BLOQUEIOS

Além das transações existem outras formas de evitar que os dados das tabelas sejam lidos, alterados ou eliminados quando existem múltiplas sessões que concorrem para a mesma tabela. Este sistema não é idêntico para sistemas transacionais ou não transacionais, como são os casos dos formatos InnoDB e MyISAM, respetivamente.
O bloqueio de tabelas é algo que se utiliza sobretudo quando se trabalha com o formato MyISAM.
O bloqueio também terá efeito sobre as view.

Para tabelas **InnoDB**, O MySQL só utiliza bloqueio de tabelas se bloquearmos explicitamente a tabela com **LOCK TABLES** (usado para tabelas MyISAM) ou executarmos um comando quer irá modificar todos os registos na tabela, como **ALTER TABLE**. Para estes tipos de tabelas não é recomendável o **LOCK TABLES**.
Para aplicar os bloqueios devemos "desligar" o autocommit, colocando o seu valor a 0, da mesma forma que para aplicar o rollback e o commit quando o desejamos aquando das transações.

BLOQUEIOS InnoDB

Na utilização de um storage engine do tipo InnoDB, é possível alargar o conceito de transação utilizando o bloqueio de tabelas. Quando uma transação com InnoBD está em curso, é possível executar um comando SELECT, o que leva a que seja possível que os dados devolvidos não sejam os atuais, por o SELECT ser executado em simultâneo com a transação.

Tipos de bloqueios com InnoDB:

- **Shared lock** – situação em que há um **bloqueio durante a leitura do registo**.

Os pedidos de vários utilizadores podem ser satisfeitos e vão sendo bloqueados.
É possível inserir a instrução LOCK IN SHARE MODE*1 com vista a que os dados devolvidos sejam os mais atuais, permitindo leituras por outros utilizadores. Caso ocorram comandos de alteração ou eliminação durante um *shared Lock*, ficarão em fila de espera até que este termine, para depois poderem ser executados. Isto poderá resultar em *deadLock* (*)

SINTAXE:

```
SELECT…
LOCK IN SHARE MODE;
```

Exemplo:

```
SELECT * FROM Editoras WHERE NomeEditora = 'Sony+' LOCK IN SHARE MODE
```

- Exclusive lock – situação em que há um bloqueio durante a atualização ou eliminação de um registo.

SINTAXE (bloquear o registo para permitir uma ALTERAÇÃO):

```
SELECT…
FOR UPDATE;
```

SINTAXE (bloquear o registo para permitir uma ELIMINAÇÃO):

```
SELECT…
FOR DELETE;
```

Estes comandos são assumidos de imediato como sendo do tipo *exclusive lock,* impedindo que outros utilizadores possam bloquear o registo ou visualizá-lo. Contudo, se houver outro utilizador que concorra para o mesmo registo, terá de aguardar que a transação termine para que a sua se inicie. Se houver registos que estejam ligados a outros por chaves externas (foreign key), também estarão bloqueadas durante a transação.

Para terminar ambos os meios de bloqueio, basta efetuar um commit ou um rollback na transação atual.

(*)
DEADLOCK

Os deadlock são o resultado de conflitos derivados de bloqueios acionados em registos. Estes ocorrem quando dois ou mais processos concorrem entre si e se bloqueiam, esperando que cada um termine, podendo levar a uma situação de espera infinita.

Exemplo:

- Dois utilizadores X e Y acedem à base de dados em simultâneo.
- X inicia a sua tarefa criando a tabela EXEMPLO onde são inseridos dados.
- X acede à tabela EXEMPLO e inicia a sua leitura usando a instrução LOCK IN SHARE MODE, bloqueando a leitura da tabela EXEMPLO por `shared lock`.
- Entretanto, Y tenta eliminar a tabela EXEMPLO e cria um *exclusive lock*.
- Assim que X termine a sua leitura, Y eliminará a tabela EXEMPLO em *modo exclusivo*, mas X mantém-se em *shared mode* …
- Se X tentar eliminar a tabela ocorre um conflito, pois o utilizador Y ainda está em modo exclusivo à espera que X termine o modo partilhado. Mas X está a tentar realizar uma operação em modo exclusivo que Y tentou realizar antes, o que faz que este último tenha prioridade.

Este processo bloquearia, assim, a possibilidade de X realizar a eliminação da tabela, já que o processo de Y não tinha terminado, mas ao mesmo tempo Y também não pode terminar o processo porque estaria dependente de X.

Situações de Deadlock com dois utilizadores

Utilizador	Tempo T1	Tempo T2	Tempo T3	Tempo T4
X	Acesso à BD	SELECT… LOCK IN SHARE MODE		X aguarda que Y termine, pois este tem preferência em modo exclusivo
Y	Acesso à BD		SELECT… FOR DELETE	Y aguarda que X termine a transação em shared mode

É no tempo T4 que surge o deadlock, porque é a partir daqui que os dois utilizadores ficam dependentes um do outro para, à espera que que ambos os processos terminem para poderem finalizar as suas transações. O InnoDB consegue reconhecer este tipo de problema e fazer automaticamente um ROLLBACK sobre os comandos que originaram tal situação, cujo timeout é de 50 segundos. O ROLLBACK é normalmente aplicado na transação que movimenta um menor número de bytes.

BLOQUEIOS MyISAM

Como o sistema storage engine do MyISAM é não transacional (não são permitidas transações) para salvaguardar as tabelas e os seus dados são usados os bloqueios de tabelas.

Se for necessário efetuar muitas operações, o bloqueio das tabelas traduz-se num aumento de rapidez na inserção, alteração ou eliminação dos dados, terminando o bloqueio até que seja executado o comando UNLOCK TABLE. Assim, caso não sejam permitidos acessos simultâneos a uma tabela, ou que se queira salvaguardar que determinada tabela não pode ser modificada quando estiver a ser manipulada por determinado utilizador, pode aplicar-se o bloqueio por um determinado período de tempo.

Esta situação implica que este utilizador tenha acesso exclusivo, podendo alterar os dados da tabela, sem que os restantes utilizadores consigam fazê-lo até que este bloqueio seja anulado, podendo, no entanto, efetuar leituras, se tal lhes for permitido. A sequência de comandos é a seguinte:

1. Bloquear a tabela;
2. Ler, alterar e eliminar dados;
3. Desbloquear a tabela.

O Comando de bloqueio de tabela é do tipo **LOCK TABLE** mas poderão ser utilizadas instruções adicionais, conhecidas por **locktypes**:

WRITE – permite a leitura e a alteração da tabela, ficando todos os outros utilizadores bloqueados para a alteração ou leitura de dados da tabela. Só pode ser usado se não estiver outro bloqueio em curso, seja do tipo WRITE ou READ.

READ – permite que a sessão ativa leia a tabela, mas que não a altere, podendo haver várias sessões simultâneas com o `locktype` READ. Todas as sessões ativas acedem à tabela, mesmo que não usem a instrução READ, mantendo-se a restrição de não a poderem modificar.
Só pode ser usado um READ caso não esteja a decorrer um WRITE.

Exemplo (inserir um novo registo na tabela paciente):

```
LOCK TABLE paciente WRITE;
INSERT INTO paciente (nome, idade, telemóvel) VALUES ('Ana Moreira',67, 983828339);
UNLOCK TABLE
```

Caso se tivesse tentado inserir o registo com a tabela bloqueada para leitura (READ) ocorreria um erro!!!

Desbloqueio de tabelas

SINTAXE:

```
UNLOCK TABLE[S]
```

Verificar quais as tabelas em uso por bloqueio

SINTAXE:

```
SHOW OPEN TABLES
```

Instrução MySQL LOCK TABLES (para bloquear uma ou várias tabelas)

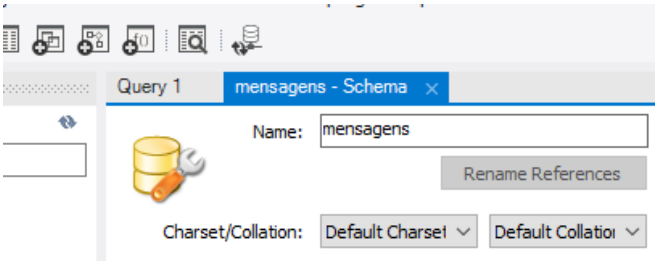
```
LOCK TABLES table_name1 [READ | WRITE],
        table_name2 [READ | WRITE],
                ...
```

Instrução MySQL UNLOCK TABLES (para desbloquear uma ou várias tabelas)

```
UNLOCK TABLES;
```

A. Cria duas sessões no MySQL WorkBench (A e B por exemplo). Clica com o botão direito do rato em cada uma delas associando-as a um grupo de trabalho.

1. Na sessão A, cria uma base de dados de nome **mensagens** (new schema).



2. Acede à base de dados e usa a função CONNECTION_ID () para obter o ID de ligação atual, da seguinte forma:

```
SELECT CONNECTION_ID();
```

CONNECTION_ID()
9

3. Cria a tabela mensagens com os campos seguintes (botão direito em cima de Table):

   Id, inteiro, e chave primária;
   Mensagem, string de tamanho variável, com no máximo 100 carateres e não nulo.

4. Insere o seguinte registo na tabela mensagens:

   Campo mensagem, valor "Hello"

id	mensagem
1	Hello

5. Visualiza o conteúdo da tabela.
6. Bloqueia a tabela mensagens em modo de leitura: LOCK TABLE mensagens READ;

7. E de seguida Tenta inserir uma nova linha na tabela mensagem:

   Campo mensagem, valor "Bye A".
   Executa o script

8. Qual o erro emitido pelo SGBD?_____
   O que significa?_____
   _____

9. Vamos aceder à base de dados numa sessão diferente(**B**). Acede à base de dados e usa a função CONNECTION_ID () para obter o ID de ligação atual.

10. Visualiza o conteúdo da tabela.

id	mensagem
1	Hello

11. Tenta inserir uma nova linha na tabela mensagem:
    Campo mensagem, valor "Bye B"
    É apresentado um output semelhante ao seguinte:

#	Time	Action	Message
1	10:36:37	insert into mensagens values (2,'Bye')	Running...

12. O que significa este output? _____
    _____

13. A partir da primeira sessão, usa a instrução SHOW PROCESSLIST para mostrar informações detalhadas: `SHOW PROCESSLIST;`

    Aparecerá um output semelhante ao seguinte:

Id	User	Host	db	Command	Time	State	Info
42	root	localhost:49961	bloqueios	Sleep	51		NULL
43	root	localhost:49962	bloqueios	Query	899	Waiting for table metadata lock	insert into mensagens values (2,'Bye')

14. Depois disso, liberta o bloqueio usando a instrução UNLOCK TABLES. Depois de libertares o bloqueio de leitura da primeira sessão, a operação INSERT na segunda sessão é executada. select * from mensagens

15. Por fim, verifica os dados da tabela de mensagens para ver se a operação INSERT da segunda sessão realmente foi executada.

    Deverão aparecer os dados seguintes na tabela mensagens:

id	mensagem
1	Hello
2	Bye B
NULL	NULL

16. Efetua um bloqueio para escrita, na primeira sessão, para a tabela Mensagens

```
LOCK TABLE mensagens WRITE;
```

17. Tenta inserir uma nova linha na tabela mensagem:

    Campo mensagem, valor "Good day A"

id	mensagem
1	Hello
2	Bye B
3	Goog Day A

18. Visualiza os dados da tabela mensagem.

19. Depois disso, na segunda sessão, tenta gravar e ler dados:

```
Insert into mensagens values (4,'Good Day B')
SELECT * FROM mensagens;
```

20. O MySQL coloca estas operações num estado de espera. Verifica-lo usando a instrução SHOW PROCESSLIST.

Id	User	Host	db	Command	Time	State	Info
3	root	localhost:50084	bloqueios	Sleep	473		NULL
4	root	localhost:50085	bloqueios	Sleep	140		NULL
5	root	localhost:50086	bloqueios	Sleep	287		NULL
12	root	localhost:50108	bloqueios	Query	134	Waiting for table metadata lock	SELECT * FROM mensagens
14	root	localhost:50114	bloqueios	Query	49	Waiting for table metadata lock	insert into mensagens values
15	root	localhost:50115	bloqueios	Query	0	starting	show processlist

21. Liberta a tabela a partir da 1.ª sessão.
    Todas as operações pendentes da segunda sessão são executadas.

id	mensagem
1	Hello
2	Bye B
3	Goog Day A
4	Goog Day B

22. Quais as diferenças entre bloquear uma tabela para leitura ou para escrita?

Bibiografia

Damas, L. (2005). SQL. Lisboa: FCA

Tavares, F. (2015). MySql. Lisboa: FCA

http://sqlparatodos.com.br/locks-mysql/

https://dev.mysql.com/doc/refman/5.5/en/innodb-locking-reads.html

http://www.mysqltutorial.org/mysql-table-locking/