

Locks

Os Mecanismos de Lock

Locks são projetados para permitir um alto nível de concorrência de dados, ou seja, muitos usuários podem de forma segura acessar os mesmos dados ao mesmo tempo.

No Oracle o mecanismo de lock garante que:

- Operações DML fazem bloqueios a nível de linha.
- Uma query não mantém locks, exceto se for especificado através do comando SELECT ...FOR UPDATE.

O Oracle nos provê vários níveis de consistência de dados, ou seja, usuários podem ver uma imagem estática dos dados, mesmo que estes estejam sendo atualizados por outro.

Transações mantém locks até um comando COMMIT ou ROLLBACK.

O Oracle não utiliza lock escalável de linha para tabela por exemplo.

Os locks do Oracle são eficientes e baratos e a grande maioria dos sites não reportam problemas de locking.

Porém locks podem causar contenção porque:

- Desenvolvedores especificam níveis de locks desnecessários
- Desenvolvedores codificam transações demasiadamente grandes
- Usuários não confirmam as alterações imediatamente
- A aplicação utiliza o Oracle em conjunto com outras aplicações que podem impor altos níveis de locks

Locks de DML

Dois tipos de estruturas de Locks são utilizadas em comandos DML.

- A transação obtém um lock compartilhado na tabela
- A transação obtém um lock exclusivo nas linhas que estão sendo modificadas

O Oracle mantém todos os locks enfileirados. Esse mecanismo, mantém as seguintes informações:

- Usuários que estão aguardando
- O modo de lock que eles desejam
- A ordem em que os usuários requisitaram seus locks

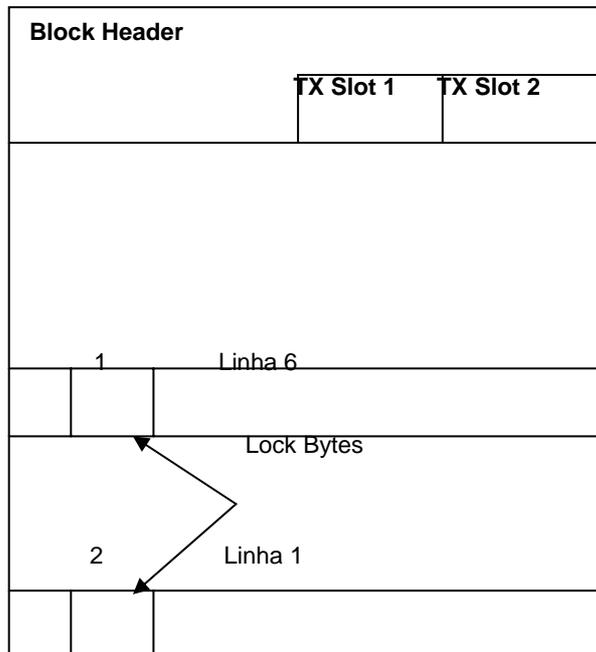
Logo, se três usuários desejam atualizar a mesma linha da mesma tabela, todos obtém um lock compartilhado da tabela, mas apenas um (o primeiro) obtém um lock exclusivo da linha. O mecanismo de enfileiramento sabe quem tem a linha lockada e quem a deseja.

Lock de DML a Nível de Bloco

Dentro de blocos, o Oracle mantém um identificador para cada transação ativa no header do bloco. A nível de linha, o byte de lock armazena um identificador da transação

Exemplo

No diagrama, a transação usando o slot 1, está lockando a linha 6 e a transação usando o slot 2 está lockando a linha 1.



Locks de DDL

Existem três tipos de Locks de DDL. São requeridos em modo NOWAIT e de forma muito rápida, apenas para a operação. Desta forma é improvável que consigamos visualizar uma contenção de Locks em DDL.

Locks DDL Exclusivos

Algumas operações DDL como CREATE, ALTER e DROP, tem que obter um lock exclusivo no objeto que estão trabalhando.

Os usuários não podem obter um lock exclusivo na tabela se um outro usuário mantém qualquer outro nível de lock. Um comando ALTER TABLE falha se existe algum usuário com uma transação aberta naquela tabela.

Locks DDL Compartilhados

Alguns comandos como GRANT, CREATE PACKAGE, necessitam de um lock DDL compartilhado no objeto referenciado.

Isto não evitará comandos DDL similares ou qualquer comando DML, mas evitará que outro usuário altere ou drop o objeto.

Outros Locks

Além dos locks exclusivos de tabela e exclusivos de linha, o Oracle provê outros tipos de locks.

Você pode requerê-los explicitamente com o comando LOCK TABLE, mas existem alguns outros comandos SQL que os requerem também.

Row Share (RS)

Pode-se requerer um lock de linha durante uma consulta usando a seguinte sintaxe:

- `SELECT FOR UPDATE;`

Nota:

- Neste caso você obtém um lock tipo Row Share que inclui:
 - Um lock de tabela compartilhado.
 - Um lock exclusivo nas linhas obtidas pela sua query, muito parecido com o Row Exclusive.

Share (S)

Este lock evita qualquer operação DML na tabela.

O comando SQL que implicitamente obtém um lock tipo Share, envolve restrição de integridade referencial. Se não houver um índice na coluna de chave estrangeira na tabela filho:

- Qualquer operação DML na tabela FILHO
- Gera lock para DELETE na tabela PAI

Para evitar este tipo de comportamento, crie um índice para a coluna chave estrangeira na tabela filho. Quando existe um índice para a chave estrangeira da tabela filho, o Oracle pode evitar alterações nas linhas através do lock sobre os valores alterados no índice.

Share Row Exclusive (SRX)

Este é o mais alto nível de lock de tabela, que evita operações DML e `SELECT... FOR UPDATE`.

Da mesma forma que o lock Share, o comando que obtém este tipo de lock, está envolvido com restrição de integridade referencial.

Porém, você precisa deste tipo de lock na tabela filho, quando existe uma deleção na tabela pai sob as seguintes circunstâncias:

- Uma restrição de chave estrangeira inclui `ON DELETE CASCADE`
- Não há nenhum índice na coluna de chave estrangeira na tabela filho

Novamente a solução é indexar a coluna de chave estrangeira.

Lock Explícito

Os desenvolvedores podem decidir lockar tabelas usando a seguinte sintaxe:

- `LOCK TABLE (<nome_tabela> [, ...]) IN <nome_modos> MODE [NOWAIT];`

Os valores possíveis para <nome_modos> são:

- `ROW SHARE` (*select ... for update*)
- `ROW EXCLUSIVE` (*linha*)
- `SHARE` (*foreign key*)
- `SHARE ROW EXCLUSIVE` (*foreign key ... on delete cascade*)
- `EXCLUSIVE` (*operações DDL*)

Existem razões para em determinadas aplicações se aplicar lock explícito, porém se você estiver observando algum tipo de contenção, contacte os desenvolvedores.

Desenvolvedores sem experiência prévia no ambiente Oracle costumam utilizar locks explícitos desnecessariamente.

Locks de Usuários

O Oracle provê a package DBMS_LOCK. Desenvolvedores podem usá-la para criar os seus próprios locks sobre recursos.

Geralmente é usada para passar recursos entre sessões.

Monitorando a Atividade de Locking

Podemos monitorar as atividades de locking usando:

- A view V\$LOCK
- O resultado do script *utllockt.sql*, que mostra usuário esperando por recursos lockados

Outras views relacionadas a locks são: DBA_LOCKS, DBA_DDL_LOCKS, DBA_DML_LOCKS, DBA_BLOCKERS e DBA_WAITERS. Para visualizarmos estas views e o script *utllockt.sql* precisamos rodar o script *catblock.sql* como usuário SYS.

Exemplo – Visualizando que está mantendo recursos

```
SQL> select * from dba_blockers;

HOLDING_SESSION
-----
              7

SQL> select username, sid from v$session;

USERNAME                                SID
-----                                -
                                             1
                                             2
                                             3
                                             4
                                             5
                                             6
ALUNO1                                    7
SYSTEM                                    8
ALUNO2                                    9
```

Exemplo – Sessões “esperando por locks” e sessões que estão bloqueando

```
SQL> SELECT WAITING_SESSION, HOLDING_SESSION,
2          LOCK_TYPE, MODE_HELD, MODE_REQUESTED
3 FROM    DBA_WAITERS;

WAITING_  HOLDING_  LOCK_  MODE_  MODE_
SESSION  SESSION  TYPE   HELD   REQUESTED
-----  -
10        8        DML    S/Row-X(SSX)  Row-X(SX)
8         9        DML    Row-X(SX)     S/Row-X(SSX)
```

Nota:

- Conforme visto no primeiro exemplo, consultando a view V\$SESSION, podemos obter o nome do usuário da sessão que está gerando o lock.
- No exemplo seguinte, podemos também obter o nome dos usuários das sessões em “waiting” e “holding”.

- Como dba, rode o script utllockt.sql em sua sessão para ter uma visão identada das sessões que estão aguardando por locks e de que tipos.

Manipulando Contensões

Eliminando Sessões (Kill Session)

Se um usuário está mantendo um lock requerido por outro usuário, você pode:

- Contatar o usuário para que seja feito um commit ou rollback;
- Eliminar a sessão Oracle. Isto fará um rollback e liberará os locks.

Qualquer método de monitoração detalhado acima, nos fornecerá o identificador da sessão do usuário.

Pelo Server Manager, o comando para eliminar sessões de usuários é:

```
ALTER SYSTEM KILL SESSION '<sid, serial#>'
```

Onde: sid e serial# são valores encontrados na V\$SESSION

Observe que eliminar sessões não tem efeito imediato.

Qual a Linha Gera Contenção

Se necessitamos saber qual linha está causando contenção, observe na view V\$SESSION as seguintes colunas:

- ROW_WAIT_BLOCK#
- ROW_WAIT_ROW#
- ROW_WAIT_FILE#
- ROW_WAIT_OBJ# -> identificador do objeto na DBA_OBJECTS (*object_id*)

Deadlocks

Um deadlock pode ocorrer quando dois ou mais usuários esperam por um dado que encontra-se em poder do outro respectivamente.

O Oracle automaticamente detecta e resolve problemas de deadlocks fazendo um rollback do comando que detectou o deadlock.

Transação 1	Tempo	Transação 2
UPDATE CARGO SET SAL_MIN = SAL_MIN*1.1 WHERE CODIGO_CARGO = 110	1	UPDATE CARGO SET SAL_MAX = SAL_MAX*2 WHERE CODIGO_CARGO = 105
UPDATE CARGO SET SAL_MIN = SAL_MIN*1.1 WHERE CODIGO_CARGO = 105	2	UPDATE CARGO SET SAL_MAX = SAL_MAX*2 WHERE CODIGO_CARGO = 110 (este comando irá provocar o deadlock)
ORA-00060:deadlock detected while waiting for resource	3	

Assumindo que o segundo comando update da transação 1 detectou o deadlock, o Oracle fará o rollback naquele comando e retornará a mensagem ORA-00060.

Se um deadlock ocorre, um arquivo de trace é preenchido. Ele conterá os lock holders e lock waiters, além do ROWID das linhas bloqueadas pelo lock. O arquivo de trace está no diretório especificado por USER_DUMP_DEST.

Nota:

- Deadlocks são menos prováveis de ocorrer quando aplicações requisitam os locks na mesma ordem e, também requisitam os locks mais restritivos primeiramente.
- Cuidado com aplicações onde vários usuários atualizam linhas na mesma tabela em ordem randômica. Isto pode gerar deadlock.

Consistência de Dados**Consistência de Leitura a Nível de Comando**

Toda query, sempre visualiza uma imagem consistente dos dados que ela seleciona. O Oracle usa o segmento de rollback para reconstruir os dados caso o SCN (System Change Number) da query seja mais antigo do que o SCN no qual o bloco foi alterado por último.

Consistência de Leitura a Nível de Transação

Se você necessita visualizar uma imagem consistente através de múltiplas queries, você pode usar o seguinte comando antes de emitir a primeira query:

- SET TRANSACTION READ ONLY;

Nota:

- Você não pode emitir comandos DML em transações Read-Only
- O Oracle usa o SCN inicial para a transação como base e reconstrói qualquer bloco modificado desde então, de forma a gerar uma imagem de leitura consistente.
- O modo default para transações é READ WRITE que também pode ser obtido através do comando SET TRANSACTION READ WRITE.