

# 锁

# 目录

1 二级索引上的锁

2 覆盖索引和锁

3 备份和锁

# 二级索引上的锁

```
CREATE TABLE `t` (  
  `id` int(11) NOT NULL,  
  `c` int(11) DEFAULT NULL,  
  `d` int(11) DEFAULT NULL,  
  PRIMARY KEY (`id`),  
  KEY `c` (`c`)  
) ENGINE=InnoDB;
```

```
insert into t values(0,0,0),(5,5,5),  
(10,10,10),(15,15,15),(20,20,20),(25,25,25);
```

# 基础知识回顾：覆盖索引

1. `select id from t where c=5;`
2. `select c+id from t where c<=5;`
3. `select c from t where c+1<=20;`

# 课堂练习：二级索引上的锁

session A	session B	session C
begin; select id from t where c=5 lock in share mode;		
	update t set d=d+1 where id=5;	
		insert into t values(7,7,7);

- session2 和 sesion3 哪个语句会被锁?
- A. 只有 session2 被锁
  - B. 只有 session3 被锁
  - C. session2 和 session3 都被锁

# 二级索引上的锁

ENGINE_TRANSACTION_ID	THREAD_ID	OBJECT_NAME	index_name	OBJECT_INSTANCE_BEGIN	LOCK_TYPE	LOCK_MODE	LOCK_STATUS	LOCK_DATA
421865680507880	79	t	NULL	140390602699968	TABLE	IS	GRANTED	NULL
421865680507880	79	t	c	140390602697344	RECORD	S,GAP	GRANTED	10, 10
421865680507880	79	t	c	140390602696992	RECORD	S	GRANTED	5, 5

3 rows in set (0.01 sec)

{5,5},{10,10}  
{0,0},{5,5}

# 课堂练习：二级索引上的锁

session A	session B	session C
begin; select id from t where c=5 lock in share mode;		
	update t set d=d+1 where id=5;	
		insert into t values(7,7,7);



# 课堂练习：二级索引上的锁

```
CREATE TABLE `t` (  
  `id` int(11) NOT NULL,  
  `c` int(11) DEFAULT NULL,  
  `d` int(11) DEFAULT NULL,  
  PRIMARY KEY (`id`),  
  unique KEY `c` (`c`)  
) ENGINE=InnoDB;
```

```
insert into t values(0,0,0),(5,5,5),  
(10,10,10),(15,15,15),(20,20,20),(25,25,25);
```



# 课堂练习：begin; select id from t where c=5 for share

ENGINE_TRANSACTION_ID	THREAD_ID	OBJECT_NAME	index_name	OBJECT_INSTANCE_BEGIN	LOCK_TYPE	LOCK_MODE	LOCK_STATUS	LOCK_DATA
421865680507880	79	t	NULL	140390602699968	TABLE	IS	GRANTED	NULL
421865680507880	79	t	c	140390602696992	RECORD	S,REC_NOT_GAP	GRANTED	5, 5

2 rows in set (0.01 sec)

# 课堂练习： begin; select id from t where c=5 for update

ENGINE_TRANSACTION_ID	THREAD_ID	OBJECT_NAME	index_name	OBJECT_INSTANCE_BEGIN	LOCK_TYPE	LOCK_MODE	LOCK_STATUS	LOCK_DATA
4892	79	t	NULL	140390602699968	TABLE	IX	GRANTED	NULL
4892	79	t	c	140390602696992	RECORD	X,REC_NOT_GAP	GRANTED	5, 5
4892	79	t	PRIMARY	140390602697344	RECORD	X,REC_NOT_GAP	GRANTED	5

3 rows in set (0.01 sec)

# 二级索引上的数据版本

```
update t set c=c*100 where id=5;
```

课堂练习：二级索引上的数据要不要判断版本？

MVCC 怎么做？

```
beign
```

```
select id from t where c=5;
```

```
select id from t where c=5;
```

课堂练习：构造一个执行序列，产生以下现象（可以引入其他线程辅助构造）

```
mysql> show processlist;
```

Id	User	Host	db	Command	Time	State	Info
12	root	localhost:39856	test	Query	25	User sleep	select sleep(50) from t
14	root	localhost:39912	test	Query	22	Waiting for table flush	select * from t limit 1
15	root	localhost:39922	test	Query	0	init	show processlist

# 课堂练习：FTWRL

session1	session2	session3	session4
connect	connect	connect	connect
begin; select sleep(50) from t			
	flush tables with read lock //Waiting for table flush		
		select * from t limit 1 //blocked	
			kill query sessio2_id kill session2_id //继续block session3

# 课堂练习：FTWRL

session1	session2	session3	session4
connect	connect	connect	connect
select sleep(50) from t			
	flush table t ; //blocked		
		select * from t limit 1 //blocked	
			kill query sessio2_id kill session2_id show processlist

```
mysql> show processlist;
+-----+-----+-----+-----+-----+-----+-----+-----+
| Id | User | Host           | db  | Command | Time | State           | Info                                |
+-----+-----+-----+-----+-----+-----+-----+-----+
| 12 | root | localhost:39856 | test | Query   | 25   | User sleep      | select sleep(50) from t            |
| 14 | root | localhost:39912 | test | Query   | 22   | Waiting for table flush | select * from t limit 1          |
| 15 | root | localhost:39922 | test | Query   | 0    | init            | show processlist                   |
+-----+-----+-----+-----+-----+-----+-----+-----+
```



# mysqldump 逻辑备份, 常用 FTWRL

有myisam表

flush tables with read lock;

select my1

select my2

start transaction with consistent snapshot;

unlock tables;

select InnoDB tables

commit;

HTAP有其他引擎

flush tables with read lock;

//snapshot for engine1

//snapshot for engine2

start transaction with consistent snapshot;

unlock tables;

select InnoDB tables

select tables of engineX

commit;

开了GTID

flush tables with read lock;

select @@global.gtid\_execute;

start transaction with consistent snapshot;

unlock tables;

mysqldump --master-data

flush tables with read lock;

show master status;

start transaction with consistent snapshot;

unlock tables;



# 课堂练习: 从库备份怎么优化?

```
stop slave; mysqldump; sleep(..);start slave;
```

# mysqldump 逻辑备份，常用 FTWRL

session1	session2	session3
connect	connect	connect
update t set c=c+sleep(50)		
	flush tables with read lock //Waiting for global read lock	
		select * from t limit 1 //0.00s

# 优化备份逻辑

1. 判断是否有长时间的 `select` 语句，有的话稍后再试
2. 没有的话，启动 `mysqldump`，加 `ftwrl`，然后 `kill` 掉超过 `N` 秒的查询语句
3. `ftwrl` 配置超时退出

## 课堂练习：从主键更新索引

```
binlog_row_image=full/minimal
```

```
begin;
```

```
select d from t where id=5;
```

```
update t set c=5 where id=5;
```

```
for 10000
```

```
update t set d=0 where
```

```
id=5;
```

# FTWRL 先加 read lock, 再执行 flush table

session1	session2	session3
connect	connect	connect
select sleep(50) from t		
	flush tables with read lock;	
		create table t3 like t2 //block, 记为语句A
	ctrl+c //语句A继续被block	
	断开或 unlock tables //此时语句A执行通过	

# Q&A

THANKS