31/07/2024 | 11   |
| 7           | Laura           | Gómez             | chequeo dermatológico   | 31/07/2024 | 12   |
| 8           | Pedro           | Vázquez           | evaluación urológica    | 31/07/2024 | 12   |
| 9           | Sofía           | Méndez            | limpieza dental         | 31/07/2024 | 13   |
| 10          | Carlos          | Álvarez           | consulta psiquiátrica   | 31/07/2024 | 13   |
| 11          | Lucía           | Paniagua          | cirugía de apendicitis  | 01/08/2024 | 9:   |

## 11)Concurrencia

### 1. Uso de Transacciones

Implementa transacciones para asegurar que una serie de operaciones de bases de datos se realicen de manera atómica, es decir, todas o ninguna de las operaciones son ejecutadas



```
-- Establecer el nivel de aislamiento a Serializable
•set transaction isolation level serializable;

OK, 0 records retrieved in 28.982ms

sql> -- Iniciar una transacción
•start transaction;

OK, 0 records retrieved in 0s

sql> -- Consultar datos
•select * from pacientes;
```

| id_paciente | nombre | apellido  | fecha_nac  |
|-------------|--------|-----------|------------|
| 1           | Ana    | García    | 14/04/1985 |
| 2           | Luis   | Pérez     | 21/05/1990 |
| 3           | María  | Rodríguez | 29/09/1978 |
| 4           | Juan   | Martínez  | 10/11/1982 |
| 5           | Carmen | Hernández | 16/03/1995 |
| 6           | Javier | López     | 24/07/1987 |
| 7           | Laura  | Gómez     | 29/08/1992 |
| 8           | Pedro  | Vázquez   | 11/01/1986 |
| 9           | Sofía  | Méndez    | 03/10/1989 |
| 10          | Carlos | Álvarez   | 17/06/1991 |
| 11          | Lucía  | Paniagua  | 19/02/1988 |

```
OK, 50 records retrieved in 18.305ms View: [icon] Pages: [icon]

sql> -- Confirmar (commit) la transacción
•commit;
```

## Aislamiento de Transacciones

MySQL soporta diferentes niveles de aislamiento de transacciones que controlan cómo y cuándo los cambios realizados por una transacción son visibles para otras transacciones concurrentes:

- **READ UNCOMMITTED:** Las transacciones pueden ver cambios no confirmados de otras transacciones.
- **READ COMMITTED:** Las transacciones solo ven los cambios que han sido confirmados.
- **REPEATABLE READ:** Las lecturas realizadas durante la transacción verán siempre los mismos datos, aunque otras transacciones hayan hecho cambios.
- **SERIALIZABLE:** Las transacciones se ejecutan como si fueran secuenciales, lo que evita conflictos de concurrencia pero puede reducir el rendimiento.

Ejemplo de uso:

```

-- Establecer el nivel de aislamiento a Serializable
•set transaction isolation level serializable;

OK, 0 records retrieved in 28.982ms

```

### 3. Bloqueos (Locks)

MySQL proporciona mecanismos de bloqueo que permiten controlar el acceso a filas específicas durante las transacciones:

- **LOCK IN SHARE MODE:** Permite que otras transacciones lean las filas bloqueadas, pero no modificarlas.
- **FOR UPDATE:** Bloquea las filas seleccionadas, impidiendo que otras transacciones las lean o modifiquen hasta que se complete la transacción.

```

> -- Consultar datos
•select * from pacientes;

```

| id_paciente | nombre | apellido  | fecha_nacimiento | direccion             |
|-------------|--------|-----------|------------------|-----------------------|
| 1           | Ana    | García    | 14/04/1985       | Av. Libertador 20     |
| 2           | Luis   | Pérez     | 21/05/1990       | Calle Mayor 12        |
| 3           | María  | Rodríguez | 29/09/1978       | Plaza Central 5       |
| 4           | Juan   | Martínez  | 10/11/1982       | Calle del Sol 89      |
| 5           | Carmen | Hernández | 16/03/1995       | Av. del Norte 100     |
| 6           | Javier | López     | 24/07/1987       | Calle del Río 55      |
| 7           | Laura  | Gómez     | 29/08/1992       | Calle de la Luna      |
| 8           | Pedro  | Vázquez   | 11/01/1980       | Calle del Mar 22      |
| 9           | Sofía  | Méndez    | 03/10/1989       | Av. de la Paz 13      |
| 10          | Carlos | Álvarez   | 17/06/1991       | Calle de la Esperanza |
| 11          | Lucía  | Paniagua  | 19/02/1986       | Calle de la Alegria   |

### Control de Concurrency Optimista

Una técnica que puedes implementar es el control de concurrencia optimista, donde se verifica que los datos no han cambiado desde la última vez que fueron leídos antes de hacer un UPDATE. Esto se puede lograr mediante el uso de un campo de versión:

```
OK, 0 records retrieved in 0.000ms
> • ALTER TABLE citas ADD COLUMN version INT DEFAULT 1;
```

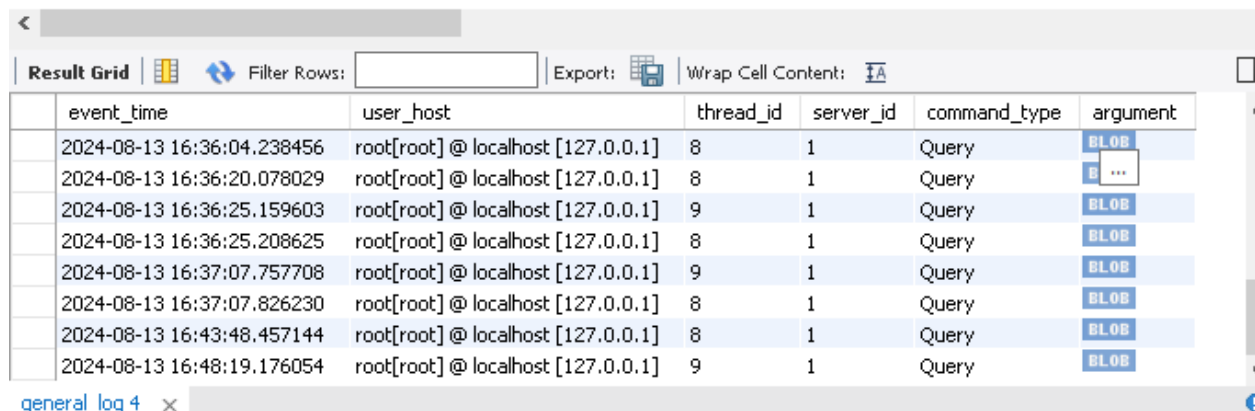
## 12) Auditoría y Seguridad

En esta parte proporcionamos la creación de usuarios ya más especificados para que se realice una cosa a la vez generamos los log en la cual sirven como depuradores y control en la cual se realiza un código en la base de datos

```
417
418 -- seguridad: crear un usuario de solo lectura
419 • create user 'lectura'@'localhost' identified by 'password123';
420 • grant select on clínica.* to 'lectura'@'localhost';
421
422 -- seguridad: crear un usuario con permisos completos
423 • create user 'admin'@'localhost' identified by 'admin123';
424 • grant all privileges on clínica.* to 'admin'@'localhost';
425 • flush privileges;
426
427 -- auditoría: habilitar el log de consultas generales (para rastrear todas las consultas reali
428 • set global general_log = 'on';
429 • set global log_output = 'table';
430
431 -- auditoría: revisar el log de consultas generales
432 • select * from mysql.general_log where argument like '%clínica%';
```

Como se puede apreciar en la parte de cómo se visualizan los log.

```
431 -- auditoría: revisar el log de consultas generales
432 • select * from mysql.general_log where argument like '%clínica%';
433
```



| event_time                 | user_host                        | thread_id | server_id | command_type | argument |
|----------------------------|----------------------------------|-----------|-----------|--------------|----------|
| 2024-08-13 16:36:04.238456 | root[root]@localhost [127.0.0.1] | 8         | 1         | Query        | BLOB     |
| 2024-08-13 16:36:20.078029 | root[root]@localhost [127.0.0.1] | 8         | 1         | Query        | BLOB     |
| 2024-08-13 16:36:25.159603 | root[root]@localhost [127.0.0.1] | 9         | 1         | Query        | BLOB     |
| 2024-08-13 16:36:25.208625 | root[root]@localhost [127.0.0.1] | 8         | 1         | Query        | BLOB     |
| 2024-08-13 16:37:07.757708 | root[root]@localhost [127.0.0.1] | 9         | 1         | Query        | BLOB     |
| 2024-08-13 16:37:07.826230 | root[root]@localhost [127.0.0.1] | 8         | 1         | Query        | BLOB     |
| 2024-08-13 16:43:48.457144 | root[root]@localhost [127.0.0.1] | 8         | 1         | Query        | BLOB     |
| 2024-08-13 16:48:19.176054 | root[root]@localhost [127.0.0.1] | 9         | 1         | Query        | BLOB     |

## 15. Monitoreo del consumo de recursos y el registro de actividades

- **Monitoreo del Buffer Pool de InnoDB:**

**Propósito:** Muestra estadísticas sobre el buffer pool de InnoDB, que es el área de memoria utilizada por InnoDB para almacenar datos y cachés de índices.

792 • `SHOW STATUS LIKE 'Innodb_buffer_pool%';`

| Variable_name                      | Value                                            |
|------------------------------------|--------------------------------------------------|
| Innodb_buffer_pool_dump_status     |                                                  |
| Innodb_buffer_pool_load_status     | Buffer pool(s) load completed at 240813 20:41:52 |
| Innodb_buffer_pool_resize_status   |                                                  |
| Innodb_buffer_pool_load_incomplete | OFF                                              |
| Innodb_buffer_pool_pages_data      | 256                                              |
| Innodb buffer pool bytes data      | 4194304                                          |

- **Monitoreo de Hilos y Conexiones:**

**Propósito:** Proporciona información sobre el número de hilos activos y las conexiones actuales a la base de datos.

794 • `SHOW STATUS LIKE 'Threads%';`  
 795 • `SHOW STATUS LIKE 'Connections';`

| Variable_name     | Value |
|-------------------|-------|
| Threads_cached    | 0     |
| Threads_connected | 2     |
| Threads_created   | 3     |
| Threads_running   | 1     |

- **Información sobre el Esquema de Rendimiento:**

**Propósito:** Verifica si el esquema de rendimiento está habilitado (performance\_schema). El performance\_schema recopila datos sobre el rendimiento del servidor MySQL, como tiempos de espera y eventos.

797 • `SHOW VARIABLES LIKE 'performance_schema%';`  
 798 • `SHOW TABLES FROM performance_schema;`

| Tables_in_performance_schema                 |
|----------------------------------------------|
| accounts                                     |
| cond_instances                               |
| events_stages_current                        |
| events_stages_history                        |
| events_stages_history_long                   |
| events_stages_summary_by_account_by_event... |

- **Estado Global del Servidor:**

**Propósito:** Muestra el estado global del servidor MySQL, incluyendo estadísticas sobre el tráfico de la base de datos, el uso de memoria, y el rendimiento general.

```
800 • SHOW GLOBAL STATUS;
```

```
801
```

| Variable_name            | Value |
|--------------------------|-------|
| Aborted_clients          | 0     |
| Aborted_connects         | 0     |
| Aborted_connects_preauth | 0     |
| Access_denied_errors     | 0     |
| Ad_column_grants         | 0     |
| Ad database grants       | 4     |

- **Listado de Procesos Activos:**

**Propósito:** Muestra una lista de procesos activos en el servidor MySQL. Esto incluye información sobre qué consultas se están ejecutando, cuánto tiempo llevan ejecutándose, y otros detalles relevantes.

```
802 • SELECT * FROM INFORMATION_SCHEMA.PROCESSLIST;
```

```
803
```

```
804 • SHOW ENGINE INNODB STATUS;
```

```
805
```

| ID | USER        | HOST            | DB      | COMMAND | TIME | STATE                    | INFO                                    | TIME_MS    |
|----|-------------|-----------------|---------|---------|------|--------------------------|-----------------------------------------|------------|
| 19 | root        | localhost:60110 | clinica | Query   | 0    | Filling schema table     | SELECT * FROM INFORMATION_SCHEMA.PRO... | 5.039      |
| 18 | root        | localhost:60109 | clinica | Sleep   | 251  |                          |                                         | 251071.564 |
| 5  | system user |                 | NULL    | Daemon  | 0    | InnoDB shutdown handler  | NULL                                    | 0.000      |
| 3  | system user |                 | NULL    | Daemon  | 0    | InnoDB purge worker      | NULL                                    | 0.000      |
| 2  | svstem user |                 | NULL    | Daemon  | 0    | InnoDB purge coordinator | NULL                                    | 0.000      |

- **Estado del Motor InnoDB:**

**Propósito:** Proporciona un informe detallado sobre el estado del motor de almacenamiento InnoDB.

```
804 • SHOW ENGINE INNODB STATUS;
```

```
805
```

| Type   | Name    | Status |
|--------|---------|--------|
| InnoDB | =====.. |        |

## CONSIDERACIONES

- Informe detallado paso a paso del desarrollo del proyecto.
- Registro de 50 usuarios.

- Grabación de un video demostrativo.
- Generación de un backup final.

### Temáticas Disponibles

#### 4. Sistema de Citas Médicas (Integrantes: 4) Joe, Richard, Angel ,Marco

- **Descripción:** Una base de datos para gestionar citas, pacientes, doctores y tratamientos en una clínica.
- **Tablas:** Pacientes, Doctores, Citas, Tratamientos, Consultorios, Especialidades.

### Entregables

- **Informe Detallado:** Documento paso a paso del desarrollo del proyecto.
- **Registro de 50 Usuarios:** Inclusión de 50 usuarios en la base de datos.
- **Video Demostrativo:** Grabación de un video mostrando las principales funcionalidades del proyecto, indicando cada tema, deben participar todo los integrantes del grupo. Hasta 30 minutos el video.
- **Backup Final:** Generar un backup de la base de datos completa.

---

Cada grupo debe asegurarse de seguir estas instrucciones y completar todos los componentes y entregables especificados.