**1) No contexto de Banco de Dados Relacional, o que é uma Transação e por que ela deve ser utilizada?**

Uma **transação** no contexto de banco de dados relacional é um conjunto de operações (instruções SQL) que são tratadas como uma única unidade de trabalho. Essas operações podem incluir comandos de inserção, atualização, deleção, entre outros. Uma transação garante que todas essas operações sejam concluídas com sucesso ou, em caso de falha, que todas sejam desfeitas, garantindo assim a consistência dos dados.

Ela deve ser utilizada por dois motivos principais:

* **Integridade dos Dados:** Assegura que todas as operações dentro de uma transação sejam concluídas corretamente, sem quebras de integridade.
* **Isolamento:** Assegura que transações concorrentes não interfiram entre si, evitando problemas como leituras sujas (dirty reads) ou atualizações perdidas (lost updates).

**2) Propriedades de uma Transação (ACID)**

As transações possuem quatro propriedades que garantem sua confiabilidade, conhecidas pela sigla **ACID**:

* **Atomicidade (A)**: Assegura que todas as operações da transação sejam tratadas como uma única unidade. Ou seja, ou todas as operações são bem-sucedidas, ou nenhuma delas é aplicada.
* **Consistência (C)**: Garante que a transação leva o banco de dados de um estado consistente para outro estado consistente, mantendo as regras de integridade do banco.
* **Isolamento (I)**: Garante que as operações de uma transação não afetem outras transações concorrentes, evitando conflitos.
* **Durabilidade (D)**: Uma vez que uma transação é concluída com sucesso (commit), as mudanças realizadas no banco de dados são permanentes, mesmo que ocorra uma falha de sistema.

**3) Comandos COMMIT TRANSACTION e ROLLBACK**

* **COMMIT TRANSACTION**: Confirma todas as operações realizadas dentro da transação, tornando permanentes as alterações no banco de dados. Após um COMMIT, os dados não podem ser revertidos, e outras transações podem acessar essas mudanças.
* **ROLLBACK TRANSACTION**: Desfaz todas as operações realizadas durante a transação, retornando o banco de dados ao estado anterior ao início da transação. Esse comando é utilizado em caso de falha ou erro dentro da transação.

**4) Script SQL para criar uma Transação**

Aqui está um exemplo de script SQL para ilustrar uma transação:

BEGIN TRANSACTION;

-- Inserção de um novo cliente na tabela Clientes

INSERT INTO Clientes (Nome, Email)

VALUES ('João Silva', 'joao.silva@email.com');

-- Inserção de uma nova venda associada ao cliente inserido

INSERT INTO Vendas (ClienteID, Valor, DataVenda)

VALUES (SCOPE\_IDENTITY(), 500.00, GETDATE());

-- Se tudo ocorreu bem, confirmar a transação

COMMIT TRANSACTION;

**Explicação do código:**

* **O que o código faz?**: O script começa uma transação e tenta inserir um novo cliente na tabela Clientes. Em seguida, insere uma nova venda na tabela Vendas associada ao cliente recém-inserido. Se ambas as operações forem bem-sucedidas, a transação é confirmada com o COMMIT.

**Resultados esperados:**

* **Execução bem-sucedida**: Se o script for executado com sucesso, o novo cliente e a nova venda serão inseridos no banco de dados, e essas alterações serão permanentes após o COMMIT.
* **Execução com falha**: Se qualquer uma das operações falhar (por exemplo, um erro ao inserir a venda), você pode adicionar um ROLLBACK manualmente para garantir que nenhuma mudança seja aplicada ao banco de dados. Aqui está o exemplo:

BEGIN TRANSACTION;

BEGIN TRY

-- Inserção de um novo cliente na tabela Clientes

INSERT INTO Clientes (Nome, Email)

VALUES ('João Silva', 'joao.silva@email.com');

-- Inserção de uma nova venda associada ao cliente inserido

INSERT INTO Vendas (ClienteID, Valor, DataVenda)

VALUES (SCOPE\_IDENTITY(), 500.00, GETDATE());

-- Se tudo ocorrer bem, confirmar a transação

COMMIT TRANSACTION;

END TRY

BEGIN CATCH

-- Em caso de erro, desfazer a transação

ROLLBACK TRANSACTION;

PRINT 'Erro: A transação foi revertida.';

END CATCH;

**Explicação adicional:**

* Se a transação falhar, o ROLLBACK irá reverter as operações, garantindo que nenhuma inserção será feita no banco.