



UNIVERSIDAD TECNOLÓGICA DE PANAMÁ
FACULTAD DE INGENIERÍA DE SISTEMAS COMPUTACIONALES
LABORATORIO # 9



Facilitador(a): _____ Asignatura: _____

Estudiante: _____

Fecha: _____ Grupo: _____

A. TÍTULO DE LA EXPERIENCIA: SQL – CONTROL DE CONCURRENCIA

B. TEMAS:

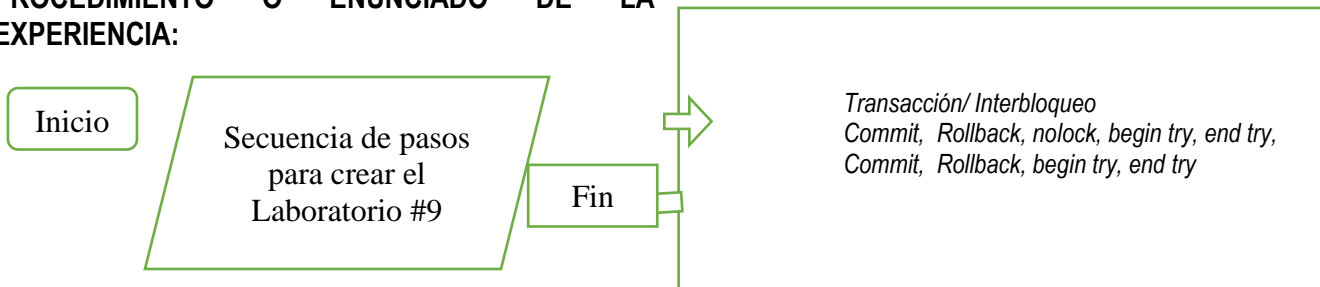
Transacción/ Interbloqueo

Comandos y Sentencias: Begin Transaction, Commit, Rollback, nolock, begin try, end try, begin catch, end catch

OBJETIVO(S): Repaso del lenguaje de manipulación de datos en SQL Server, usando con ejemplos sus principales sentencias

C. METODOLOGÍA: Laboratorio práctico en clases

D. PROCEDIMIENTO O ENUNCIADO DE LA EXPERIENCIA:



E. RECURSOS:

- Puntualidad en la asistencia a la hora correspondiente al laboratorio
- Equipo computacional completo (CPU, Teclado, Mouse, Monitor o Pantalla), software (Microsoft SQL Server Managment Studio), Tablero, Marcadores y borrador
- Libretas y bolígrafo o lápiz, para anotaciones.

F. INTRODUCCION:

Los laboratorios son proyectos simplificados, que pueden generarse rápidamente y poner en práctica los temas del contenido. No pretenden tener la complejidad de proyectos reales. Los laboratorios tienen estructuras de tareas, nombres y cantidades de recursos ficticios, que no reflejan los valores de una BD real. Un proyecto real generalmente requiere de mayores niveles de detalle y complejidad en el plan de trabajo que los que se utilizan en los laboratorios. Las instrucciones que encontrará en los laboratorios no son 100% detalladas ni compatibles con la versión que tengan en sus equipos, tomar en cuenta, pues puede haber ciertas variaciones.

G. Instrucciones:

Al igual que el laboratorio pasado, ir guardando todo en un solo query llamado LABBD9

Repaso de términos:

Las transacciones se encargan de mantener la integridad de los datos almacenados dentro de nuestras bases de datos.

Una transacción es un grupo de comandos que cambian la data almacenada en una base de datos, y se asegura que todos los comandos se completen y finalicen con éxito, esto se conoce como **COMMIT** y en caso de existir alguna falla que no se ejecute ninguno de ellos.

Si alguno de los comandos falla, todos los demás comandos fallarán y por ende ningún dato será modificado en la base de datos, esto se conoce como **ROLLBACK**.

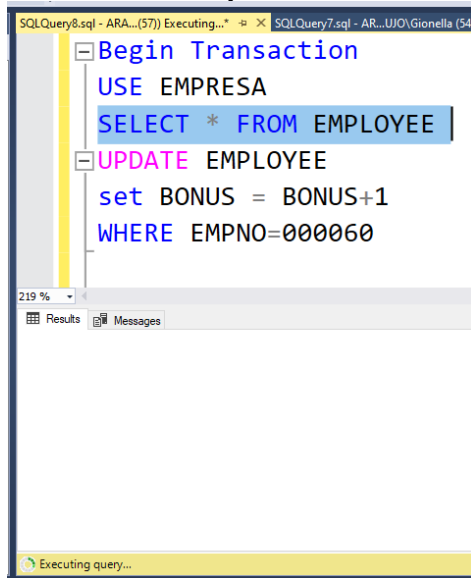
Primera parte

PASOS:

I. Realizar un interbloqueo sencillo como lo explicado en clases y desbloquear con el nolock

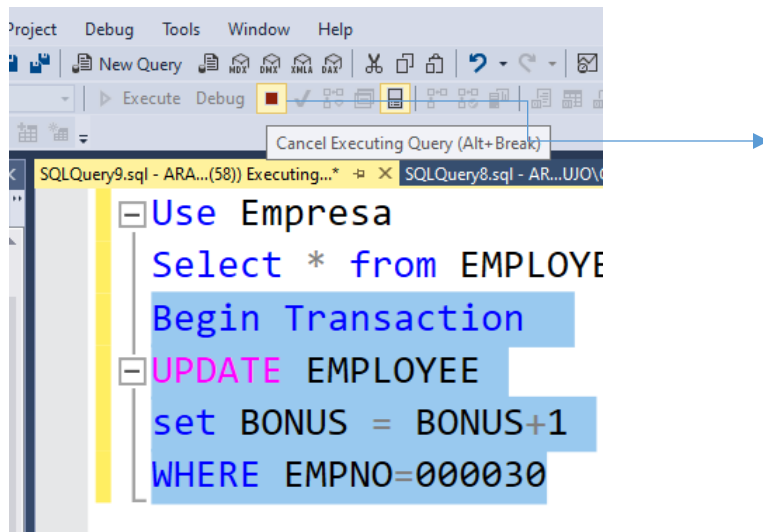
Pasos:

1. Usando la misma base de datos anterior creada de Empresa (use Empresa), hacer un select de la tabla Employee para visualizarla
2. Iniciar una transacción (begin transaction), actualizar el campo bonus sumando 1 y usando una condición de where con un número de empleado que ud. escoja, ejecutar y confirmar que se ejecute el cambio
3. Abrir otra consulta y realizar la misma transacción para validar que el interbloqueo se de



Aquí se ve el interbloqueo, no ejecuta.

4. Hacer un select nuevamente para visualizar la tabla y al no poder continuar, primero cancelar



y luego probar la visualización con el (nolock)

5. Observar el comportamiento y si es posible ahora revisar la consulta, luego, por último
6. Regresar al query inicial y deshacer el cambio con ROLLBACK y observar el comportamiento al visualizar nuevamente la tabla, si es necesarios cierre todos los query y abra uno nuevo haciendo el select de la tabla y verá que no se hizo el cambio porque la transacción había quedado abierta, ahora que le ejecutó el rollback se revirtió.

Segunda parte (ejemplo de transacción entre cuentas):

Problema:

De las cuentas que hay en el banco, se quiere pasar un dinero de una cuenta a otra. Laurence y Marcos son amigos desde hace muchos años y tienen cuenta en el mismo banco, Marcos le pide a Laurence que le preste \$1300 por medio de una transacción.

PASOS:

1. Ejecutar el query proporcionado para correr: base de datos, tabla y datos
 - 1.1 Revisar y comprender el código entregado antes de ejecutarlo, usted debe poder hacerlo solo, pero para agilizar el tiempo y nos rinda en el lab. Se le ha proporcionado.
2. Luego visualizar la tabla de CuentasTrans
3. El banco debe asegurarse de restarle a Laurence los \$1300 de su cuenta y abonarlos a la cuenta de Marcos, en caso contrario, deshacer la transacción y devolverle los \$1300 a Laurence, esto lo hace por medio usando el *begin try, end try, begin catch, end catch*.

Usar la siguiente guía:

```
BEGIN TRY
  BEGIN TRANSACTION
```

***** hacer las sentencias necesarias para dicho cambio *****

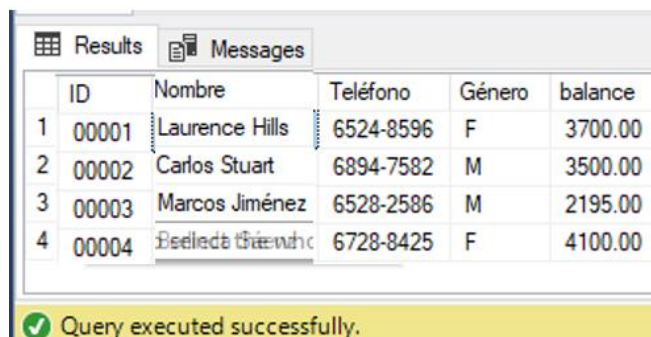
```
    COMMIT TRANSACTION
    PRINT 'Transacción completada'
  END TRY
  BEGIN CATCH
    ROLLBACK TRANSACTION
```

```
PRINT 'Transacción cancelada'
END CATCH
```

Al usar **BEGIN TRY** en nuestro código lo que estamos haciendo es la prueba de errores, es decir, indicamos que ejecute todo el script que hay dentro de **BEGIN TRY** hasta **END TRY**, si no ha surgido ningún problema en ninguna de las sentencias, entonces que ejecute el **COMMIT** e imprime en pantalla el mensaje.

Si hubiera existido algún error o falla en la transacción entonces se lanzaría el **CATCH** y por ende se dispararía el **ROLLBACK** y su respectivo mensaje.

4. EJECUTAR el código completo, visualizar la tabla para ver los cambios y debe salir así:



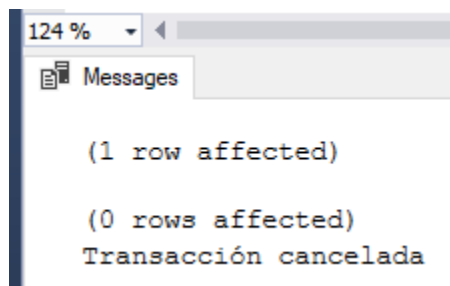
	ID	Nombre	Teléfono	Género	balance
1	00001	Laurence Hills	6524-8596	F	3700.00
2	00002	Carlos Stuart	6894-7582	M	3500.00
3	00003	Marcos Jiménez	6528-2586	M	2195.00
4	00004	Esleida Hernández	6728-8425	F	4100.00

Query executed successfully.

5. Ahora veremos parte del uso de la sentencia anterior junto con el Rollback
Que pasaría si Marcos le pide al banco que devuelva los \$1300 a Laurence y que adicional le de unos \$150.00 más por el favor que le hizo del préstamo, es decir que le de en total unos \$1450.00, pero el banco, sin embargo, por error de dedo en el número de cuenta (id) de Laurence coloca es su nombre.

```
WHERE ID = 'Laurence'
```

- 5.1 Hagan los cambios y ejecuten el código completo, debe darles este mensaje en la ejecución



124 %

Messages

(1 row affected)

(0 rows affected)

Transacción cancelada

Como se puede visualizar, la primera transacción se efectúa, sin embargo, la segunda falla haciendo que se dispare el **ROLLBACK** y su respectivo mensaje. Esto origina que la primera transacción sea deshecha y los datos queden sin modificación alguna y con esto se valida la propiedad de atomicidad, en donde una transacción es todo o nada.

*****SUERTE*****