bobby · 更新于 2022-11-16

上一节 3. 为什么需要分... 5. 基于redis实... 下一节

条件

> FOR UPDATE 仅适用于InnoDB存储引擎，且必须在事务区块(BEGIN/COMMIT)中才能生效。

mysql默认情况下每个sql都是单独的一个事务，并且是自动提交事务。

测试之前需要设置成非自动提交事务，不然无法模拟并发访问:

代码块 预览 复制

```
mysql> select @@autocommit;
+--------------+
| @@autocommit |
+--------------+
|            1 |
+--------------+
1 row in set (0.00 sec)
mysql> set autocommit = 0;
Query OK, 0 rows affected (0.00 sec)
mysql> select @@autocommit;
+--------------+
| @@autocommit |
+--------------+
|            0 |
+--------------+
1 row in set (0.00 sec)
```

此修改只针对当前窗口有效，重新打开的新窗口依然是自动提交事务的
所以要就需要两个窗口，窗口a：非自动提交事务，用于for update操作；
窗口b：用于普通update操作。

测试

我们有一数据库 test1，有一张表testa ，有自增主键ID，name，id_card

表中有两条数据

代码块

```
mysql> select * from testa;
+----+-------+--------------------+
```

```
+----+-------+--------------------+
|  1 | wangb | 322343256564545754 |
|  2 | shuna | 320990348823998792 |
+----+-------+--------------------+
2 rows in set (0.00 sec)
mysql> desc testa;
+---------+-------------+------+-----+---------+----------------+
| Field   | Type        | Null | Key | Default | Extra          |
+---------+-------------+------+-----+---------+----------------+
| id      | int(11)     | NO   | PRI | NULL    | auto_increment |
| name    | varchar(10) | NO   |     | NULL    |                |
| id_card | varchar(18) | YES  | UNI | NULL    |                |
+---------+-------------+------+-----+---------+----------------+
3 rows in set (0.00 sec)
```

1.只明确主键

- 有数据

在a窗口进行开启事务，对id为1的数据进行 for update，此时并没有commit；

代码块

```
mysql> begin;
Query OK, 0 rows affected (0.00 sec)
mysql> select * from testa where id = 1 for update;
+----+------+--------------------+
| id | name | id_card            |
+----+------+--------------------+
|  1 | wang | 322343256564545754 |
+----+------+--------------------+
1 row in set (0.00 sec)
mysql>
```

在b窗口对id=1的数据进行update name操作，发现失败：等待锁释放超时

代码块

```
mysql> update testa set name = "wangwang" where id = 1;
ERROR 1205 (HY000): Lock wait timeout exceeded; try restarting transac
```

再对id=2的数据进行update name操作，发现成功

代码块

```
mysql> update testa set name = "shunshun" where id = 2;
Query OK, 1 row affected (0.00 sec)
Rows matched: 1  Changed: 1  Warnings: 0
```

a窗口commit；之后，b窗口update操作都显示正常

- 无数据

a窗口 select for update 无数据

代码块

```
mysql> begin;
Query OK, 0 rows affected (0.00 sec)
mysql> select * from testa where id = 3
    -> ;
```

```
6    Empty set (0.00 sec)
     mysql>
```

b窗口，对两条数据update操作都是成功

代码块

```
1    mysql> update testa set name = "wanga" where id = 1;
2    Query OK, 1 row affected (0.01 sec)
3    Rows matched: 1  Changed: 1  Warnings: 0
4    mysql> update testa set name = "shun" where id = 2;
5    Query OK, 1 row affected (0.00 sec)
6    Rows matched: 1  Changed: 1  Warnings: 0
```

得出结论

> 明确主键并且有数据的情况下:mysql -> row lock;

> 明确主键无数据的情况下:mysql -> no lock;

2.明确主键和一个普通字段

- 有数据

将数据还原之后，

在a窗口进行开启事务，对id=1,name='wang'的数据进行 for update，此时并没有commit;

代码块

```
1    mysql> begin;
2    Query OK, 0 rows affected (0.00 sec)
3    mysql> select * from testa where id=1 and name = 'wang' for update
4        -> ;
5    +----+------+--------------------+
6    | id | name | id_card            |
7    +----+------+--------------------+
8    |  1 | wang | 322343256564545754 |
9    +----+------+--------------------+
10   1 row in set (0.03 sec)
11   mysql>
```

b窗口，对进行for update的那条数据的update操作无效（等待锁释放超时），其他的行的update操作正常

代码块

```
1    mysql> update testa set name = "wanga" where id = 1;
2    ERROR 1205 (HY000): Lock wait timeout exceeded; try restarting transact
3    mysql> update testa set name = "shunshun" where id = 2;
4    Query OK, 1 row affected (0.01 sec)
5    Rows matched: 1  Changed: 1  Warnings: 0
```

a窗口commit；之后，b窗口update操作都显示成功

- 无数据

同第一种情况的无数据测试
得出结论

> 明确主键和一个普通字段有数据的情况下:mysql -> row lock;

> 明确主键和一个普通字段无数据的情况下:mysql -> no lock;

3.明确一个普通字段

- 有数据

  将数据还原之后，

在a窗口进行开启事务，对name='wang'的数据进行 for update，此时并没有commit；

代码块

```
mysql> begin;
Query OK, 0 rows affected (0.00 sec)
mysql> select * from testa where name = 'wang' for update;
+----+------+--------------------+
| id | name | id_card            |
+----+------+--------------------+
|  1 | wang | 322343256564545754 |
+----+------+--------------------+
1 row in set (0.00 sec)
mysql>
```

b窗口，对进行for update的那条数据的update操作失败（等待锁释放超时），其他的行的update操作也显示失败（等待锁释放超时）

代码块

```
mysql> update testa set id_card = '222' where id = 1;
ERROR 1205 (HY000): Lock wait timeout exceeded; try restarting transact
mysql> update testa set id_card = '333' where id = 2;
ERROR 1205 (HY000): Lock wait timeout exceeded; try restarting transact
```

a窗口commit；之后，b窗口update操作都显示成功

- 无数据

同第一种情况的无数据测试
得出结论

> 只明确一个普通字段有数据的情况下:mysql -> table lock;

> 只明确一个普通字段无数据的情况下:mysql -> no lock;

4.明确一个unique字段

- 有数据

将数据还原之后，

在a窗口进行开启事务，对id_card='111'的数据进行 for update，此时并没有commit；

代码块

```
mysql> begin;
Query OK, 0 rows affected (0.00 sec)
mysql> select * from testa where id_card='111' for update;
+----+------+---------+
| id | name | id_card |
+----+------+---------+
|  1 | wang | 111     |
+----+------+---------+
1 row in set (0.00 sec)
mysql>
```

b窗口，对进行for update的那条数据的update操作失败（等待锁释放超时），其他的行的update操作显示正常！！

代码块

```
mysql> update testa set id_card = '222' where id = 1;
ERROR 1205 (HY000): Lock wait timeout exceeded; try restarting transact
mysql> update testa set id_card = '333' where id = 2;
Query OK, 1 row affected (0.00 sec)
```

- 无数据

同第一种情况的无数据测试
得出结论

> 只明确一个unique字段有数据的情况下:mysql -> row lock;

> 只明确一个unique字段无数据的情况下:mysql -> no lock;

思考

为什么对主键和unique字段进行for update操作的时候，mysql进行的是row lock；而对普通字段for update操作的时候进行的是table lock，是根据什么判断呢?

primary key和unique的共同特点是mysql会自动为其创建索引，他们都有索引，那把name字段创建索引，是不是就进行row lock呢?
查看表中的索引:

代码块

```
mysql> show keys from testa\G;
*************************** 1. row ***************************
        Table: testa
   Non_unique: 0
     Key_name: PRIMARY
  Seq_in_index: 1
```

```
8       Collation: A
9     Cardinality: 2
10       Sub_part: NULL
11         Packed: NULL
12           Null:
13     Index_type: BTREE
14        Comment:
15  Index_comment:
16  *************************** 2. row ***************************
17         Table: testa
18    Non_unique: 0
19      Key_name: id_card
20   Seq_in_index: 1
21    Column_name: id_card
22      Collation: A
23    Cardinality: 2
24       Sub_part: NULL
25         Packed: NULL
26           Null: YES
27     Index_type: BTREE
28        Comment:
29  Index_comment:
30  2 rows in set (0.00 sec)
31  ERROR:
32  No query specified
```

发现testa表中的索引只包含了id，id_card

添加name字段的索引

代码块

```
1   mysql> alter table testa add index index_name (name);
2   Query OK, 0 rows affected (0.03 sec)
3   Records: 0  Duplicates: 0  Warnings: 0
```

查看建表语句：

代码块

```
1   mysql> show create table testa \G;
2   *************************** 1. row ***************************
3          Table: testa
4   Create Table: CREATE TABLE `testa` (
5     `id` int(11) NOT NULL AUTO_INCREMENT,
6     `name` varchar(10) NOT NULL,
7     `id_card` varchar(18) DEFAULT NULL,
8     PRIMARY KEY (`id`),
9     UNIQUE KEY `id_card` (`id_card`),
10    KEY `index_name` (`name`)
11  ) ENGINE=InnoDB AUTO_INCREMENT=3 DEFAULT CHARSET=utf8
12  1 row in set (0.00 sec)
13  ERROR:
14  No query specified
```

发现name字段已经创建了普通索引index_name

在a窗口,对name字段再进行一次for update测试,不commit

代码块

```
1   mysql> begin;
2   Query OK, 0 rows affected (0.00 sec)
3   mysql> select * from testa where name = 'wang' for update;
4   +----+------+----------+
```

```
|  1 | wang | 222      |
+----+------+----------+
1 row in set (0.01 sec)
mysql>
```

在b窗口 对进行for update的数据进行update操作失败（锁释放等待超时）

代码块

```
mysql> update testa set id_card = '111' where id = 1;
ERROR 1205 (HY000): Lock wait timeout exceeded; try restarting transact
```

在b窗口 对其他行数据进行update操作,成功！！！

代码块

```
mysql> update testa set id_card = '4353' where id = 2;
Query OK, 1 row affected (0.02 sec)
Rows matched: 1  Changed: 1  Warnings: 0
```

a窗口commit之后，在b敞口操作正常

总结

select … for update; 操作

> 未获取到数据的时候，mysql不进行锁 （no lock）
> 获取到数据的时候，进行对约束字段进行判断，存在有索引的字段则进行row lock
> 否则进行 table lock

注意
当使用 ‘<>’,‘like’等关键字时，进行for update操作时，mysql进行的是table lock

网上其他博客说是因为主键不明确造成的，其实并非如此；

mysql进行row lock还是table lock只取决于是否能使用索引，而 使用’<>’,'like’等操作时，索引会失效，自然进行的是table lock；
什么情况索引会失效:
1.负向条件查询不能使用索引

负向条件有：!=、<>、not in、not exists、not like 等。
2.索引列不允许为null

单列索引不存null值，复合索引不存全为null的值，如果列允许为 null，可能会得到不符合预期的结果集。
3.避免使用or来连接条件

应该尽量避免在 where 子句中使用 or 来连接条件，因为这会导致索引失效而进行全表扫描，虽然新版的MySQL能够命中索引，但查询优化耗费的 CPU比in多。
4.模糊查询

以上情况索引都会失效，所以进行for update的时候，会进行table lock

参考：https://juejin.im/post/5b14e0fd6fb9a01e8c5fc663

再思考

为什么存在索引，mysql进行row lock，不存在索引，mysql进行table lock?

这是存储引擎InnoDB特性决定的：

InnoDB这种行锁实现特点意味着：只有通过索引条件检索数据，InnoDB才会使用行级锁，否则，InnoDB将使用表锁！

再总结

在上述例子中 ，我们使用给name字段加索引的方法，使表锁降级为行锁，不幸的是这种方法只针对 *属性值重复率低* 的情况。当属性值重复率很高的时候，索引就变得低效，MySQL 也具有自动优化 SQL 的功能。低效的索引将被忽略。就会使用表锁了。

3. 为什么需要分布式锁 上一节 下一节 5. 基于redis实现分布式锁

索引目录