Transações e Controle de Concorrência em Sistemas de Banco de Dados

Garantindo integridade e eficiência nas operações simultâneas

Agenda da Apresentação

- Fundamentos e Conceitos de Transação
- ACID: Princípios Essenciais das Transações
- Níveis de Isolamento de Transações
- Controle de Concorrência e Locks

Fundamentos e Conceitos de Transação

Definição de transação e sua importância

Conceito de Transação

Uma transação é uma sequência lógica de operações que forma uma unidade de trabalho no banco de dados.

Consistência do Banco de Dados

Transações garantem que o banco de dados permaneça consistente após todas as operações serem concluídas.

Proteção Contra Falhas e Concorrência

Transações protegem dados contra falhas e acessos concorrentes, assegurando integridade e segurança.

Comandos START TRANSACTION, COMMIT e ROLLBACK com exemplos práticos

Início da Transação

O comando START TRANSACTION inicia uma nova transação para agrupar operações no banco de dados.

Confirmação das Alterações

O comando COMMIT confirma todas as alterações feitas durante a transação, salvando as modificações.

Reversão das Operações

O comando ROLLBACK desfaz todas as operações desde o início da transação, revertendo modificações.

Fluxo básico de uma transação em bancos de dados

Iniciação da Transação

O processo começa com a iniciação da transação para garantir controle e integridade dos dados.

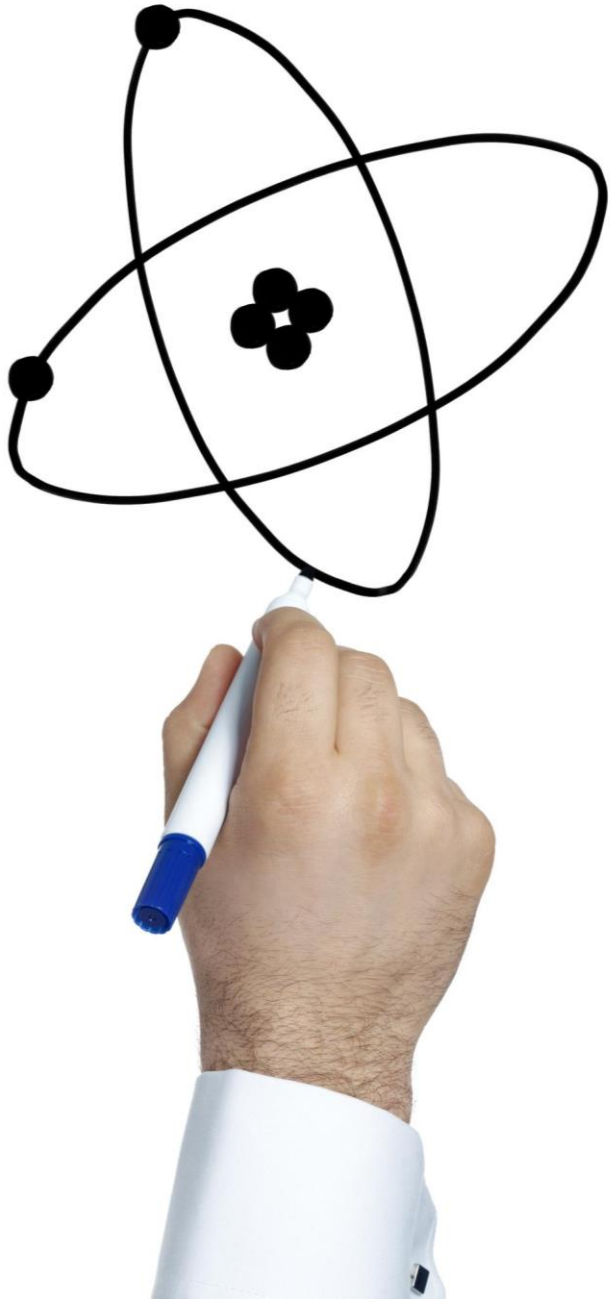
Operações de Leitura e Escrita

Durante a transação, são realizadas operações de leitura e escrita para manipular os dados.

Confirmação ou Desfazer Alterações

A transação termina com a decisão de confirmar (commit) ou desfazer (rollback) as alterações feitas.

ACID: Princípios Essenciais das Transações



Atomicidade: garantia de tudo ou nada com exemplos

Definição de Atomicidade

Atomicidade garante que operações em uma transação sejam concluídas integralmente ou não executadas.

Prevenção de Estados Inconsistentes

Evita estados intermediários que possam causar inconsistências no banco de dados durante operações.

Exemplo Prático

Transações financeiras, onde débito e crédito devem ocorrer juntos para manter integridade dos dados.

Consistência: manutenção da integridade dos dados

Definição de Consistência

Consistência assegura que o banco de dados transita apenas entre estados válidos, respeitando regras internas.

Regras e Restrições do Sistema

Todas as regras e restrições definidas no sistema devem ser respeitadas durante as transações para manter a integridade.

Isolamento e durabilidade: proteção e permanência dos dados após transações

Isolamento de Transações

O isolamento evita interferências entre transações concorrentes, garantindo integridade e consistência dos dados.

Durabilidade de Dados

A durabilidade assegura que as alterações confirmadas persistam mesmo após falhas ou reinicializações do sistema.

Níveis de Isolamento de Transações

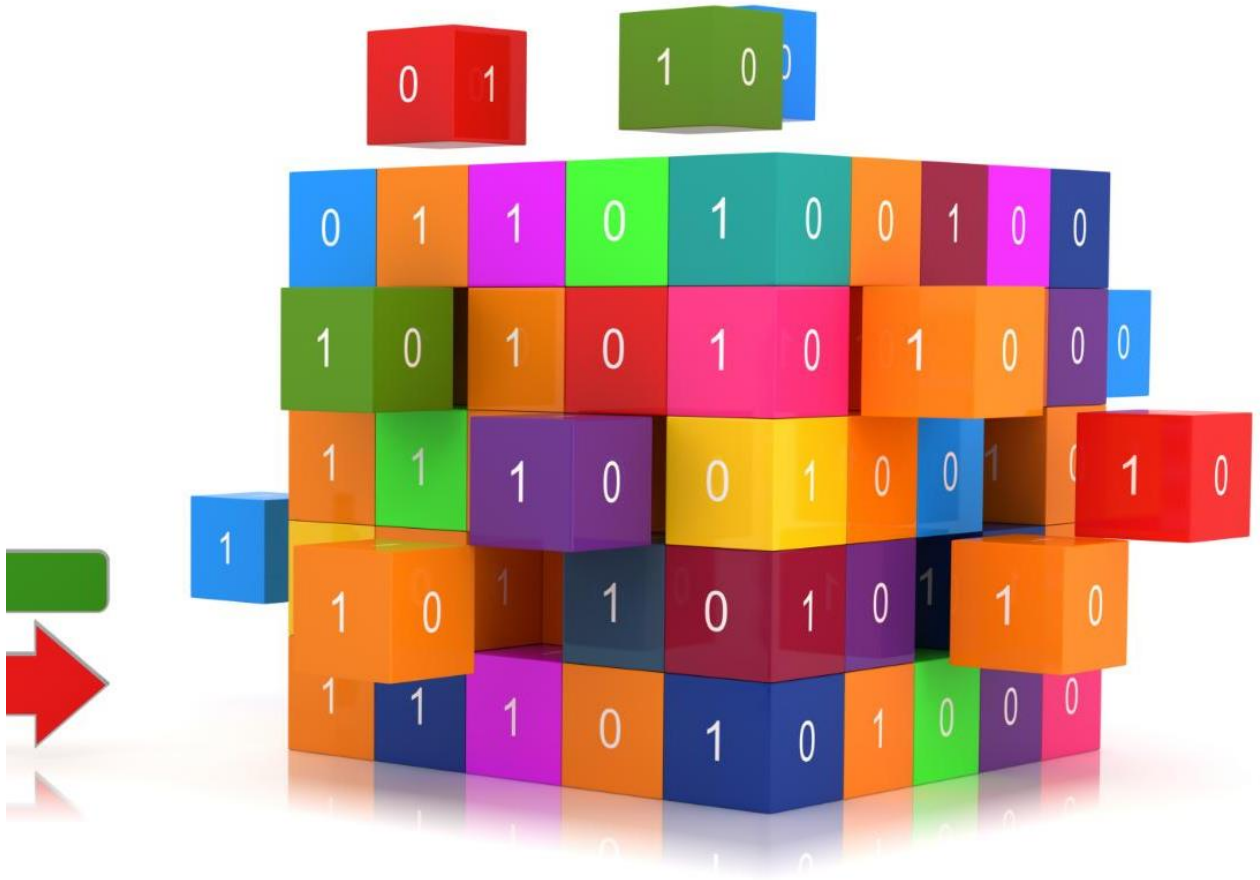
READ UNCOMMITTED e READ COMMITTED: diferenças e exemplos

READ UNCOMMITTED

Permite leitura de dados não confirmados, o que pode resultar em leituras sujas e inconsistentes.

READ COMMITTED

Evita leituras sujas garantindo que apenas dados confirmados sejam acessados nas consultas.



REPEATABLE READ: características e casos de uso

Definição do REPEATABLE READ

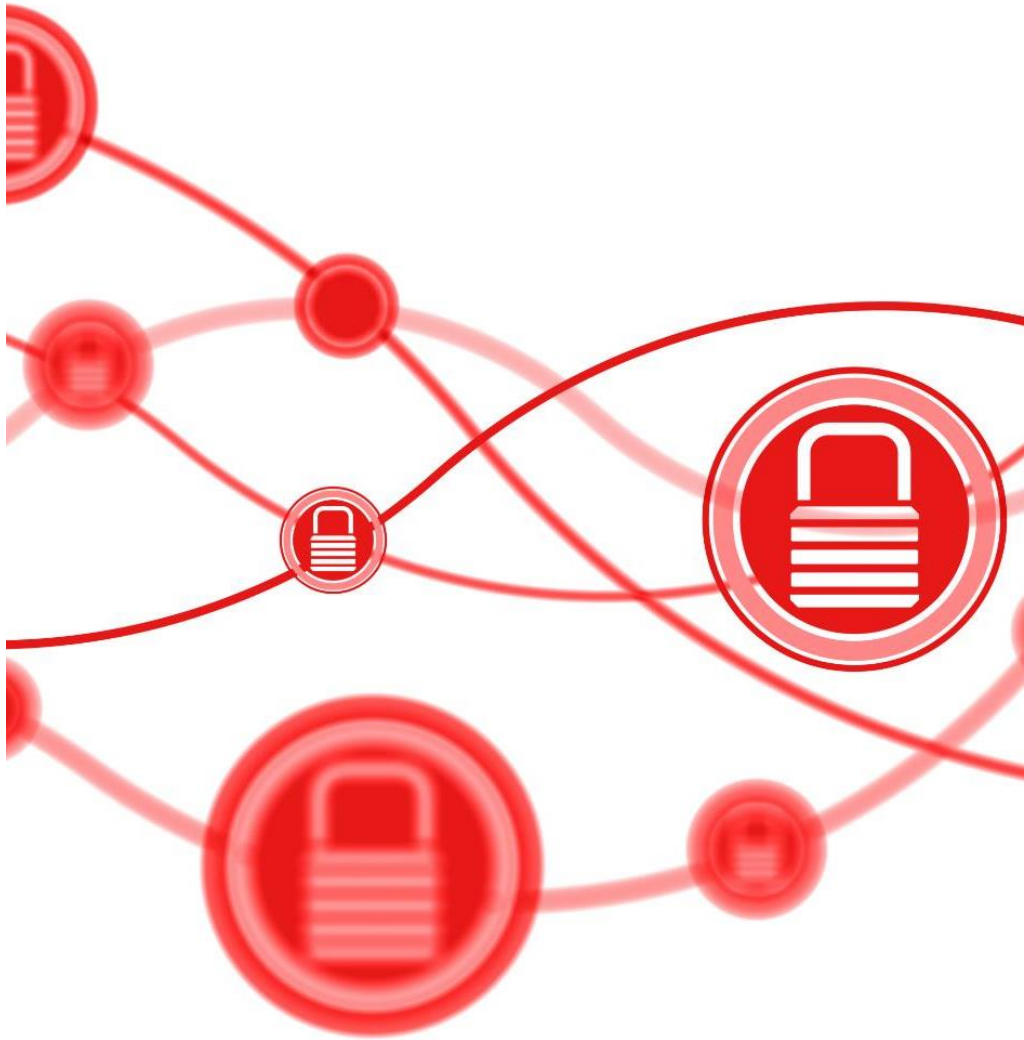
Esta isolação impede que dados lidos sejam alterados por outras transações durante sua execução.

Prevenção de leituras não repetíveis

Garante que dados lidos permaneçam consistentes para a mesma transação evitando leituras inconsistentes.

Ambientes concorrentes

Ideal para sistemas com múltiplas transações simultâneas que requerem alta consistência dos dados.



SERIALIZABLE: máxima proteção contra concorrência com exemplos práticos

Isolamento Total das Transações

O nível SERIALIZABLE assegura que as transações ocorram isoladamente, evitando conflitos de concorrência.

Prevenção de Problemas de Concorrência

Este nível elimina condições de corrida, leituras sujas e leituras fantasmas em sistemas de banco de dados.

Impacto na Performance

Garantir isolamento total pode afetar o desempenho do sistema devido ao aumento da espera e bloqueios.

Controle de Concorrência e Locks

Locks pessimistas: funcionamento e aplicação com exemplos

Definição de Locks Pessimistas

Locks pessimistas bloqueiam recursos antes do uso para evitar conflitos de acesso concorrente.

Aplicação em Ambientes de Alta Contenção

São usados em sistemas com alta contenção para garantir acesso exclusivo aos dados por transação.

Garantia de Acesso Exclusivo

Permitem que apenas uma transação acesse os dados simultaneamente, prevenindo conflitos.

Locks otimistas: como funcionam e quando utilizar

Princípio do Lock Otimista

Locks otimistas presumem que conflitos são raros e não bloqueiam dados durante a operação, verificando a integridade apenas na confirmação.

Ambientes Indicados

São mais eficientes em ambientes com baixa contenção e alta concorrência, onde conflitos simultâneos são incomuns.

Comparação entre abordagens pessimista e otimista com exemplos ilustrativos

Performance das Abordagens

Abordagem pessimista pode reduzir performance devido a bloqueios. Otimista oferece melhor performance em cenários com poucas colisões.

Riscos Associados

Pessimista evita conflitos bloqueando recursos, mas pode causar deadlocks. Otimista pode falhar em detectar conflitos rapidamente.

Uso Ideal e Exemplos

Pessimista é indicado para sistemas com alta contenção. Otimista funciona melhor em ambientes com baixa concorrência e alta leitura.

Conclusão

Importância das Transações

Compreender transações assegura integridade e consistência nos bancos de dados modernos.

Princípios ACID

Princípios ACID garantem confiabilidade nas operações e consistência dos dados.

Níveis de Isolamento e Concorrência

Níveis de isolamento e controle de concorrência promovem eficiência e evitam conflitos nas transações.