



INFORME DE TALLER

I. PORTADA

Tema: Transacciones
Unidad de Organización Curricular: PROFESIONAL
Nivel y Paralelo: 5 A
Alumnos participantes:
Analuiza Castillo Jimmy Sebastián
Gordillo Guevara Luis Josué
Manobanda Chango Ana Patricia
Peñaloza Narváez Johnny Alexander
Sistemas de Base de Datos Distribuidos
Ing. José Caiza
Asignatura:
Docente:

II. INFORME DE TALLER

2.1 Objetivos

General:

Implementar y validar operaciones de bases de datos distribuidas en SQL Server, aplicando transacciones, integridad referencial y manejo de errores para garantizar la consistencia de los datos.

Específicos:

- Crear y gestionar transacciones atómicas para asegurar que operaciones múltiples se completen exitosamente o se reviertan en caso de error.
- Verificar la integridad referencial entre tablas relacionadas mediante claves primarias y foráneas.
- Manejar errores con TRY-CATCH para controlar excepciones durante operaciones críticas.

2.2 Modalidad

- Práctica guiada en laboratorio con supervisión del docente.
- Individual o en equipos de 2 personas (según indicaciones del profesor).

2.3 Duración

- **Presenciales:** 2 horas (tiempo en clase).
- **No presenciales:** 1 hora adicional para redacción del informe.

2.4 Instrucciones

1. Conectarse a SQL Server Management Studio (SSMS).
2. Ejecutar scripts SQL proporcionados en la guía.
3. Documentar resultados y capturas de pantalla.
4. Validar cada paso con el docente antes de continuar.

2.5 Listado de materiales

Listado de equipos y materiales generales empleados en la guía práctica:

- Computadora con Windows/Linux/MacOS
- SQL Server Management Studio (SSMS).
- Acceso a la base de datos CentroMedicoBD.

TAC (Tecnologías para el Aprendizaje y Conocimiento) empleados en la guía práctica:

- Plataformas educativas
- Simuladores y laboratorios virtuales
- Aplicaciones educativas
- Recursos audiovisuales



- Gamificación
 Inteligencia Artificial

2.6 Desarrollo de la actividad

1. Creación de la base de datos CentroMedico

En esta sección se describe el proceso para la creación de la base de datos CentroMedico en SQL Server, utilizando SQL Server Management Studio (SSMS). En la siguiente imagen (Fig 1) se presenta el modelo ER final de la base de datos.

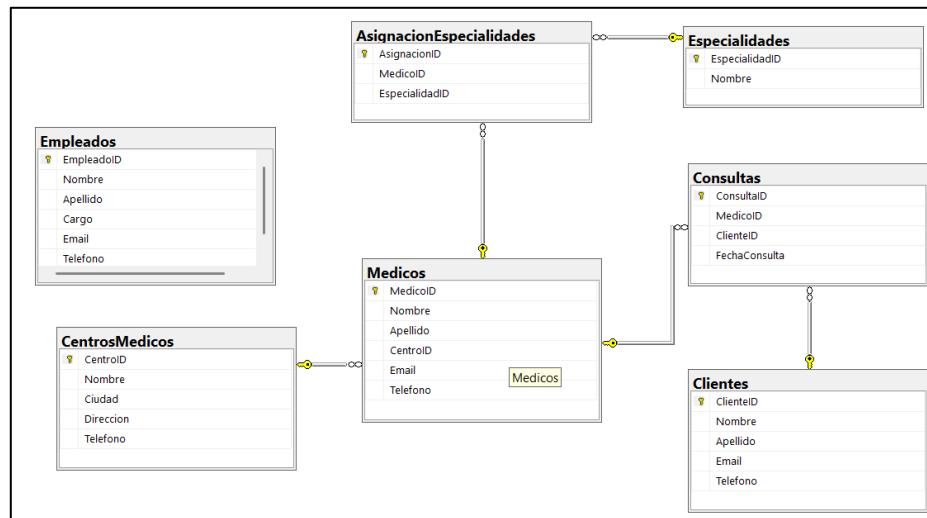


Fig. 1. Modelo ER de la base de datos CentroMedico

2. Verificación de conexión y estructura de datos

Abrir SQL Server Management Studio (SSMS) y conectarse al servidor.

Seleccionar la base de datos CentroMedicoBD.

Verificar las tablas existentes ejecutando:

```
SELECT * FROM INFORMATION_SCHEMA.TABLES;
```

Como se puede observar en la siguiente imagen (Fig 2).

La captura de pantalla muestra una ejecución de SQL en SSMS. El código ejecutado es:

```
USE CentroMedicoBD;
GO
SELECT * FROM INFORMATION_SCHEMA.TABLES;
```

El resultado es una tabla que enumera las tablas existentes en el esquema dbo:

TABLE_CATALOG	TABLE_SCHEMA	TABLE_NAME	TABLE_TYPE
1	dbo	CentrosMedicos	BASE TABLE
2	dbo	Especialidades	BASE TABLE
3	dbo	Medicos	BASE TABLE
4	dbo	Clientes	BASE TABLE
5	dbo	Empleados	BASE TABLE
6	dbo	Consultas	BASE TABLE
7	dbo	AsignacionEspecialidades	BASE TABLE
8	dbo	sysdiagrams	BASE TABLE

Fig. 2. Visualización de las tablas dentro de CentroMédico.

3. Verificar integridad referencial (IR) con datos existentes

Ejemplo 1 (Fig 3): Inserción de un médico con una especialidad inexistente.



UNIVERSIDAD TÉCNICA DE AMBATO
FACULTAD DE INGENIERÍA EN SISTEMAS ELECTRÓNICA E INDUSTRIAL
CARRERA DE TECNOLOGÍAS DE LA INFORMACIÓN
CICLO ACADÉMICO: AGOSTO 2025 – ENERO 2026



```
78  v INSERT INTO Medicos
79    (Nombre, Apellido, EspecialidadID, CentroID, Email, Telefono)
80    VALUES
81    ('Marta', 'Vega', 1, 1, 'marta.vega@clinicanorte.ec', '0991111111')
82

Línea: 81  Carácter: 1  SP
mensajes
Mens. 547, Nivel 16, Estado 0, Línea 78
The INSERT statement conflicted with the FOREIGN KEY constraint "FK_Medicos_Especialidades". The conflict
The statement has been terminated.
```

Fig. 3. Error al intentar Insertar un Médico con una especialidad Inexistente.

Ejemplo 2 (Fig 4): Inserción de un médico con una especialidad existente.

```
96
97  v INSERT INTO Medicos
98    (Nombre, Apellido, EspecialidadID, CentroID, Email, Telefono)
99    VALUES
100   ('Marta', 'Vega', 1, 1, 'marta.vega@clinicanorte.ec', '0991111111');
101

Línea: 103  Carácter: 1  SP
mensajes
(1 fila afectada)
```

Fig. 4. Inserción de Médico con Especialidad Existente.

Ejemplo 3: Eliminar una especialidad en uso (Pediatría ID=1) (Fig 5).

DELETE FROM Especialidades WHERE EspecialidadID = 1;

```
139  ||| -- VERIFICACION DE IR
140  |  DELETE FROM Especialidades WHERE EspecialidadID = 1;
141

%  ✓ No se encontraron problemas.  Línea: 140
mensajes
Mens. 547, Nivel 16, Estado 0, Línea 140
The DELETE statement conflicted with the REFERENCE constraint "FK_Medicos_Especialidades".
```

Fig. 5. Error al intentar borrar una Especialidad con Médicos asignados.

4. Transacciones con atomicidad (COMMIT/ROLLBACK)

Transacción exitosa: Crear un centro médico y asignar un médico (Fig 6).}

BEGIN TRANSACTION;

INSERT INTO CentrosMedicos (Nombre, Ciudad, Direccion, Telefono)
VALUES ('Clínica Este', 'Puyo', 'Av. Principal 123', '0987408137');

DECLARE @NuevoCentroID INT = SCOPE_IDENTITY();

INSERT INTO Medicos (Nombre, Apellido, EspecialidadID, CentroID, Email)
VALUES ('John', 'Peñaloza', 1, @NuevoCentroID, 'jpenaloza@clinicanorte.ec');

COMMIT TRANSACTION;



UNIVERSIDAD TÉCNICA DE AMBATO
FACULTAD DE INGENIERÍA EN SISTEMAS ELECTRÓNICA E INDUSTRIAL
CARRERA DE TECNOLOGÍAS DE LA INFORMACIÓN
CICLO ACADÉMICO: AGOSTO 2025 – ENERO 2026



```
145 BEGIN TRANSACTION;
146 ✓ INSERT INTO CentrosMedicos (Nombre, Ciudad, Direccion, Telefono)
147 VALUES ('Clínica Este', 'Puyo', 'Av. Principal 123', '0987408137');
148 DECLARE @NuevoCentroID INT = SCOPE_IDENTITY();
149 ✓ INSERT INTO Medicos (Nombre, Apellido, EspecialidadID, CentroID, Email)
150 VALUES ('John', 'Peñaloza', 1, @NuevoCentroID, 'jpenaloza@clinicanoorte.ec');
151 COMMIT TRANSACTION;
152

[+] No se encontraron problemas. ▶ Línea: 145 Carácter: 1 SPC

mensajes

(1 fila afectada)

(1 fila afectada)

Hora de finalización: 2025-10-03T10:21:29.2825336-05:00
```

Fig. 6. Creación de Centro Médico y asignación de médico en la misma Transacción.

Verificación de datos (Fig. 7)

```
152
153 -- VERIFICACION DE TRANSACCION
154 SELECT * FROM CentrosMedicos WHERE Nombre = 'Clínica Este';
155
156 ✓ SELECT *
157 FROM Medicos
158 WHERE Nombre = 'John'
159 AND CentroID IN (SELECT CentroID
160 FROM CentrosMedicos
161 WHERE Nombre = 'Clínica Este');

162

[+] No se encontraron problemas. ▶ Línea: 162

Resultados Mensajes



|   | CentroID | Nombre       | Ciudad | Direccion         | Telefono   |
|---|----------|--------------|--------|-------------------|------------|
| 1 | 4        | Clínica Este | Puyo   | Av. Principal 123 | 0987408137 |



|   | MedicoID | Nombre | Apellido | EspecialidadID | CentroID | Email                      | Telefono |
|---|----------|--------|----------|----------------|----------|----------------------------|----------|
| 1 | 5        | John   | Peñaloza | 1              | 4        | jpenaloza@clinicanoorte.ec | NULL     |


```

Fig. 7. Verificación de Transacción exitosa.

Transacción fallida: Eliminar una especialidad en uso (ID=3) (Fig 8).

BEGIN TRANSACTION;

DELETE FROM Especialidades WHERE EspecialidadID = 5;

ROLLBACK TRANSACTION;

```
52
53 --- TRANSACCION FALLIDA
54 BEGIN TRANSACTION;
55 DELETE FROM Especialidades WHERE EspecialidadID = 5;
56 ROLLBACK TRANSACTION;

57
58 SELECT * FROM Especialidades;

[+] 8 8 Mensajes

Mens. 547, Nivel 16, Estado 0, Línea 155
The DELETE statement conflicted with the REFERENCE constraint "FK_Asignacion_Especialidad".
```

Fig. 8. Transacción Fallida por Intergridad Referencial.

Se considera una transacción fallida porque se intenta ejecutar algo que viola la integridad referencial. Como estás usando ROLLBACK TRANSACTION; después, la transacción queda revertida totalmente.

5. Pruebas concurrentes

Sesión 1 (Actualización) (Fig 9)

BEGIN TRANSACTION;



UNIVERSIDAD TÉCNICA DE AMBATO
FACULTAD DE INGENIERÍA EN SISTEMAS ELECTRÓNICA E INDUSTRIAL
CARRERA DE TECNOLOGÍAS DE LA INFORMACIÓN
CICLO ACADÉMICO: AGOSTO 2025 – ENERO 2026



```
UPDATE Medicos SET Email = 'marta.nuevo@clinicanorte.ec' WHERE MedicoID = 1;  
-- Mantener la transacción abierta (sin COMMIT)
```

The screenshot shows a SQL query window with the following content:

```
163    -- PRUEBAS CONCURRENTES  
164    -- SESIÓN 1  
165    BEGIN TRANSACTION;  
166    UPDATE Medicos SET Email = 'marta.nuevo@clinicanorte.ec' WHERE MedicoID = 1;  
167    -- Mantener la transacción abierta (sin COMMIT)  
168  
169
```

Below the code, there is a status bar with "Línea: 169" and "Carácter: 1 SPC". At the bottom, it says "(1 fila afectada)".

Fig. 9. Actualización de correo al Médico con id 1.

Sesión 2 (Lectura) (Fig 10)

```
SELECT * FROM Medicos WHERE MedicoID = 1;
```

-- Bloqueado hasta que Sesión 1 finalice

The screenshot shows a SQL query window with the following content:

```
169    -- SESIÓN 2 (Lectura)  
170    SELECT * FROM Medicos WHERE MedicoID = 1;  
171    -- Bloqueado hasta que Sesión 1 finalice  
109 %
```

Below the code, there is a status bar with "109 %". At the bottom, it says "Resultados" and "Mensajes". A table is shown with the following data:

	MedicoID	Nombre	Apellido	CentroID	Email	Teléfono
1	1	Marta	Vega	1	marta.nuevo@clinicanorte.ec	0995111111

Fig. 10. Comprobación de actualización.

6. Manejo de errores con TRY-CATCH

Ejemplo (Fig 11):

```
BEGIN TRY
```

```
    BEGIN TRANSACTION;
```

```
    INSERT INTO Consultas (MedicoID, ClienteID, FechaConsulta)
```

```
    VALUES (1, 9999, GETDATE());
```

```
    COMMIT TRANSACTION;
```

```
END TRY
```

```
BEGIN CATCH
```

```
    ROLLBACK TRANSACTION;
```

```
    PRINT 'Error: ' + ERROR_MESSAGE();
```

```
END CATCH;
```

The screenshot shows a SQL query window with the following content:

```
177    --- MANEJO DE ERRORES CON TRY-CATCH  
178    BEGIN TRY  
179    BEGIN TRANSACTION;  
180    INSERT INTO Consultas (MedicoID, ClienteID, FechaConsulta)  
181    VALUES (1, 9999, GETDATE());  
182    COMMIT TRANSACTION;  
183    END TRY  
184    BEGIN CATCH  
185    ROLLBACK TRANSACTION;  
186    PRINT 'Error: ' + ERROR_MESSAGE();  
187    END CATCH;  
188
```

Below the code, there is a status bar with "188" and "Línea". At the bottom, it says "(0 filas afectadas)" and "Error: The INSERT statement conflicted with the FOREIGN KEY constraint "FK_Consultas_Clientes". Hora de finalización: 2025-10-03T11:31:51.2969127-05:00".

Fig. 11. Transacción con control de errores mediante Try-Catch.



7. Verificación final

Consultar datos nuevos (Fig 12):

-- Verificar el nuevo centro médico

```
SELECT * FROM CentrosMedicos WHERE Nombre = 'Clínica Este';
```

-- Verificar el médico asignado al nuevo centro

```
SELECT *
```

```
FROM Medicos
```

```
WHERE Medicoid = 5 AND Nombre = 'John' AND Apellido = 'Peñaloza';
```

```
190
191    --- VERIFICACION FINAL
192    -- Verificar el nuevo centro médico
193    SELECT * FROM CentrosMedicos WHERE Nombre = 'Clínica Este';
194    -- Verificar el médico asignado al nuevo centro
195    SELECT *
196    FROM Medicos
197    WHERE Medicoid = 5 AND Nombre = 'John' AND Apellido = 'Peñaloza';

09 %  7  0  Línea: 197
```

CentroID	Nombre	Ciudad	Direccion	Telefono
1	4	Puyo	Av. Principal 123	0987408137

MedicoID	Nombre	Apellido	CentroID	Email	Telefono
1	5	John	Peñaloza	jpenaloza@clicanorte.ec	NULL

Fig. 12. Comprobación de Transacciones.

2.7 Resultados obtenidos

- Se creó y validó la base de datos CentroMedicoBD en SQL Server, incluyendo tablas como CentrosMedicos, Medicos, Especialidades y Consultas.
- Se comprobó la integridad referencial mediante intentos de inserción y eliminación de registros con claves foráneas inválidas, lo que generó errores controlados.
- Se implementaron transacciones atómicas con COMMIT y ROLLBACK, asegurando que las operaciones se completen totalmente o se reviertan en caso de error.
- Se simuló bloqueo por concurrencia al mantener una transacción abierta en una sesión, impidiendo la lectura en otra sesión hasta su finalización.
- Se utilizó la estructura TRY-CATCH para manejar errores durante transacciones, mostrando mensajes de error sin interrumpir la ejecución.
- Todas las operaciones fueron validadas mediante consultas SELECT, confirmando la consistencia de los datos después de cada transacción exitosa.

2.8 Habilidades blandas

- Liderazgo
- Trabajo en equipo
- Comunicación assertiva
- La empatía
- Pensamiento crítico
- Flexibilidad
- La resolución de conflictos
- Adaptabilidad
- Responsabilidad

2.9 Conclusiones



- Las transacciones en SQL Server son esenciales para mantener la consistencia y atomicidad en operaciones críticas, especialmente en entornos distribuidos.
- La integridad referencial garantiza la calidad de los datos al evitar relaciones inconsistentes entre tablas.
- El manejo de errores con TRY-CATCH permite una gestión robusta y controlada de excepciones en procesos transaccionales.
- La concurrencia puede afectar el rendimiento y la disponibilidad de los datos si no se gestiona adecuadamente, por lo que es crucial diseñar transacciones eficientes.
- El uso de herramientas como SSMS facilita la implementación y verificación de transacciones y relaciones entre entidades.

2.10 Recomendaciones

- Utilizar transacciones explícitas siempre que se realicen operaciones que involucren múltiples pasos críticos.
- Implementar índices en claves foráneas y primarias para mejorar el rendimiento en consultas con joins y restricciones.

2.11 Referencia Bibliográfica

- [1] hdeleon.net, *La magia de las transacciones SQL | Ejemplo en Sql Server*, (el 21 de agosto de 2020). Consultado: el 3 de octubre de 2025. [En línea Video]. Disponible en: <https://www.youtube.com/watch?v=keL9-EtE-zE>
- [2] Microsoft Learn, “BEGIN TRANSACTION (Transact-SQL) - SQL Server”. Consultado: el 3 de octubre de 2025. [En línea]. Disponible en: <https://learn.microsoft.com/es-es/sql/t-sql/language-elements/begin-transaction-transact-sql?view=sql-server-ver17>
- [3] Microsoft Learn, “ROLLBACK TRANSACTION (Transact-SQL) - SQL Server”. Consultado: el 3 de octubre de 2025. [En línea]. Disponible en: <https://learn.microsoft.com/es-es/sql/t-sql/language-elements/rollback-transaction-transact-sql?view=sql-server-ver17>