# Parte 1 – Questões Teóricas

1. **Explique com suas palavras o que é uma transação em um banco de dados e por que ela é importante em sistemas que envolvem múltiplas operações de escrita.**

**Uma transação em um banco de dados é uma sequência de uma ou mais operações, ela é importante para poder se fazer múltiplas operações e só depois aplica-las no banco de dados.**

1. **Enumere e explique as quatro propriedades ACID das transações. Dê um exemplo prático de sistema real onde cada propriedade é crucial.**
2. Atomicidade
3. Consistência
4. Isolamento
5. Durabilidade
   * + 1. – Ao transferir saldo a uma conta, o dinheiro é retirado da conta A e inserido na conta B, a transação só se dará como concluída quando tudo finalizar
       2. – Ao transferir um valor de 20 reais de uma conta que só tem 10, a propriedade consistência não permite que a transferência seja realizada.
       3. - Ao transferir um valor de uma conta A para uma conta B, a conta A é bloqueada para que outras transações bloqueiem ou modifiquem a conta até que a primeira transação seja concluída
       4. – Quando vc transfere um valor da conta A para a conta B, a durabilidade garante que as mudanças sejam aplicadas e mantidas no banco de dados.

1. **Qual a diferença entre os comandos COMMIT e ROLLBACK? Em que situações cada um deve ser usado?**

Commit é pra finalizar e colocar as alterações no banco de dados e RollBack é para retroceder as ultimas alterações no banco de dados.

Commit deve ser utilizado pra finalizar e aplicar as mudanças no banco de dados. O Rollback é usado pra reverter as ultimas mudanças feitas no banco de dados.

1. **Uma transação T1 atualiza o estoque de um produto. Antes de dar COMMIT, outra transação T2 tenta consultar o estoque. O que acontece em um nível de isolamento READ COMMITTED? E em READ UNCOMMITTED? Justifique.**

No isolamento READ Committed , a transação T2 não terá acesso aos dados porque o commit não foi dado.Já o READ UNCOMMITTED é o exato oposto, a transação T2 conseguirá ver os arquivos mesmo sem o commit.

# Parte 2 – Questões Práticas (com SQL)

⚙ **Estrutura do Banco de Dados (Execute antes das atividades):**

CREATE TABLE contas ( id SERIAL PRIMARY KEY, titular VARCHAR(100), saldo NUMERIC(10, 2)

);

INSERT INTO contas (titular, saldo) VALUES

('Ana', 1000.00),

('Bruno', 500.00),

('Carlos', 1500.00);

**Exemplo de Commit e Rollback (Somente um exemplo não tem haver com os exercícios da atividade)**

-- Criação

CREATE TABLE contas ( id SERIAL PRIMARY KEY, nome TEXT, saldo NUMERIC

);

INSERT INTO contas (nome, saldo) VALUES ('Maria', 1000), ('João', 500);

-- Transação com erro

BEGIN;

UPDATE contas SET saldo = saldo - 300 WHERE nome = 'Maria';

UPDATE contas SET saldo = saldo + 300 WHERE nome = 'Carlos'; -- erro

ROLLBACK;

-- Verificar saldos

SELECT \* FROM contas;

1. **Realize uma transferência de R$ 300,00 da conta de Ana para Bruno dentro de uma transação.**
2. **Simule uma tentativa de transferência de R$ 200,00 de Bruno para Carlos, mas force um erro antes do COMMIT. Use ROLLBACK.**
3. Após o ROLLBACK da questão anterior, execute uma consulta SQL para mostrar os saldos atuais.

1. **Realize três operações dentro de uma única transação:** 
   * Depositar R$ 100,00 na conta de Carlos

* + Sacar R$ 50,00 da conta de Ana

* + Inserir um erro proposital (como tentar inserir em uma tabela inexistente)

* + Use ROLLBACK ao detectar o erro.

1. **Refaça a transação anterior, mas agora utilize um SAVEPOINT antes da operação de erro. Após o erro, faça ROLLBACK TO SAVEPOINT e finalize com COMMIT.**
2. **Crie uma nova transação em que Carlos transfere R$ 250,00 para Ana, e depois consulte os saldos das contas.**