PROYECTO DE ESTUDIO E INVESTIGACIÓN

UNIVERSIDAD NACIONAL DEL NORDESTE

FACULTAD DE CIENCAS EXCACTAS, NATURALES Y AGRIMENSURA

CÁTEDRA: BASES DE DATOS I

AÑO: 2025

AUTORES: Grupo 37

Gamba, Bruno

García Martin, Jorge Jonathan

Landriel, Gabriel

Rodríguez, Lucas Joel

PROFESOR:

CAPÍTULO I: INTRODUCCIÓN

1. Tema

El tema central del trabajo práctico es la implementación y validación de una arquitectura de base de datos relacional robusta y eficiente para un sistema de gestión transaccional. El problema sobre el cual se investiga es la necesidad de garantizar simultáneamente la seguridad de la información, la integridad de las operaciones y la eficiencia en la recuperación de datos en un entorno de negocio real.

El proyecto toma como base el modelo de datos de un Sistema de Gestión para una Clínica Veterinaria, donde se manipulan datos sensibles de pacientes **(Mascota)** y transacciones **(Turno, Certificado_Medico, Factura).**

Título del Trabajo Práctico:

Implementación y Evaluación de Arquitecturas Avanzadas de T-SQL para Garantizar Seguridad, Consistencia y Optimización en un Sistema de Gestión para Clínica Veterinaria.

2. Planteamiento del problema

El Trabajo Práctico se plantea como un problema de investigación aplicada que busca esclarecer cómo las herramientas avanzadas de gestión de bases de datos abordan los desafíos de la seguridad, la integridad y el rendimiento.

La implementación de un sistema transaccional como el de la Clínica Veterinaria exige que el DBMS pueda manejar concurrencia y fallos de manera segura, al mismo tiempo que ofrece un rendimiento analítico óptimo.

Declaración del Problema: ¿De qué manera la aplicación de herramientas avanzadas de T-SQL (Permisos, Transacciones, Procedimientos y Optimización de Consultas) puede garantizar la integridad, la seguridad y la eficiencia operativa en el sistema de gestión de la Clínica Veterinaria?

Preguntas de Investigación:

a. Seguridad (Tema 4): ¿Cómo se debe configurar la Autorización (Control) mediante roles y permisos (GRANT, REVOKE) para implementar el Principio de Mínimo Privilegio, asegurando que solo el personal autorizado acceda a la información sensible de los certificados médicos?
b. Integridad y Consistencia (Tema 3): ¿Cómo se pueden utilizar las transacciones para asegurar que un conjunto de operaciones se ejecute de manera atómica, garantizando que ante un fallo, los datos permanezcan consistentes?
c. Modularidad y Encapsulamiento (Tema 1): ¿Cómo contribuyen los procedimientos y funciones almacenadas a la estandarización de las operaciones y al encapsulamiento de la seguridad, permitiendo la ejecución de lógica de negocio sin otorgar permisos directos sobre las tablas base?
d. Eficiencia y Rendimiento (Tema 2): ¿Cómo se debe aplicar el concepto del Orden Lógico de Procesamiento (OLP) y la correcta indexación para evaluar y optimizar el rendimiento de consultas que operan sobre grandes lotes de datos, reduciendo los tiempos de respuesta?

3. Objetivos del trabajo práctico

Objetivo General

Desarrollar, implementar y evaluar la base de datos relacional para la Clínica Veterinaria, focalizando la implementación en la integración exitosa de los mecanismos de **Autorización(Tema 4)**, **Transacciones(Tema 3)**, **Modularidad (Tema 1)** y **Optimización(Tema 2)**, cumpliendo con los criterios de seguridad, integridad y eficiencia.

Objetivos Específicos

i. **Implementar la arquitectura de seguridad** basada en la jerarquía de Principales de Seguridad, Roles y Permisos, siguiendo el **Principio de Mínimo Privilegio**.

ii. **Modularizar y estandarizar** las operaciones DML mediante la **implementación y prueba de procedimientos y funciones almacenadas**, incluyendo la inserción de lotes de datos mediante estos.

iii. **Configurar permisos granulares** (utilizando GRANT) y el permiso **EXECUTE** sobre procedimientos almacenados para demostrar el encapsulamiento y el control de acceso diferenciado a nivel de usuarios y roles.

iv. **Implementar y validar transacciones T-SQL** que aseguren la **atomicidad y consistencia** de las operaciones DML compuestas, incluyendo el control explícito de fallos y *rollbacks*.

v. **Medir y comparar** la eficiencia de las consultas sobre grandes conjuntos de datos, evaluando el impacto de diferentes configuraciones de **índices** y documentando los planes de ejecución.

CAPÍTULO II: MARCO CONCEPTUAL

Este capítulo presenta los conceptos teóricos fundamentales que sustentan el desarrollo del Trabajo Práctico. El objetivo es situar el problema planteado en el Capítulo I dentro de un marco de conocimientos sólidos sobre bases de datos, seguridad, integridad, modularidad y optimización en motores SQL.

1. Bases de Datos Relacionales

Una base de datos relacional es un modelo estructurado basado en tablas, relaciones y restricciones formales. Este modelo organiza la información de manera que sea accesible mediante operaciones declarativas (SQL) y garantiza ACID en entornos transaccionales.

Características principales:

- Uso de tablas vinculadas mediante claves primarias y foráneas.
- Integridad referencial asegurada por el motor.
- Lenguaje SQL para manipulación y consulta.
- Independencia lógica y física de los datos.

2. Lenguaje T-SQL

T-SQL (Transact-SQL) es una extensión del estándar SQL utilizada por Microsoft SQL Server. Incorpora:

- Variables locales
- Control de flujo
- Procedimientos almacenados
- Funciones definidas por el usuario
- Manejo explícito de transacciones
- Manejo avanzado de errores (TRY/CATCH)

T-SQL permite implementar lógica de negocio compleja directamente dentro del motor de base de datos.

3. Seguridad en SQL Server: Principales, Roles y Permisos

La seguridad en SQL Server se basa en los siguientes elementos:

Principales (Principals)

Son entidades que pueden solicitar acceso al servidor:

- **Logins (nivel servidor)**
- **Usuarios (nivel base de datos)**
- **Roles (agrupación de permisos)**

Roles

Permiten asignar permisos de forma colectiva.

Se dividen en:

- **Roles fijos de servidor**
- **Roles fijos de base de datos**
- **Roles definidos por el usuario**

Permisos (GRANT, REVOKE, DENY)

Permiten controlar qué operaciones puede realizar un usuario sobre un objeto.

Ejemplos:

- **GRANT SELECT ON Mascota TO recepcionista**
- **REVOKE INSERT FROM invitado**
- **DENY DELETE TO pasante**

La filosofía aplicada es el **Principio de Mínimo Privilegio**, que busca otorgar únicamente los permisos estrictamente necesarios.

4. Transacciones: Atomicidad e Integridad

Una transacción es una unidad lógica de trabajo que agrupa una o varias instrucciones T-SQL.

Su propósito es asegurar que todas las operaciones se ejecuten de manera segura, consistente y controlada, especialmente en sistemas transaccionales como el de la Clínica Veterinaria.

Las transacciones garantizan las propiedades **ACID**:

Atomicidad:

La transacción se ejecuta por completo o no se ejecuta. Si ocurre un error, todo se revierte.

Consistencia:

El estado final de la base de datos siempre debe ser válido, cumpliendo restricciones y reglas de negocio.

Aislamiento:

Evita que transacciones concurrentes interfieran entre sí, dependiendo del nivel de aislamiento configurado.

Durabilidad:

Una vez ejecutado un COMMIT, los cambios quedan guardados de forma permanente incluso ante fallos del sistema.

Ejemplo genérico de manejo de transacciones

```
BEGIN TRY
    BEGIN TRAN;

        -- Operaciones DML
        -- INSERT / UPDATE / DELETE
        -- Validaciones

    COMMIT;   -- Si todo salió bien
END TRY
BEGIN CATCH
    ROLLBACK; -- Si ocurre un error, se revierte todo

    -- Mensaje de error opcional
    SELECT ERROR_MESSAGE() AS Error;
END CATCH;
```

En entornos reales (como una clínica veterinaria), su uso es esencial para garantizar consistencia durante operaciones críticas como facturación o registro de turnos.

5. Procedimientos Almacenados (Stored Procedures)

Los procedimientos almacenados (SP) son bloques de código T-SQL precompilado que encapsulan operaciones repetitivas o sensibles dentro del motor SQL Server.

Ventajas principales:

- Reutilización de lógica de negocio.
- Reducción del tráfico entre cliente y servidor.
- Permiten validar datos antes de ejecutar operaciones DML.
- Mejoran la seguridad mediante permisos EXECUTE.
- Reducen el riesgo de inyección SQL.

En este proyecto, los SP permiten estandarizar operaciones como el alta, modificación y baja lógica de mascotas, evitando que los usuarios interactúen directamente con las tablas base.

6. Funciones Definidas por el Usuario (UDF)

Las funciones almacenadas devuelven siempre un valor, ya sea:

- Escalar: retorna un único dato (INT, VARCHAR, DATE, etc.).
- De tabla: retorna un conjunto de filas, similar a una vista parametrizada.

Se utilizan para cálculos y operaciones reutilizables, como:

- Obtener la edad de una mascota.
- Contar cuántos registros activos tiene un cliente.
- Generar descripciones legibles o formateadas.

Las UDF centralizan la lógica de consulta y evitan duplicación de código en distintas partes del sistema.

7. Optimización de Consultas y Orden Lógico de Procesamiento (OLP)

El Orden Lógico de Procesamiento (OLP) describe cómo SQL Server evalúa una consulta internamente, lo cual no coincide con el orden en el que el usuario escribe la sentencia.

Orden conceptual del OLP:

1. FROM
2. ON
3. JOIN
4. WHERE
5. GROUP BY
6. HAVING
7. SELECT
8. ORDER BY

Comprender este orden permite:

- Evitar errores comunes como usar columnas no disponibles en determinadas fases.
- Escribir consultas más eficientes y predecibles.
- Reducir operaciones costosas en tiempo y memoria.

Además, SQL Server mejora los tiempos de respuesta utilizando índices, que agilizan:

- Búsquedas
- Ordenamientos
- Filtrados

Una correcta indexación es fundamental cuando se trabaja con grandes volúmenes de datos.

8. Integridad de Datos

La integridad asegura que los datos permanezcan correctos, coherentes y válidos en todo momento.

Se implementa mediante:

- Claves primarias
- Claves foráneas
- Restricciones CHECK
- Campos NOT NULL
- Reglas de actualización y eliminación en cascada
- Triggers

Estos mecanismos garantizan que la base de datos mantenga su coherencia incluso ante operaciones complejas o simultáneas.

9. Baja Lógica

La baja lógica consiste en marcar un registro como inactivo (baja = 1) en lugar de eliminarlo físicamente de la tabla.

Ventajas:

- Mantiene trazabilidad histórica.
- Evita la pérdida irreversible de información.
- Facilita auditorías.
- Reduce riesgos de romper integridad referencial.

Es un patrón ampliamente utilizado en sistemas que manejan información sensible o que requieren conservar historial de las operaciones.

Con estos conceptos se establecen las bases teóricas necesarias para comprender la implementación detallada en los capítulos siguientes.

CAPÍTULO III: METODOLOGÍA SEGUIDA

Este capítulo describe el proceso seguido para la realización del Trabajo Práctico, detallando las actividades ejecutadas, las herramientas utilizadas y la organización metodológica del equipo. El objetivo es proporcionar una visión clara del enfoque aplicado para integrar los cuatro temas: **procedimientos y funciones almacenadas, índices, transacciones y permisos.**

1. Descripción de cómo se realizó el Trabajo Práctico

El trabajo se realizó siguiendo una metodología incremental, práctica y colaborativa, combinando investigación teórica, desarrollo técnico y verificación experimental dentro del motor SQL Server.

Análisis del modelo de datos existente

Se inició con una revisión de la estructura de la base de datos de la Clínica Veterinaria.

Se identificaron las entidades principales (Mascota, Cliente, Raza, Turno, Certificado_Medico, Factura) y se determinó cuáles serían utilizadas para cada tema.

Investigación teórica individual

Cada integrante del equipo investigó su tema asignado:

- Tema 1: Procedimientos y funciones almacenadas.
- Tema 2: Índices y optimización mediante OLP.
- Tema 3: Transacciones (exitosas, fallidas y anidadas).
- Tema 4: Permisos mediante usuarios y roles.

El contenido teórico se integró al Marco Conceptual para asegurar coherencia y correlación entre temas.

Desarrollo práctico dentro de SQL Server

Cada tema incluyó la implementación de scripts reales:

- Procedimientos CRUD y funciones UDF aplicados a la tabla Mascota.
- Índices agrupados, no agrupados y filtrados sobre la tabla Turno.
- Transacciones explícitas con COMMIT y ROLLBACK, además de transacciones anidadas.
- Configuración de seguridad mediante creación de logins, usuarios y roles con GRANT, REVOKE y DENY.

Las pruebas se realizaron en la base DB_Integrador_Grupo37, utilizando datos existentes y datos generados.

Trabajo colaborativo utilizando Git y GitHub

El equipo trabajó de forma conjunta usando control de versiones:

- Cada integrante utilizó su rama personal.
- Se realizaron commits descriptivos.
- Las ramas se integraron al repositorio mediante pull requests.
- Se resolvieron conflictos para mantener la coherencia de la carpeta de documentación y scripts.
- Cada tema quedó alojado en su carpeta correspondiente (Tema 1, Tema 2, Tema 3, Tema 4).

Esto permitió mantener orden y trazabilidad del trabajo realizado.

Validación y recolección de evidencias

Cada implementación se probó varias veces para verificar su funcionamiento. Se recopilaron:

- Resultados antes y después de transacciones.
- Rendimiento de consultas con y sin índices.
- Ejecución correcta de procedimientos con distintos permisos.
- Tiempos medidos mediante SET STATISTICS IO / TIME.

- Inserción de lotes de datos tanto directos como usando SP.
- Capturas de pantalla para documentar resultados.

Todas las evidencias fueron incorporadas a la documentación final.

2. Herramientas (Instrumentos y procedimientos)

SQL Server Management Studio (SSMS)

Usado para:

- Ejecutar scripts T-SQL
- Analizar planes de ejecución
- Configurar usuarios y roles
- Verificar resultados de transacciones e índices

Git y GitHub

Herramientas clave de colaboración:

- Control de versiones mediante ramas
- Gestión de conflictos
- Integración del trabajo con pull requests
- Historial claro y organizado del desarrollo

Fuentes de información consultadas

El marco teórico se apoyó en:

- Documentación oficial de SQL Server
- Material teórico proporcionado por la cátedra
- Bibliografía técnica
- Artículos y blogs de especialistas
- Pruebas experimentales dentro del motor

Procedimientos de recolección y validación

Se utilizaron métodos como:

- Ejecuciones repetidas de pruebas
- Generación de datos masivos para índices
- Verificación visual mediante SELECTs comparativos
- Pruebas con distintos usuarios para validar permisos
- Comparación de tiempos y estadísticas del motor

Criterios de documentación

Para el informe final se siguieron criterios de:

- Claridad técnica
- Normalización de capítulos
- Coherencia con la introducción y el marco teórico
- Inclusión equilibrada de imágenes y código
- Lenguaje formal y académico

3. Conclusión del capítulo

La metodología aplicada permitió abordar el trabajo de manera ordenada y progresiva.

La combinación de teoría, práctica, experimentación y validación colaborativa garantizó que los cuatro temas se integraran coherentemente dentro de un mismo sistema real, cumpliendo los objetivos planteados en la introducción.

CAPÍTULO IV: DESARROLLO DEL TEMA / PRESENTACIÓN DE RESULTADOS

En este capítulo se presentan los resultados obtenidos a partir del desarrollo práctico de los cuatro temas asignados:

1. Procedimientos y funciones almacenadas
2. Optimización de consultas mediante índices
3. Manejo de transacciones y transacciones anidadas
4. Manejo de permisos a nivel de usuarios y roles

La información se expone de manera objetiva, mostrando los datos, pruebas, gráficos, capturas y scripts que evidencian el desarrollo técnico realizado sobre la base de datos del sistema veterinario.

1. Procedimientos y Funciones Almacenadas

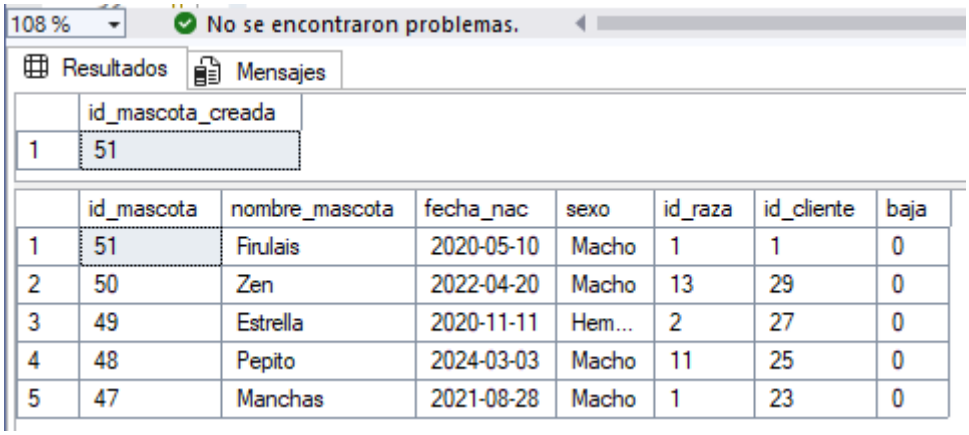
Implementación de procedimientos CRUD sobre Mascota

Se desarrollaron tres procedimientos almacenados para la gestión de la tabla Mascota.

Las pruebas se realizaron sobre registros reales de la base de datos.

Inserción de Mascotas

Se ejecutó el procedimiento usp_InsertarMascota, validando correctamente cliente, raza, sexo y fecha.


108 % No se encontraron problemas.

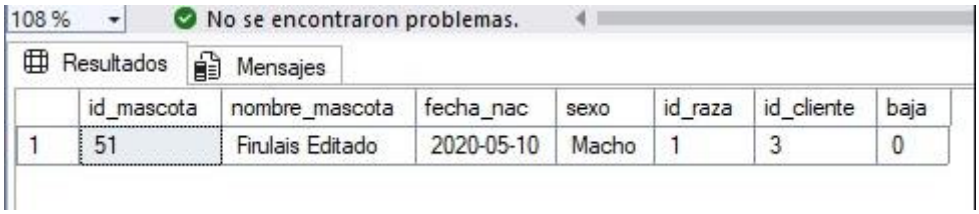
Resultados Mensajes

	id_mascota_creada
1	51

	id_mascota	nombre_mascota	fecha_nac	sexo	id_raza	id_cliente	baja
1	51	Firulais	2020-05-10	Macho	1	1	0
2	50	Zen	2022-04-20	Macho	13	29	0
3	49	Estrella	2020-11-11	Hem...	2	27	0
4	48	Pepito	2024-03-03	Macho	11	25	0
5	47	Manchas	2021-08-28	Macho	1	23	0

Modificación de Mascotas

Se probó el procedimiento usp_ModificarMascota sobre un registro existente, mostrando la correcta actualización.


108 % No se encontraron problemas.

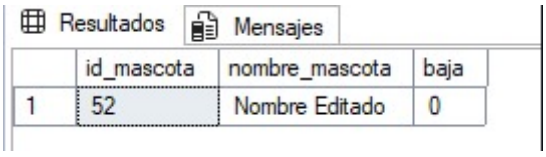
Resultados Mensajes

	id_mascota	nombre_mascota	fecha_nac	sexo	id_raza	id_cliente	baja
1	51	Firulais Editado	2020-05-10	Macho	1	3	0

Baja lógica

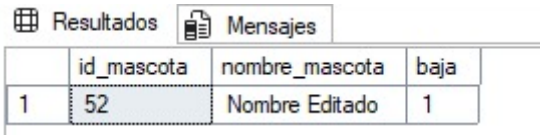
Se utilizó el procedimiento usp_BajaLogicaMascota, marcando el campo baja = 1.

Tabla antes de la baja lógica:


Resultados Mensajes

	id_mascota	nombre_mascota	baja
1	52	Nombre Editado	0

Después de la baja lógica:


Resultados Mensajes

	id_mascota	nombre_mascota	baja
1	52	Nombre Editado	1

Funciones Almacenadas

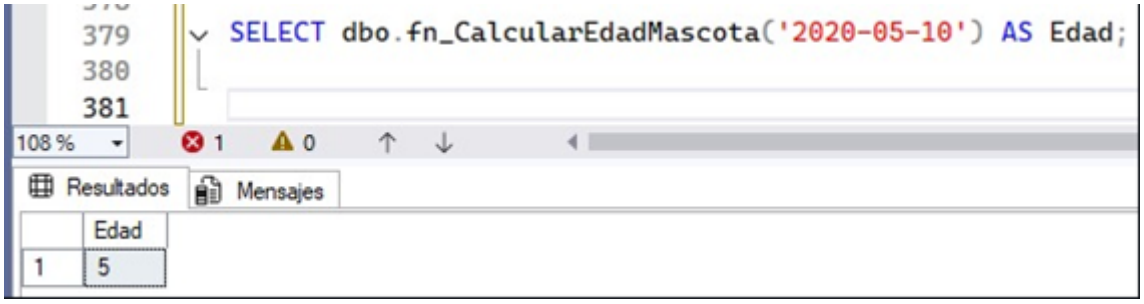
Se implementaron tres funciones:

fn_CalcularEdadMascota

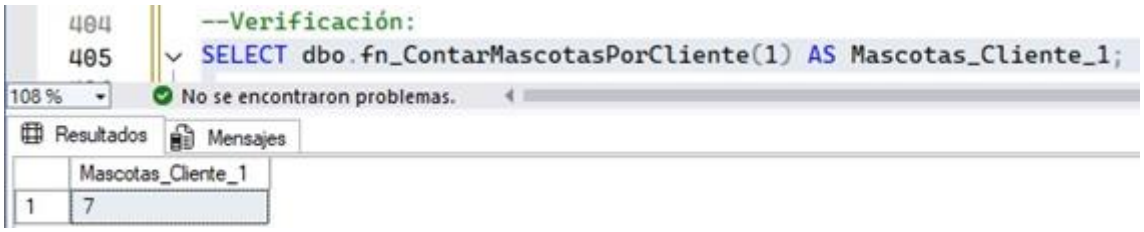
fn_ContarMascotasPorCliente

fn_MascotaDescripcion

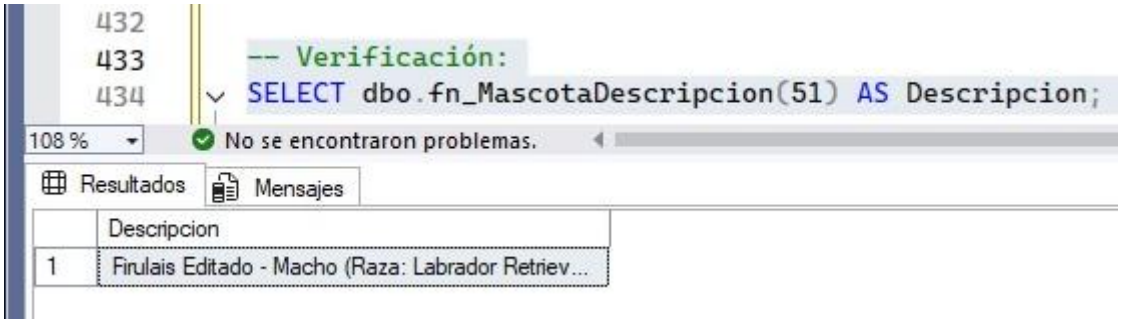
Resultado edad mascota:



Resultado contar mascotas por cliente:



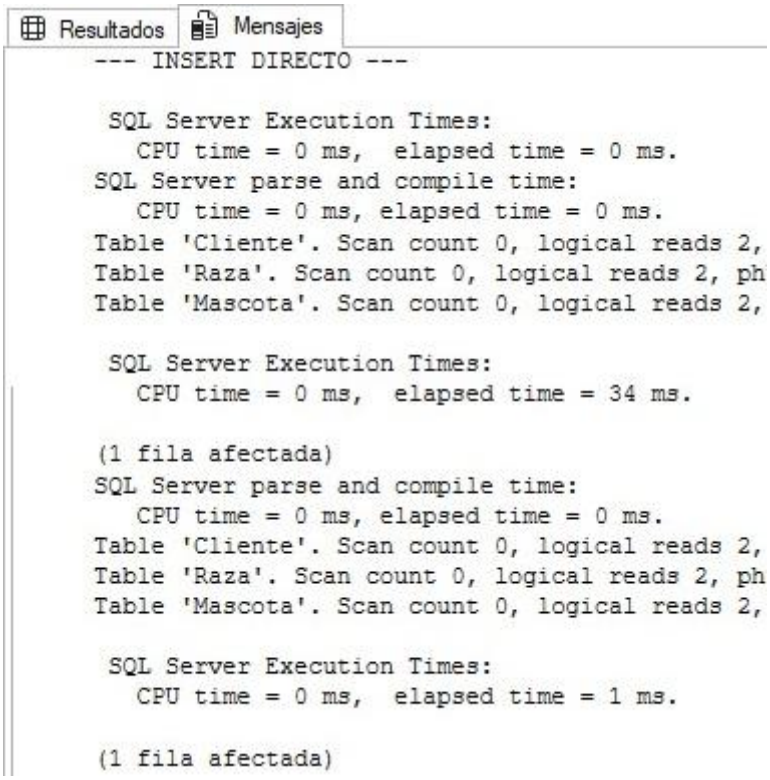
Resultado descripción mascota:



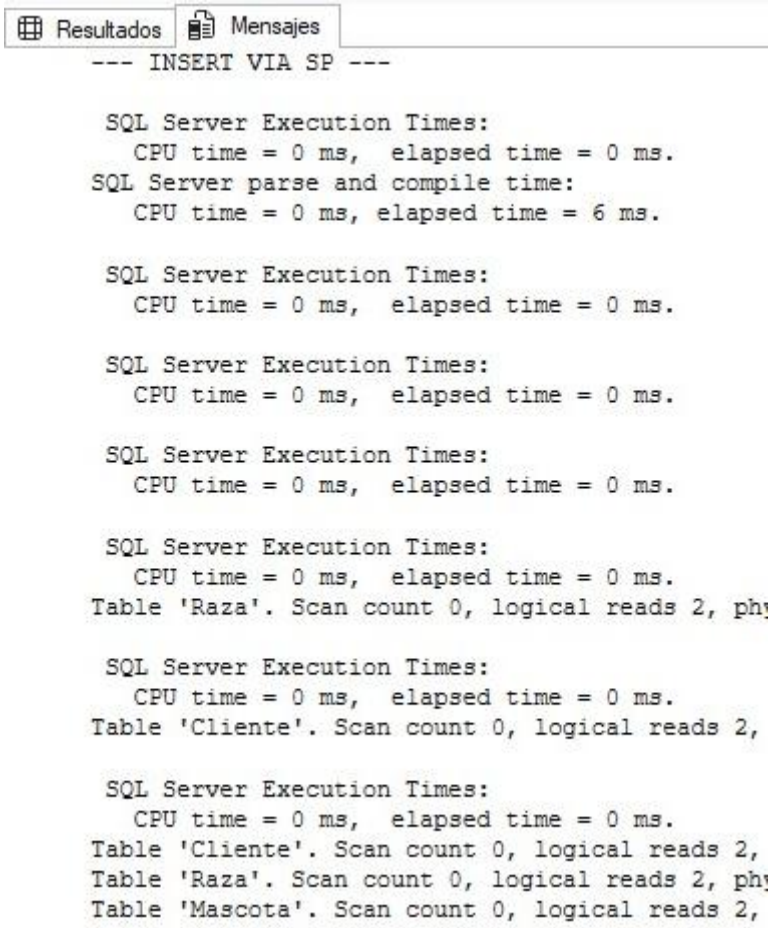
Comparativa de rendimiento: inserción directa vs SP

Se activaron métricas de SQL Server (STATISTICS IO/TIME) y se realizaron dos operaciones equivalentes:

Resultado del INSERT directo:



Resultado del INSERT vía SP:



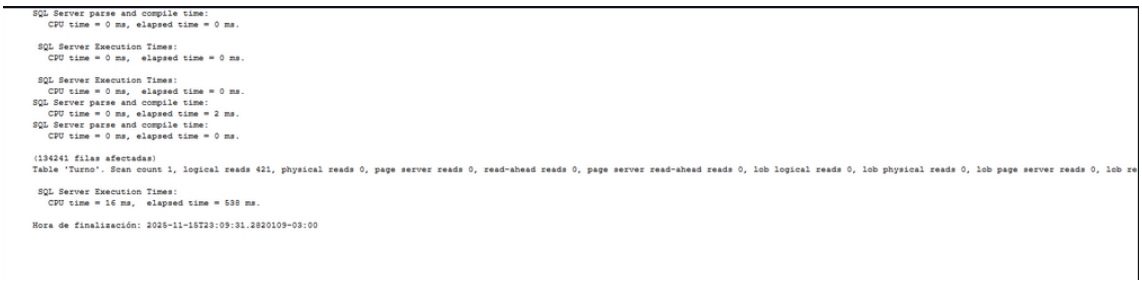
Hallazgos:

- La diferencia en milisegundos es mínima.
- El SP realiza más validaciones, lo cual se refleja en el consumo de CPU.
- La ejecución mediante SP centraliza la lógica y evita errores.

2. Optimización de Consultas mediante Índices

Se evaluó el rendimiento de una consulta sobre la tabla **Turno** con 200.000 registros, en tres escenarios:

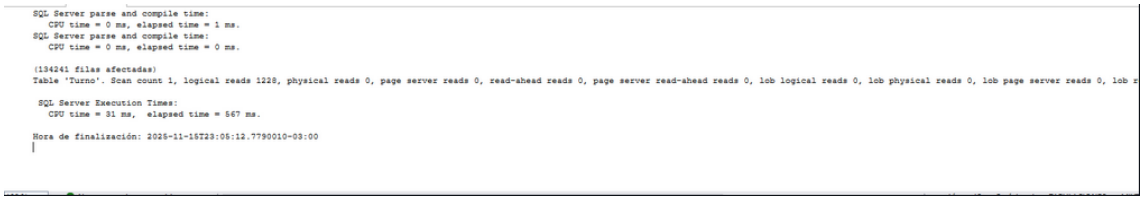
Consulta sin índice:



Se observa:

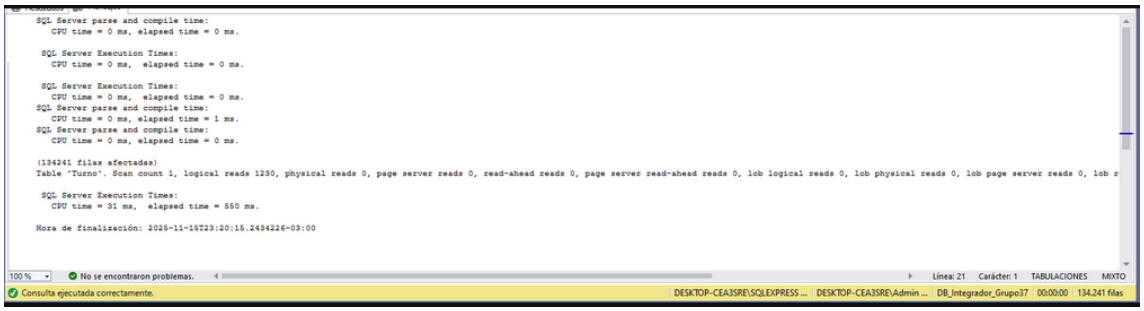
- Lecturas lógicas elevadas
- Mayor tiempo de CPU
- Búsqueda completa de tabla

Consulta con índice agrupado:



Se reducen las lecturas lógicas, mejorando la eficiencia del acceso.

Consulta con índice agrupado con columnas incluidas:



Se obtiene la mejor lectura lógica al cubrir todas las columnas necesarias para la consulta.

Resultados observados

- Las búsquedas *con* índices con columnas incluidas reducen accesos a la tabla.
- La consulta con índice agrupado con columnas incluidas consume menos CPU.
- Sin índices, SQL Server debe leer la tabla completa.

3. Transacciones y Transacciones Anidadas

Transacción exitosa

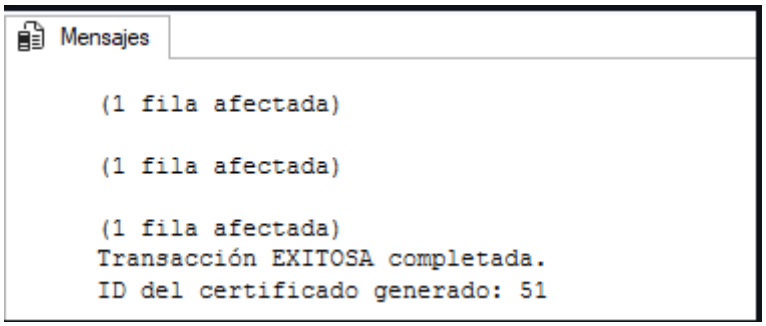
Se implementó un bloque transacional que:

1. Inserta un certificado
2. Inserta medicamento aplicado
3. Actualiza stock
4. Confirma con COMMIT

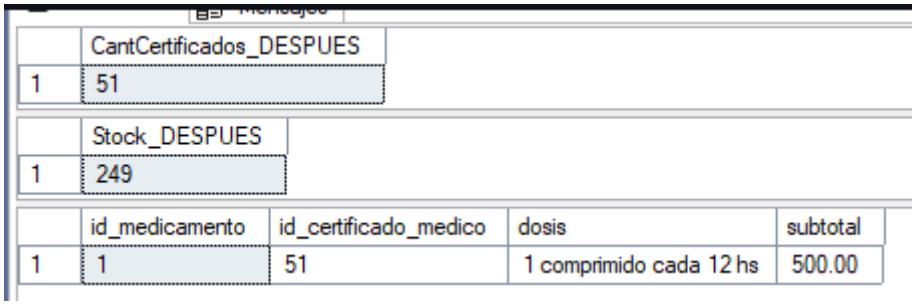
Antes de ejecutar la transacción, se obtuvieron métricas iniciales**.**

Certificados antes, stock antes:

Ejecución exitosa:



Valores después del commit:



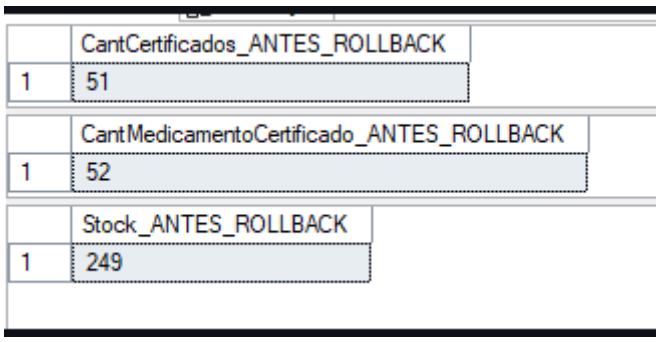
	CantCertificados_DESPUES
1	51

	Stock_DESPUES
1	249

	id_medicamento	id_certificado_medico	dosis	subtotal
1	1	51	1 comprimido cada 12 hs	500.00

Transacción fallida (con rollback)

Se forzó un error mediante un stock negativo para validar comportamiento ACID.

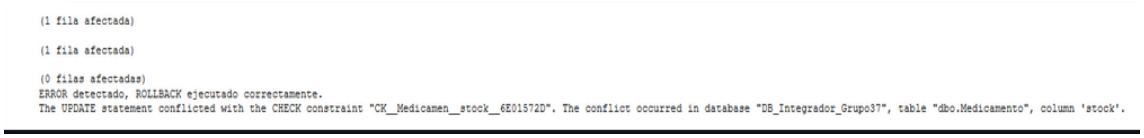


	CantCertificados_ANTES_ROLLBACK
1	51

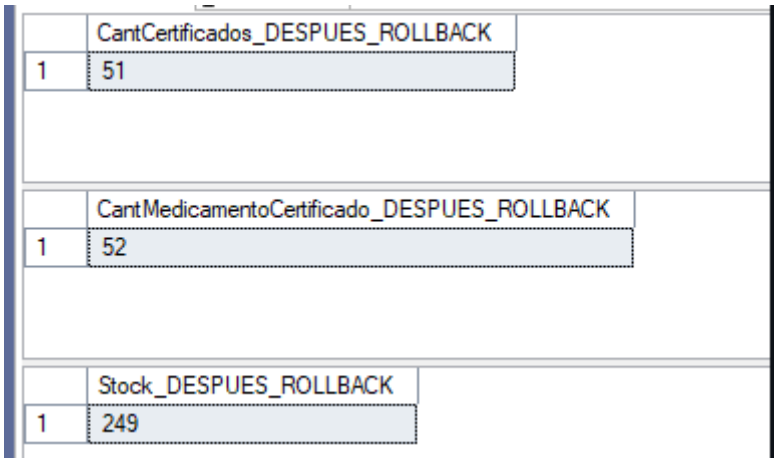
	CantMedicamentoCertificado_ANTES_ROLLBACK
1	52

	Stock_ANTES_ROLLBACK
1	249

Error intencional y mensaje de rollback:



Estado posterior (sin cambios aplicados):



	CantCertificados_DESPUES_ROLLBACK
1	51

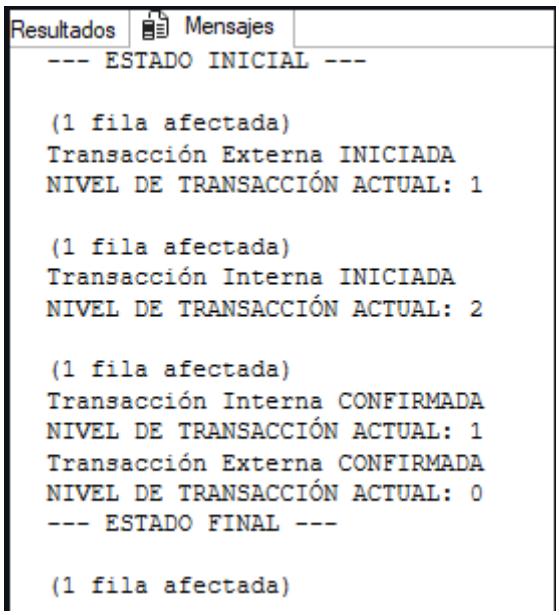
	CantMedicamentoCertificado_DESPUES_ROLLBACK
1	52

	Stock_DESPUES_ROLLBACK
1	249

Transacción anidada

Se mostró el comportamiento del contador @@TRANCOUNT.

Impresión de niveles de transacción (1 → 2 → 1):



Resultados | Mensajes

```
--- ESTADO INICIAL ---

(1 fila afectada)
Transacción Externa INICIADA
NIVEL DE TRANSACCIÓN ACTUAL: 1

(1 fila afectada)
Transacción Interna INICIADA
NIVEL DE TRANSACCIÓN ACTUAL: 2

(1 fila afectada)
Transacción Interna CONFIRMADA
NIVEL DE TRANSACCIÓN ACTUAL: 1
Transacción Externa CONFIRMADA
NIVEL DE TRANSACCIÓN ACTUAL: 0
--- ESTADO FINAL ---

(1 fila afectada)
```

Hallazgo:

Solo la transacción externa confirma los cambios, la interna solo incrementa el contador.

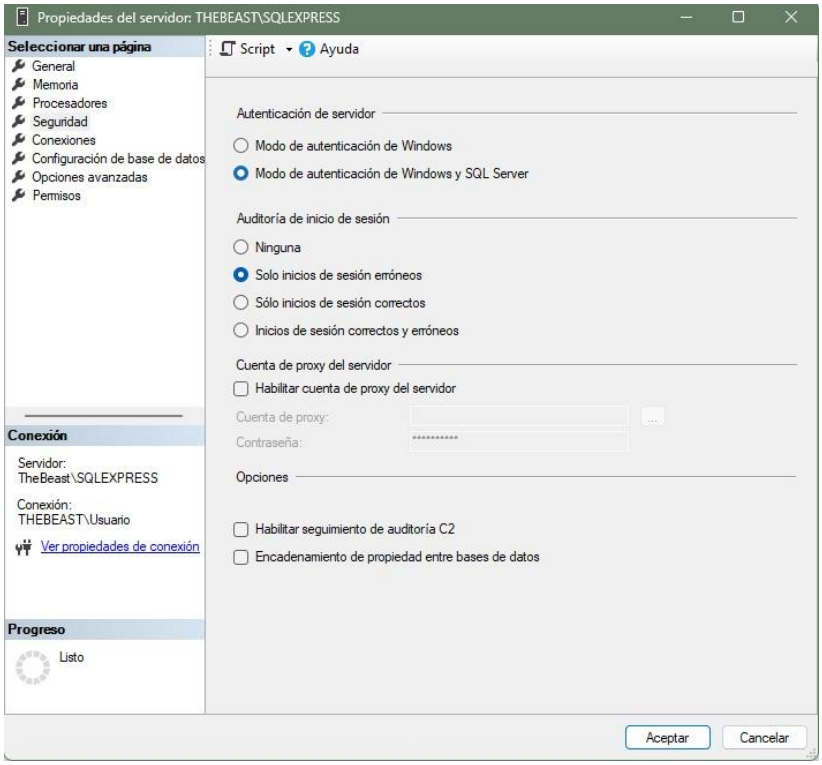
4. Manejo de Permisos a Nivel de Usuarios y Roles

Configuración y creación de usuarios.

Se habilitó el modo mixto y se crearon dos logins:

- Login_Admin
- Login_Lector

Verificación de modo mixto:



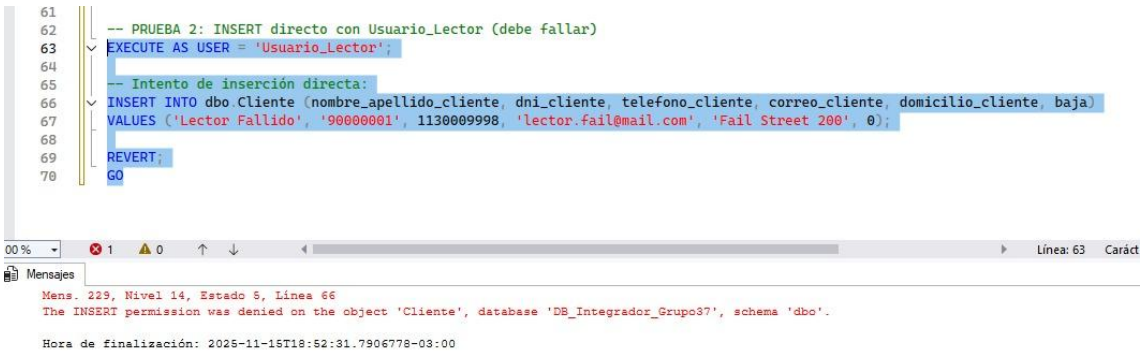
Inicio sesión con usuario sa:

Prueba con principio de mínimo privilegio

Se otorgó:

- **Usuario_Admin → CONTROL**
- **Usuario_Lector → SELECT sobre Cliente**

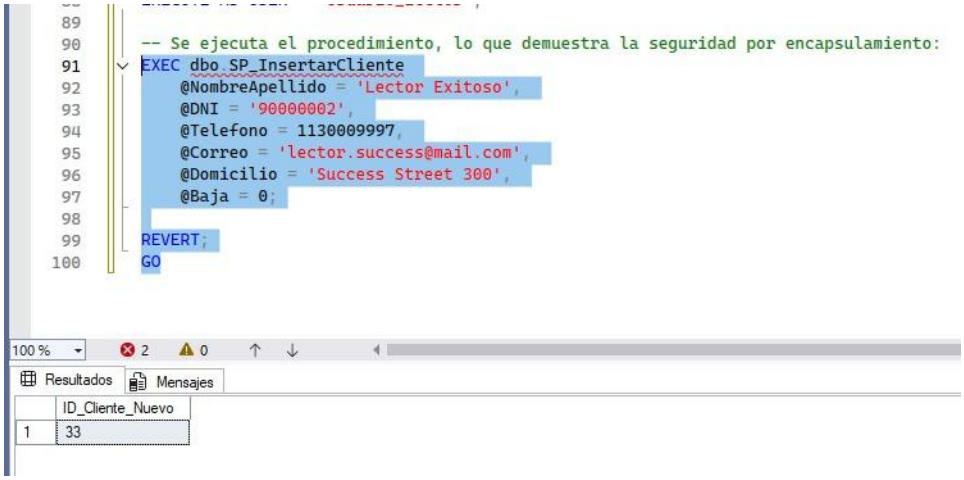
Prueba A: intento de INSERT directo:



Prueba de seguridad por encapsulamiento

Al usuario lector se le otorgó permission de EXECUTE sobre el SP:

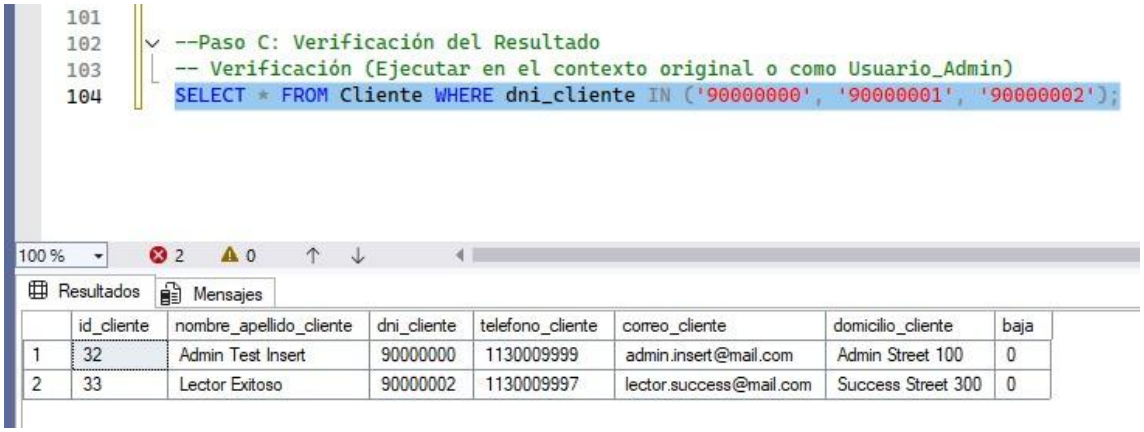
Prueba B: INSERT vía SP (funciona):



Pruebas con roles

Se creó el rol Rol_Lector. Solo **Usuario_a** fue agregado.

Prueba A: Usuario_a ejecuta SELECT:



Prueba B: Usuario_b ejecuta SELECT:

```
47
48    -- Intento de lectura de datos sensibles (Certificado_Medico)
49    SELECT TOP 5 *
50    FROM dbo.Certificado_Medico;
51
52    REVERT;
53    GO
```

100 % No se encontraron problemas.

Resultados Mensajes

	id_certificado_medico	fecha_emision	observacion	diagnostico	id_turno	id_usuario
1	1	2025-11-15	Mascota en buen estado general. Vacuna al día.	Chequeo anual OK	1	2
2	2	2025-11-15	Retiro de puntos sin complicaciones. Herida limp...	Post-Operatorio Limpio	2	3
3	3	2025-11-15	Tos seca. Se descarta cuerpo extraño.	Faringitis leve	3	4
4	4	2025-11-15	Se aplica antiparasitario interno. Cliente informado.	Desparasitación Rutinaria	4	2
5	5	2025-11-15	Piel y pelaje sano. Peso ideal.	Control de Rutina	5	3

Hallazgo:

Los permisos asignados al rol se heredan correctamente a sus miembros.

CAPÍTULO V: CONCLUSIONES

A lo largo del desarrollo de este Trabajo Práctico pudimos analizar, aplicar y comprobar distintos conceptos fundamentales del motor SQL Server. Cada uno de los cuatro temas requirió no solo implementar código, sino también observar su comportamiento real dentro del sistema de la veterinaria. Esto nos permitió entender cómo funcionan en conjunto los procedimientos almacenados, los índices, las transacciones y la gestión de permisos, y cómo influyen directamente en la calidad, seguridad y rendimiento de una base de datos.

En el caso de los **procedimientos y funciones almacenadas**, pudimos ver que encapsular la lógica mejora la organización del sistema y evita errores al validar los datos antes de insertarlos. Comprobamos que un SP rechaza datos inválidos y que una función puede reutilizarse en distintas consultas reforzó la importancia de centralizar reglas de negocio en el motor.

En el tema de **índices**, los resultados mostraron claramente la diferencia entre consultar una tabla sin índices y hacerlo con distintos tipos de índices aplicados. Aunque en las pruebas los tiempos variaban por pocos milisegundos, entendimos que en tablas más grandes esto tiene un impacto enorme en el rendimiento general del sistema. Esto confirma que el diseño de índices no es un agregado opcional, sino parte clave de la optimización.

Con respecto al **manejo de transacciones**, pudimos comprobar el cumplimiento del modelo ACID en situaciones reales. Tanto la transacción exitosa como la fallida mostraron que SQL Server garantiza la integridad de los datos incluso frente a errores. Además, observar el funcionamiento de @@TRANCOUNT en una transacción anidada nos ayudó a entender mejor cómo se organizan internamente las operaciones del motor.

Finalmente, en el tema de **permisos y roles**, las pruebas demostraron que la seguridad no solo depende de limitar accesos, sino de hacerlo de forma ordenada y escalable. Ver cómo un usuario sin privilegios podía ejecutar una operación compleja solo a través de un procedimiento almacenado nos permitió entender la importancia del encapsulamiento. Y al usar roles, comprobamos que administrar permisos de manera grupal simplifica mucho la gestión y reduce errores.

En conjunto, consideramos que los objetivos del trabajo fueron alcanzados. No solo por implementar los temas solicitados, sino que pudimos interpretar los resultados y comprender por qué cada concepto es necesario dentro de un sistema real. Este proyecto nos permitió conectar teoría con práctica y entender cómo cada decisión técnica impacta en el funcionamiento, rendimiento y seguridad de la base de datos.

CAPITULO VI: BIBLIOGRAFÍA

Para el desarrollo de los temas asignados, se utilizó como referencia principal la documentación oficial de SQL Server, donde se consultaron los apartados relacionados a los mismos. También se empleó material brindado por la cátedra a través del aula virtual institucional.

Sitios consultados:

Documentación oficial de Microsoft SQL Server:

https://learn.microsoft.com/en-us/sql/sql-server/?view=sql-server-ver17

Aula virtual de la UNNE – Material de la asignatura Base de Datos I (Bibliografía):

https://virtual-moodle.unne.edu.ar/mod/page/view.php?id=834956#bibliografia