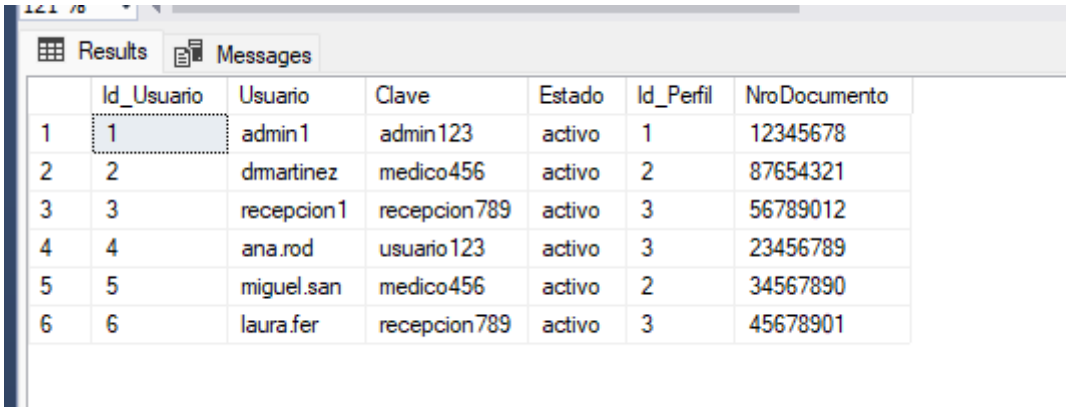
Aplicación de lo investigado – Manejo de Permisos a Nivel Usuario

1. En nuestro sistema, contamos con 3 roles: 1.admin, 2.medico 3.recepcionista. Esto lo haremos mediante la **asignación de roles**. Ahora bien, este es nuestro escenario actual:



Results | Messages

	Id_Usuario	Usuario	Clave	Estado	Id_Perfil	NroDocumento
1	1	admin1	admin123	activo	1	12345678
2	2	dmartinez	medico456	activo	2	87654321
3	3	recepcion1	recepcion789	activo	3	56789012
4	4	ana.rod	usuario123	activo	3	23456789
5	5	miguel.san	medico456	activo	2	34567890
6	6	laura.fer	recepcion789	activo	3	45678901

Supongamos que, en nuestra clínica, necesitamos más de un administrador, con lo cual, quien posea acceso a la base de datos, podrá realizar la siguiente consulta:

```
---Asignar roles a usuarios
---cambiamos de "id_perfil = 2 (médico) a id_perfil = 1 (admin)"
UPDATE Usuario
SET id_perfil = 1
WHERE id_usuario = 2;
```

Verificamos nuestro resultado:

	Id_Usuario	Usuario	Clave	Estado	Id_Perfil	NroDocumento
1	1	admin1	admin123	activo	1	12345678
2	2	dmartinez	medico456	activo	1	87654321

Vemos que user de id = 2 ahora tiene un rol de administrador.

2. Pasamos al siguiente escenario, supongamos que nuestro gestor de la base de datos necesita saber quienes tienen permisos sobre la modificación de una cita programada. Esto se logra mediante la **verificación de permisos**.

```
---Verificacion de permisos de acceso
select id_usuario, p.Nombre, Id_Perfil from usuario u
join Persona p on p.NroDocumento = u.NroDocumento
where Id_Perfil in (2 , 3);
```

Estos serán los usuarios que puedan modificar una cita:

Results Messages

	id_usuario	Nombre	Id_Perfil
1	3	Carlos	3
2	4	Ana	3
3	5	Miguel	2
4	6	Laura	3

O bien, si queremos saber que usuarios tienen permiso para acceder a información de pacientes:

Obtenemos el paciente que queremos modificar

```
select * from paciente where NroDocumento = 23456789
```

Results Messages

	NroPaciente	ObraSocial	NroDocumento
1	4	PAMI	23456789

Solo se podrá ejecutar el siguiente query si se tiene permisos concedidos:

```
--accedemos al usuario logueado al moemnto
DECLARE @UserID INT;
SET @UserID = (SELECT Id_Usuario FROM Usuario WHERE Usuario = SYSTEM_USER);


update Persona
set Telefono = '123456559'
where NroDocumento = (
    select p.NroDocumento from Paciente pa
    join Persona p on pa.NroDocumento = p.NroDocumento
    where pa.NroPaciente = 4
  )
and exists (
    select 1 from Usuario u
    join Perfil pf ON u.Id_Perfil = pf.Id_Perfil
    WHERE u.Id_Usuario = @UserID
      AND pf.Id_Perfil IN (2, 3)  -- Solo medico o recepcionista pueden actualizar
  );
```

3. Supongamos que queremos quitar permisos que hacen posible el acceso a información sensible a alguno de los usuarios, esto lo logramos **revocando el permiso** de uno de ellos

```
--Revocacion de permisos
update Usuario
set Id_Perfil = null where Id_Usuario = 2
```

Esto no sería posible ya que nuestro sistema no admite campos nulos en Id_Perfil, con lo cual, deberemos crear un nuevo id_perfil, como ser “Visitante” (un perfil sin permisos de modificación) y asignarlo a este usuario en cuestión:

```
---creamos  un perfil sin permisos
insert into Perfil(Descripcion)
values ('Visitante')

select * from Perfil
update Usuario
set Id_Perfil = 4  where Id_Usuario = 2
```

Y obtendremos lo siguiente:

2	2	dmartinez	medico456	activo	4	87654321

4. Ahora, si necesitamos manejar permisos a nivel base de datos, es decir, no hacerlo manualmente, una buena opción sería trabajar con GRANT

Vemos que usuarios tenemos en la base:

Results Messages

	Id_Usuario	Usuario	Clave	Estado	Id_Perfil	NroDocumento
1	1	admin1	admin123	activo	1	12345678
2	2	dmartinez	medico456	activo	4	87654321
3	3	recepcion1	recepcion789	activo	3	56789012
4	4	ana.rod	usuario123	activo	3	23456789
5	5	miguel.san	medico456	activo	2	34567890
6	6	laura.fer	recepcion789	activo	3	45678901

Vemos que tenemos un usuario médico

5	5	miguel.san	medico456	activo	2	34567890

El médico solo necesita ver información sobre citas y pacientes, por lo que otorgaremos permisos de SELECT en las tablas relevantes

```
---otorgamos permisos especificos a diferentes usuarios
---medico
grant select on Cita to [miguel.san];
grant select on Paciente to [miguel.san];
grant select on Ficha_Medica to [miguel.san];
```

Vemos que tenemos 3 recepcionistas:

3	3	recepcion1	recepcion789	activo	3	56789012
4	4	ana.rod	usuario123	activo	3	23456789

6	6	laura.fer	recepcion789	activo	3	45678901

El recepcionista necesita gestionar citas, pero solo tiene permisos de lectura para pacientes.

```
--recep
select * from Usuario

-- Permisos para gestionar citas
grant select, insert, update, delete on Cita to [recepcion1];
grant select, insert, update, delete on Cita to [ana.rod];
grant select, insert, update, delete on Cita to [laura.fer];
```

```
-- Permiso solo de lectura para ver información de pacientes
grant select ON Paciente TO [recepcion1];
grant select ON Paciente TO [ana.rod];
grant select ON Paciente TO [laura.fer];
```

```
-- Permiso solo de lectura sobre las fichas médicas
--(si es necesario para ver antecedentes antes de crear o modificar una cita)
grant select Ficha_Medica TO [recepcion1];
grant select Ficha_Medica TO [ana.rod];
grant select Ficha_Medica TO [laura.fer];
```

Vemos que tenemos 1 Admin:

Results Messages

	Id_Usuario	Usuario	Clave	Estado	Id_Perfil	NroDocumento
1	1	admin1	admin123	activo	1	12345678

El admin necesita acceso completo a la base de datos para poder gestionar médicos, pacientes y citas.

```
---admin
grant all privileges on Medico to [admin1];
grant all privileges on Paciente to [admin1];
grant all privileges on Cita to [admin1];
grant all privileges on Ficha_medica to [admin1];
grant all privileges on Especialidad to [admin1];
```