

Curso de Ciência da Computação

Banco de Dados 2 Aula 05

Prof. Dr. rer. nat. Eros Comunello



1

Controle de concorrência em BD

Propósitos do controle de concorrência

Isolamento da transação (Serialização)

Bloqueios

2

Propósitos do controle de concorrências

- ❑ PostgreSQL mantém a consistência dos dados utilizando o modelo multiversão (Multiversion Concurrency Control, MVCC);
- ❑ Isto significa que ao consultar o BD, cada transação enxerga um instantâneo (snapshot) dos dados; conforme estes dados eram há algum tempo atrás, sem levar em consideração o estado corrente dos dados subjacentes.
- ❑ Este modelo impede que a transação enxergue dados inconsistentes, fornecendo um isolamento da transação para cada uma das sessões do banco de dados.



Ciência da Computação

3

Propósitos do controle de concorrências

- ❑ A diferença principal entre os modelos multiversão e de bloqueio é que, no MVCC, os bloqueios obtidos para consultar (ler) os dados não conflitam com os bloqueios obtidos para escrever os dados e, portanto, a **leitura nunca bloqueia a escrita**, e a **escrita nunca bloqueia a leitura**.
- ❑ As funcionalidades de bloqueio, no nível de tabela e de linha, também estão disponíveis no PostgreSQL para aplicações que não podem se adaptar facilmente ao comportamento MVCC.
- ❑ Entretanto, a utilização apropriada do MVCC geralmente produz um desempenho melhor que os bloqueios.



Ciência da Computação

4

ISOLAMENTO DA TRANSAÇÃO



Ciência da Computação

5

Isolamento da transação

- ❑ O padrão SQL define quatro níveis de isolamento de transação em termos de **três fenômenos** que devem ser evitados entre transações concorrentes. Os fenômenos **não desejáveis** são:
 - dirty read** (leitura suja)
 - ❑ A transação lê dados não efetivados (*uncommitted*) escritos por uma transação concorrente.
 - nonrepeatable read** (leitura que não pode ser repetida)
 - ❑ A transação lê uma segunda vez os dados, e descobre que os dados foram modificados por outra transação (que os efetuou após ter sido feita a leitura anterior).



Ciência da Computação

6

Isolamento da transação

phantom read (leitura fantasma)

- ❑ A transação executa uma segunda vez uma consulta que retorna um conjunto de linhas que satisfaz uma determinada condição de procura, e descobre que o conjunto de linhas que satisfaz a condição é diferente devido a uma outra transação efetivada recentemente.



Ciência da Computação

7

Níveis de isolamento da transação no SQL

Nível de isolamento	Dirty Read	Nonrepeatable Read	Phantom Read
Read uncommitted	Possível	Possível	Possível
Read committed	Impossível	Possível	Possível
Repeatable read	Impossível	Impossível	Possível
Serializable	Impossível	Impossível	Impossível

- ❑ O PostgreSQL disponibiliza os níveis de isolamento *read committed* e *serializable* (serializável).



Ciência da Computação

8

Nível de isolamento Read Committed

- ❑ Uma transação sob este nível de isolamento, o comando **SELECT** enxerga apenas os dados efetivados antes da consulta começar;
- ❑ Nunca enxerga dados não efetivados, ou as mudanças efetivadas durante a execução da consulta pelas transações concorrentes;
 - ❑ o **SELECT** enxerga os efeitos das atualizações anteriores executadas dentro de sua própria transação, mesmo que ainda não tenham sido efetivadas.
- ❑ Na verdade, o comando **SELECT** enxerga um *snapshot* do BD, conforme este era no instante que a consulta começou a executar.



Ciência da Computação

9

Nível de isolamento Read Committed

- ❑ Observe que dois comandos **SELECTs** sucessivos enxergam dados diferentes;
 - ❑ Mesmo estando dentro de uma mesma transação, se outras transações efetuarem alterações durante a execução do primeiro comando **SELECT**.
- ❑ Os comandos **UPDATE**, **DELETE** e **SELECT FOR UPDATE** se comportam do mesmo modo que o **SELECT** para encontrar as linhas de destino: somente encontram linhas de destino efetivadas até a hora do início do comando.
- ❑ Entretanto, alguma linha de destino pode ter sido atualizada (ou excluída ou marcada para atualização) por outra transação concorrente, na hora que for encontrada.



Ciência da Computação

10

Nível de isolamento Read Committed

- ❑ Neste caso, o atualizador (*would-be updater*) aguarda a transação de atualização que começou primeiro efetivar ou desfazer (se ainda estiver executando).
- ❑ Se o primeiro atualizador desfizer (*rolls back*), então seus efeitos são negados e o segundo atualizador pode prosseguir com a atualização da linha original encontrada.
- ❑ Se o primeiro atualizador efetivar, o segundo atualizador ignora a linha se esta foi excluída pelo primeiro atualizador, senão tenta aplicar esta operação na versão atualizada da linha.
- ❑ A condição de procura do comando (cláusula WHERE) é avaliada novamente para verificar se a versão atualizada da linha ainda corresponde à condição de procura.
- ❑ Se corresponder, o segundo atualizador prossegue sua operação começando a partir da versão atualizada da linha.



Ciência da Computação

11

Nível de isolamento Read Committed

- ❑ Devido à regra acima é possível os comandos de atualização enxergarem instantâneos inconsistentes;
 - ❑ Podem enxergar os efeitos de comandos de atualização concorrentes que afetam as mesmas linhas que estão tentando atualizar, mas não enxergam os efeitos destes comandos em outras linhas do banco de dados.
 - ❑ Este comportamento torna o **Read Committed** menos apropriado para os comandos envolvendo condições de procura complexas.
 - ❑ É apropriado para casos mais simples.



Ciência da Computação

12

Nível de isolamento Read Committed
<ul style="list-style-type: none"> ❑ Por exemplo, considere a atualização do saldo bancário pela transação mostrada abaixo: <pre>BEGIN; UPDATE conta SET saldo = saldo + 100.00 WHERE num_conta = 12345; UPDATE conta SET saldo = saldo - 100.00 WHERE num_conta = 7534; COMMIT;</pre> ❑ Se duas transações deste tipo tentam mudar concorrentemente o saldo da conta 12345 é claro que se deseja que a segunda transação comece a partir da versão atualizada da linha da conta. ❑ Como cada comando afeta apenas uma linha predeterminada, permitir enxergar a versão atualizada da linha não cria nenhuma inconsistência problemática.

13

Nível de isolamento serializável
<ul style="list-style-type: none"> ❑ Serializável é o nível mais rigoroso de isolamento da transação. ❑ Este nível emula a execução serial da transação, como se todas as transações fossem executadas uma após a outra, em série, em vez de concorrentemente. ❑ Entretanto, as aplicações que utilizam este nível de isolamento devem estar preparadas para tentar novamente as transações devido a falhas na serialização.

15

Nível de isolamento serializável
<ul style="list-style-type: none"> ❑ Os comandos UPDATE, DELETE e SELECT FOR UPDATE se comportam do mesmo modo que o SELECT para encontrar as linhas de destino: somente encontram linhas de destino efetivadas até a hora do início da transação. ❑ Entretanto, alguma linha de destino pode ter sido atualizada (ou excluída ou marcada para atualização) por outra transação concorrente na hora que foi encontrada. ❑ Neste caso, a transação serializável aguarda a transação de atualização que começou primeiro efetivar ou desfazer as alterações (se ainda estiver executando).

17

Nível de isolamento Read Committed
<ul style="list-style-type: none"> ❑ O isolamento parcial da transação fornecido pelo modo Read Committed é adequado para muitas aplicações, e este modo é rápido e fácil de ser utilizado. ❑ Entretanto, para aplicações que efetuam consultas e atualizações complexas, pode ser necessário garantir uma visão consistente mais rigorosa do banco de dados que a fornecida pelo modo Read Committed.

14

Nível de isolamento serializável
<ul style="list-style-type: none"> ❑ Quando uma transação está no nível serializável, o comando a SELECT enxerga apenas os dados efetivados antes da transação começar; <ul style="list-style-type: none"> ❑ Nunca enxerga dados não efetivados ou mudanças efetivadas durante a execução da transação por transações concorrentes; ❑ Entretanto, o comando SELECT enxerga os efeitos de atualizações anteriores executadas dentro de sua própria transação, mesmo que ainda não tenham sido efetivadas. ❑ Isto é diferente do Read Committed, porque o comando SELECT enxerga um snapshot do início da transação, e não do início do comando corrente dentro da transação. ❑ Portanto, comandos SELECTs sucessivos dentro de uma mesma transação sempre enxergam os mesmos dados.

16

Nível de isolamento serializável
<ul style="list-style-type: none"> ❑ Se a transação desfizer as alterações, então seus efeitos são negados e a transação serializável pode prosseguir com a atualização da linha original encontrada. ❑ Porém, se as alterações forem efetivadas (e a linha for realmente atualizada ou excluída, e não apenas selecionada para atualização), então a transação serializável é desfeita com a mensagem ERROR: Can't serialize access due to concurrent update ❑ porque uma transação serializável não pode modificar linhas alteradas por outra transação após ter começado.

18

Nível de isolamento serializável
<ul style="list-style-type: none"> ❑ Quando uma aplicação recebe esta mensagem de erro, deve abortar a transação corrente e tentar executar novamente toda a transação a partir do início. ❑ Da segunda vez em diante, a transação passa a enxergar a modificação efetivada anteriormente como parte da sua visão inicial do BD; <ul style="list-style-type: none"> ❑ Portanto, não existirá conflito lógico em usar a nova versão da linha como o ponto de partida para a atualização na nova transação. ❑ Observe que somente as transações que atualizam podem precisar de novas tentativas; <ul style="list-style-type: none"> ❑ as transações apenas de leitura nunca ocasionam conflito de serialização.



Ciência da Computação

19

Nível de isolamento serializável
<ul style="list-style-type: none"> ❑ Serializável fornece uma garantia rigorosa que cada transação enxerga apenas visões completamente consistentes do BD. ❑ Aplicação precisa estar preparada para executar novamente as transações quando as atualizações concorrentes tornam impossível sustentar a ilusão de uma execução serial. ❑ Alto custo: é recomendado somente quando as transações possuem atualizações complexas que podem produzir respostas erradas no modo Read Committed. ❑ O modo serializável é necessário quando a transação realiza várias consultas sucessivas que necessitam enxergar visões idênticas do banco de dados.



Ciência da Computação

20

BLOQUEIOS

21

Bloqueios
<ul style="list-style-type: none"> ❑ O PostgreSQL fornece vários modos de bloqueio para controlar o acesso concorrente aos dados nas tabelas. ❑ Estes modos podem ser utilizados para o bloqueio controlado pela transação, nas situações onde o MVCC não fornece o comportamento adequado. ❑ Também, a maioria dos comandos do PostgreSQL obtém, automaticamente, bloqueios com modos apropriados para garantir que as tabelas referenciadas não serão excluídas ou modificadas de forma incompatível enquanto o comando executa; <ul style="list-style-type: none"> ❑ Por exemplo, o comando ALTER TABLE não pode executar concorrentemente com outras operações na mesma tabela.



Ciência da Computação

22

Bloqueios
<ul style="list-style-type: none"> ❑ Tipos de bloqueios: <ul style="list-style-type: none"> ❑ Bloqueios no nível de tabela ❑ Bloqueios no nível de linha ❑ Impasses (deadlocks)



Ciência da Computação

23

Bloqueios
<h4>Bloqueios no nível de tabela</h4> <ul style="list-style-type: none"> ❑ A única diferença real entre um modo de bloqueio e outro é o conjunto de modos de bloqueio com o qual cada um conflita. ❑ Duas transações não podem obter modos de bloqueio conflitantes na mesma tabela ao mesmo tempo; <ul style="list-style-type: none"> ❑ Entretanto, uma transação nunca conflita consigo mesma ❑ por exemplo, pode obter o bloqueio ACCESS EXCLUSIVE e mais tarde obter o bloqueio ACCESS SHARE na mesma tabela. ❑ Modos de bloqueio não conflitantes podem ser obtidos concorrentemente por muitas transações.



Ciência da Computação

24

Bloqueios

Bloqueios no nível de tabela

- ❑ Em particular, deve ser observado que alguns modos de bloqueio são auto-conflitantes;
 - ❑ por exemplo, o modo de bloqueio ACCESS EXCLUSIVE não pode ser obtido por mais de uma transação ao mesmo tempo,
- ❑ Enquanto outros não são auto-conflitantes
 - ❑ por exemplo, o modo de bloqueio ACCESS SHARE pode ser obtido por várias transações.
- ❑ Uma vez obtido, o modo de bloqueio é mantido até o fim da transação.



Ciência da Computação

25

Bloqueios

Bloqueios no nível de tabela

- ❑ Para examinar a lista de bloqueios correntemente mantidos pelo servidor de banco de dados, deve ser utilizada a visão do sistema pg_locks.
- ❑ Existe uma lista no nível de tabelas com pelo menos 8 bloqueios.
- ❑ De alguma forma os nomes refletem a utilização típica de cada modo de bloqueio --- mas as semânticas são todas as mesmas.



Ciência da Computação

26

Bloqueios

Bloqueios no nível de Linha

- ❑ Um bloqueio no nível de linha, para uma linha específica, é obtido automaticamente quando a linha é atualizada;
 - ❑ ou excluída ou marcada para atualização.
- ❑ O bloqueio é mantido até a transação efetivar ou desfazer as alterações.
- ❑ Os bloqueios no nível de linha não afetam a consulta aos dados; bloqueiam apenas escritas na mesma linha.



Ciência da Computação

27

Bloqueios

Bloqueios no nível de Linha

- ❑ Bloqueio no nível de linha sem na verdade modificar a linha, deve-se selecionar a linha por meio do comando **SELECT FOR UPDATE**.
- ❑ Após um determinado bloqueio ser obtido a transação pode **atualizar** a linha várias vezes sem que haja conflito.
- ❑ O PostgreSQL não guarda nenhuma informação em memória relativa às linhas modificadas, portanto não existe limite no número de linhas bloqueadas de uma vez.
 - ❑ Entretanto, o bloqueio de uma linha pode causar escrita no disco; por exemplo, o comando SELECT FOR UPDATE modifica as linhas selecionadas para marcá-las ocasionando escrita no disco.



Ciência da Computação

28

Bloqueios

Bloqueios no nível de Páginas

- ❑ Além dos bloqueios de tabela e de linha, também são utilizados bloqueios no nível de página, compartilhados e exclusivos, para controlar o acesso de leitura e gravação nas páginas da tabela no shared buffer pool.
- ❑ Estes bloqueios são liberados imediatamente após a tupla ser lida ou atualizada.
- ❑ Normalmente os desenvolvedores de aplicação não precisam se preocupar com bloqueios no nível de página.



Ciência da Computação

29

Bloqueios

Impasse (deadlocks)

- ❑ A utilização de bloqueios explícitos pode causar impasses (deadlocks), em particular quando duas (ou mais) transações mantêm bloqueios que outra deseja.
 - ❑ Por exemplo:
 - ❑ se a transação 1 obtém um bloqueio exclusivo na tabela A e, então, tenta obter um bloqueio exclusivo na tabela B,
 - ❑ enquanto a transação 2 já possui um bloqueio exclusivo na tabela B, e
 - ❑ agora tenta obter um bloqueio exclusivo na tabela A,
 - ❑ então nenhuma das duas transações pode continuar.



Ciência da Computação

30

Bloqueios

Impasse (deadlocks)

- ❑ O PostgreSQL detecta automaticamente as situações de impasse, resolvendo-as abortando uma das transações envolvidas, permitindo que a(s) outra(s) prossiga(m);
 - ❑ Exatamente qual transação é abortada é difícil prever, não se devendo confiar nesta previsão.
- ❑ Geralmente, a melhor defesa contra os impasses é evitá-los tendo certeza que todas as aplicações que utilizam o BD obtêm estes bloqueios em vários objetos em uma ordem consistente.



Ciência da Computação

31

Bloqueios

Impasse (deadlocks)

- ❑ Deve ser garantido, também, que o primeiro bloqueio de um objeto em uma transação seja aquele com o modo mais elevado que será necessário para este objeto.
- ❑ Se não for possível conhecer esta situação antecipadamente, então os impasses podem ser tratados em tempo de execução tentando novamente a execução das transações abortadas pelos impasses.



Ciência da Computação

32

Bloqueios

Impasse (deadlocks)

- ❑ Enquanto a situação de impasse não é detectada, uma transação aguardando um bloqueio no nível de tabela ou no nível de linha fica aguardando indefinidamente pela liberação do bloqueio conflitante.
- ❑ Por isso, é uma péssima ideia as aplicações manterem transações abertas por longos períodos;
 - ❑ por exemplo, aguardando a entrada de dados pelo usuário.



Ciência da Computação

33

Atividade

- ❑ Leia e documente sobre os diferentes modos de bloqueio no nível de tabela. Pesquise exemplos de aplicação desses bloqueios em SGBDs.
- ❑ Faça uma apresentação e poste no portfólio.
- ❑ Esta atividade irá compor a nota de participação da M2.



Ciência da Computação

34

Link

- ❑ <http://tecspace.com.br/paginas/aula/postgresql/guia/mvcc.html>



Ciência da Computação

35