**INFORME GUÍA APE Nº2**

1. **PORTADA**

Tema: APE 2. Tratamiento de 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

Asignatura: Sistemas de Base de Datos Distribuidos

Docente: Ing. José Caiza

1. **INFORME DE GUIA APE**
2. **PP**
3. **YY**
   1. **Objetivos**

**General:**

Determinar el comportamiento de un SGBD con transacciones.

**Específicos:**

* 1. **Instrucciones**
* Conéctese al motor de base de datos
* Cree una BD llamada Universidad
* Ingresemos algunos datos y verificamos la integridad referencial
* Creamos transacciones y probamos las características de atomicidad Commit y Rollback
* Habilitamos una nueva sesión para pruebas
* Manejamos errores On\_error, set xact\_abort y Try
  1. **Listado de materiales**

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

* Computadora
* Diapositivas
* SQL Server

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

* 1. **Desarrollo de la actividad**

**Creación de Base de Datos**

**Paso 1.-** Crear una Base de Datos llamada Universidad en la cual se crean las tablas con sus respectivas relaciones como se muestra en la Fig 1.

Diagrama

Descripción generada automáticamente

**Fig 1.**  Modelo Entidad Relación de BD Universidad.

**Paso 2.-** Insertar datos de prueba para poder realizar las diferentes transacciones a lo largo de la guía (Fig 2).

Interfaz de usuario gráfica, Texto, Aplicación

Descripción generada automáticamente

**Fig 2.**  Inserción de Datos de Ejemplo en las tablas de BD Universidad.

**Verificar Integridad Referencial**

**Verificación 1.-** Intentar insertar un registro con una clave foránea que no exista en este caso al querer asignar una edición de curso a un curso que no existe (Fig 3).

Interfaz de usuario gráfica, Texto, Aplicación

Descripción generada automáticamente

**Fig 3.**  Fallo por Integridad Referencial.

**Verificación 2.-** Intentar eliminar un curso que ya está siendo usado en course\_edition (Fig 4).

Interfaz de usuario gráfica, Texto, Aplicación, Word, Correo electrónico

Descripción generada automáticamente

**Fig 4.**  Fallo por Integridad Referencial.

**Verificación 3.-** Verificar que no se puede cambiar el valor de una clave primaria (por ejemplo, id de lecturer) si ya está usada como clave foránea en otra tabla (course\_edition) (Fig 5).

Interfaz de usuario gráfica, Texto, Aplicación, Word

Descripción generada automáticamente

**Fig 5.**  Fallo por Integridad Relacional.

**Crear transacciones y probar COMMIT/ROLLBACK**

Una transacción agrupa operaciones para que todas se ejecuten o ninguna (principio de atomicidad).

**Transacción 1.-** Insertar un registro correspondiente en la tabla course\_enrollment que refleje la matrícula del estudiante en la edición del curso con ID 1(Fig 6).

Interfaz de usuario gráfica, Texto, Aplicación

Descripción generada automáticamente

**Fig 6.** Inserción de Estudiante y Asignación a una Matrícula.

Se realiza una consulta específica con los datos del estudiante creado y su asignación a una matrícula para poder comprobar que la Transacción se haya realizado correctamente (Fig 7).

Interfaz de usuario gráfica, Texto, Tabla

Descripción generada automáticamente

**Fig 7.** Verificación de Estudiante y Asignación de Matrícula.

**Rollback 1**.- Insertar correctamente un nuevo estudiante en la tabla student e intentar insertar una matrícula en la tabla course\_enrollment con una edición de curso no exitente (Fig 8).

Interfaz de usuario gráfica, Texto, Aplicación

Descripción generada automáticamente

**Fig 8**. Inserción de Estudiante y Asignación de Matrícula fallida.

Se realiza una consulta específica con los datos del estudiante creado para poder comprobar que el Rollback se haya realizado correctamente, El registro se revirtió, garantizando que la atomicidad ayuda a que ningún cambio parcial quedó en la base. (Fig 9).

Interfaz de usuario gráfica, Texto, Aplicación

Descripción generada automáticamente

**Fig 9**. Verificación de Rollback exitoso al no aparecer ningún Estudiante con id 11.

**Transacción 2.-** Insertar correctamente un nuevo profesor y una nueva edición de curso asociada (Fig 10).

Interfaz de usuario gráfica, Texto, Aplicación

Descripción generada automáticamente

**Fig 10**. Inserción de Profesor y Asignación a una Edición del curso.

Se realiza una consulta específica con los datos del profesor creado y su asignación a una edición del curso para poder comprobar que la Transacción se haya realizado correctamente (Fig 11).

Interfaz de usuario gráfica, Texto, Aplicación, Correo electrónico

Descripción generada automáticamente

**Fig 11**. Verificación de Profesor creado y Asignación a una Edición del cuso.

**Rollback 2**.- Intentar insertar una edición de curso con un lecturer\_id (profesor) inexistente (Fig 12).

Interfaz de usuario gráfica, Texto, Aplicación, Correo electrónico

Descripción generada automáticamente

**Fig 12**. Insertar un Curso y Asignar un Profesor inexistente.

Se realiza una consulta específica con los datos del curso creado y de la asignación de una edición de curso para poder comprobar que el Rollback se haya realizado correctamente, Ambos registros se revirtieron, garantizando que la atomicidad ayuda a que ningún cambio parcial quedó en la base. (Fig 13).

Interfaz de usuario gráfica, Texto, Aplicación, Correo electrónico

Descripción generada automáticamente

**Fig 13**. Verificación de Rollback exitoso al no aparecer ninguno de los registros.

**Habilitar una nueva sección para pruebas**

**Paso 1.-** Abrir dos ventanas de consulta (sesión 1 y sesión 2), la Ventana 1, será Sesión A(Fig 14) y la Ventana 2, será Sesión B (Fig 15). Verificar el id de cada una.

**SELECT @@SPID AS 'ID de Sesión Actual';**

Interfaz de usuario gráfica, Texto, Aplicación, Correo electrónico

Descripción generada automáticamente

**Fig 14**. Sesión A y Verificación de su id.

Interfaz de usuario gráfica, Texto, Aplicación

Descripción generada automáticamente

**Fig 15**. Sesión B y Verificación de su id.

**Paso 2.-** En la Sesión A (Transacción abierta), ejecutar la siguiente transacción sin hacer COMMIT todavía (Fig 16).

Interfaz de usuario gráfica, Texto, Aplicación, Correo electrónico

Descripción generada automáticamente

**Fig 16**. Transacción efectuada sin Commit.

**Paso 3.-** En la Sesión B (otra ventana de consulta), probar si se puede ver el registro (Fig 17).

**SELECT \* FROM student WHERE id = 20;**

Interfaz de usuario gráfica, Texto, Aplicación, Correo electrónico

Descripción generada automáticamente

**Fig 17**. Verificación en Sesión B de la Transacción realizada.

No se verá el registro todavía, porque la transacción de la sesión A no ha hecho COMMIT. Esto demuestra el aislamiento de transacciones (cada sesión ve solo los datos confirmados).

**Paso 4.-** Ahora vuelvor a la Sesión A y ejecutar el siguiente comando.

**COMMIT TRANSACTION;**

**Paso 5.-** En la Sesión B (otra ventana de consulta), probar si se puede ver el registro (Fig 18), Ahora sí aparecerá el registro, porque ya se confirmó en la otra sesión..

Interfaz de usuario gráfica, Texto, Aplicación, Correo electrónico

Descripción generada automáticamente

**Fig 18**. Verificación de Transacción reflejada en Sesión B.

**Manejo de errores ON\_ERROR, SET XACT\_ABORT, TRY...CATCH**

**Opción 1.- SET XACT\_ABORT ON:** Si ocurre un error, SQL Server haga automáticamente ROLLBACK.

Interfaz de usuario gráfica, Texto, Aplicación, Correo electrónico

Descripción generada automáticamente

**Fig 19**. Transacción abortada debido a que no existe esa Edición de Curso.

Se realiza una consulta específica con los datos del estudiante creado y de la asignación de una Matrícula para poder comprobar que el XACT\_ABORT se haya realizado correctamente (Fig 20).

Interfaz de usuario gráfica, Texto, Aplicación, Correo electrónico

Descripción generada automáticamente

**Fig 20**. Verificación de auto Rollback por XACT\_ABORT.

**Opción 2.-** **TRY...CATCH (manejo controlado):** Permite capturar el error y hacer rollback manual (Fig 21).

Interfaz de usuario gráfica, Texto, Aplicación, Correo electrónico

Descripción generada automáticamente

**Fig 21**. Transacción detenida debido al error detectado en Edición de Curso inexistente.

Se realiza una consulta específica con los datos del estudiante creado y de la asignación de una Matrícula para poder comprobar que el TRY...CATCH se haya realizado correctamente (Fig 22).

Interfaz de usuario gráfica, Aplicación

Descripción generada automáticamente

**Fig 22**. Verificación de Rollback por Try…Catch.

**Opción 3.-** **ON\_ERROR:** SQL Server no tiene una instrucción directa llamada ON\_ERROR como Oracle. Pero podemos simular su comportamiento usando BEGIN TRAN + ROLLBACK en triggers o transacciones, y manejando el error con ERROR\_NUMBER() / ERROR\_MESSAGE().

* 1. **Resultados obtenidos**
  2. **Habilidades blandas**

Liderazgo

Trabajo en equipo

Comunicación asertiva

La empatía

Pensamiento crítico

Flexibilidad

La resolución de conflictos

Adaptabilidad

Responsabilidad

* 1. **Conclusiones**
  2. **Referencia Bibliográfica** [1][2][3]