第二十九讲：MVCC 和事务隔离级别的设计与实现

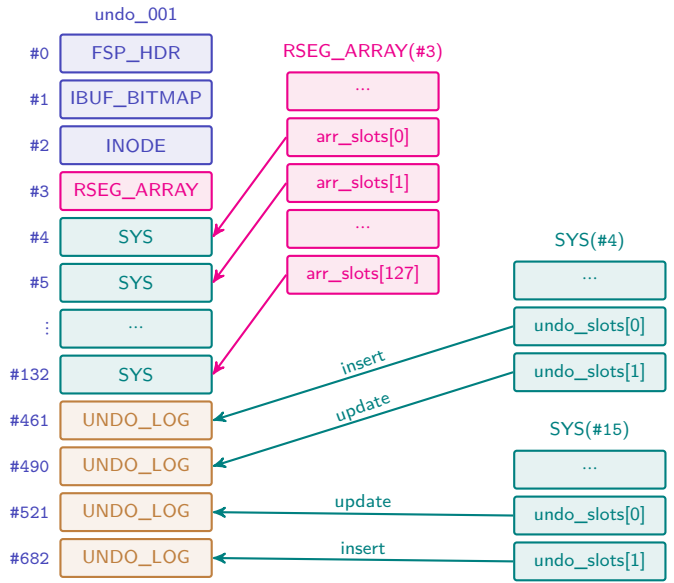
知春路遇上八里桥

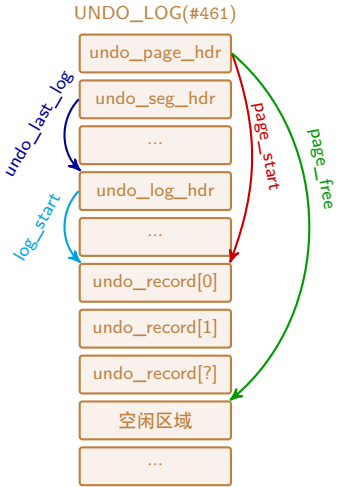
<2024-10-28 Mon>

1

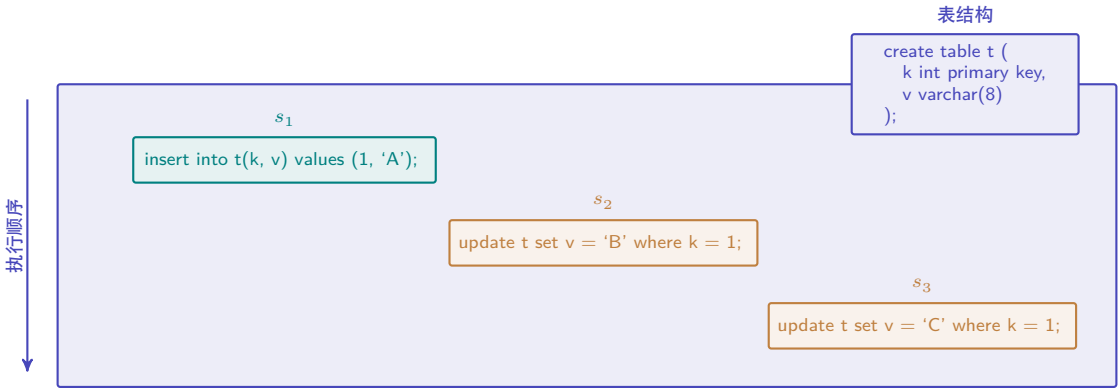
UndoLog 和 roll_ptr

回顾 undo_001 物理结构





单记录插入和更新场景


表结构
create table t (
k int primary key,
v varchar(8)
);
s_1
insert into t(k, v) values (1, 'A');
s_2
update t set v = 'B' where k = 1;
s_3
update t set v = 'C' where k = 1;
执行顺序

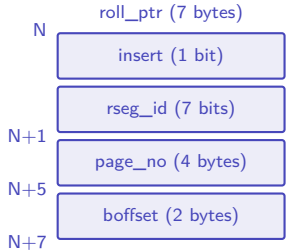
roll_ptr 回滚段指针

- trx_undo_report_row_operation() 函数，
  ☞ … /trx/trx0rec.cc

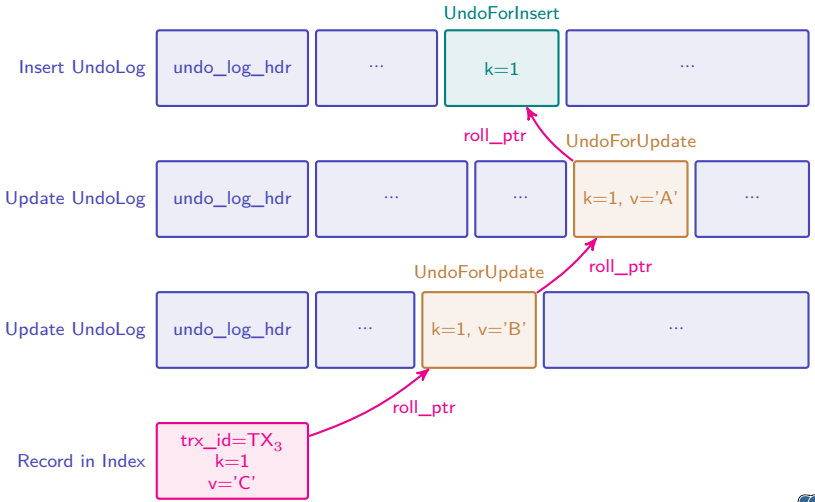
```
2319  *roll_ptr =
2320      trx_undo_build_roll_ptr(op_type == TRX_UNDO_INSERT_OP,
2321                              undo_ptr->rseg->space_id, page_no, offset);
```

  - insert 标记是否插入 (Insert) 还是更新 (Update)
  - rseg_id Undo 表空间 ID
  - page_no 所在页码
  - boffset 页内偏移
- rseg_id 解析函数，
  ☞ storage/innobase/include/trx0purge.h

```
229  inline space_id_t id2num(space_id_t space_id) {
230    if (!is_reserved(space_id)) {
231      return (space_id);
232    }
233
234    return (((dict_sys_t::s_max_undo_space_id - space_id) %
235             FSP_MAX_UNDO_TABLESPACES) +
236            1);
237  }
```

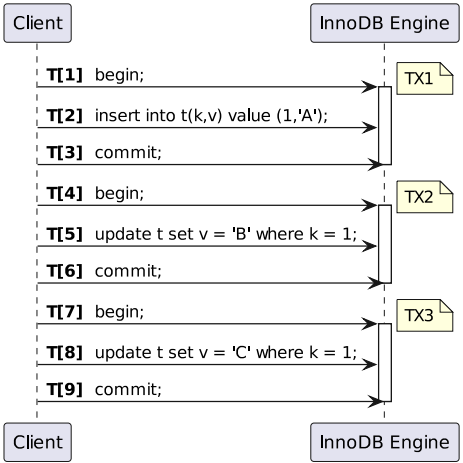


UndoLog 构成的版本链

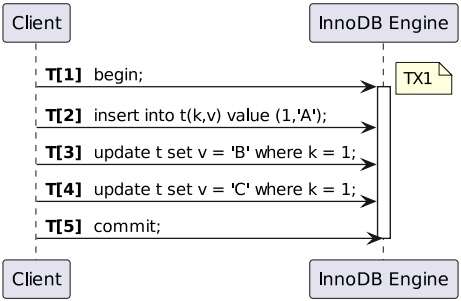


事务对 Undo 版本链影响的场景

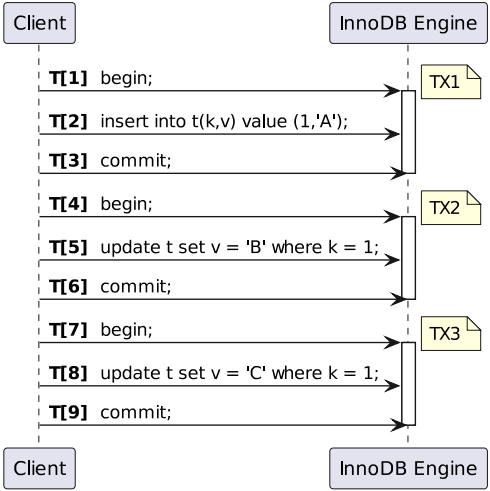
场景一：每个语句位于不同的事务内

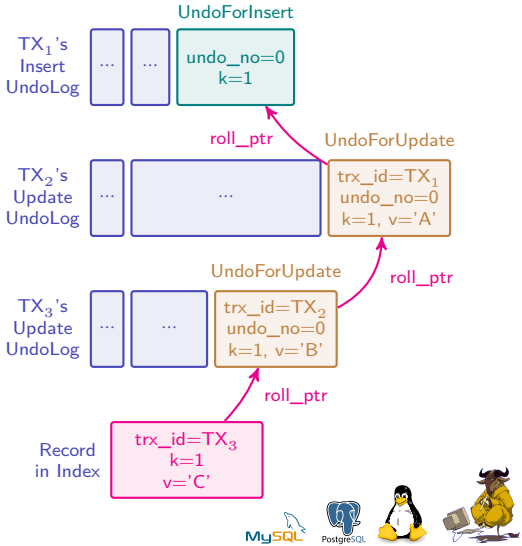


场景二：所有语句位于相同的事务内

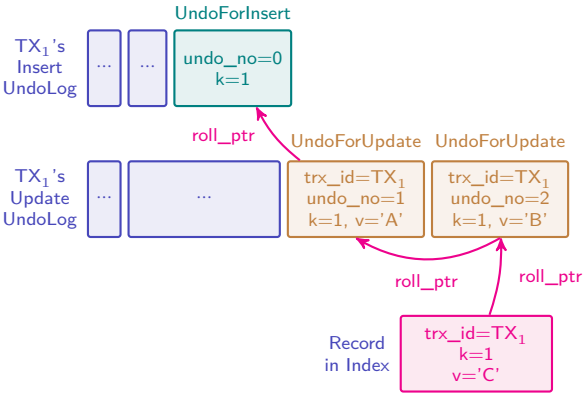


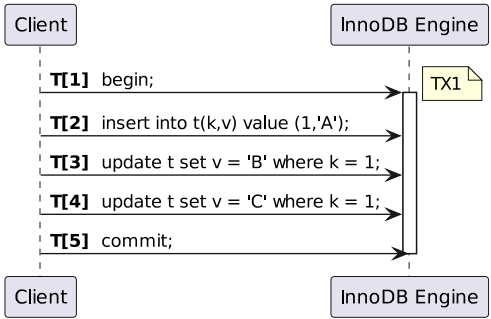
场景一：不同事务产生的 Undo 版本链





场景二：相同事务产生的 Undo 版本链

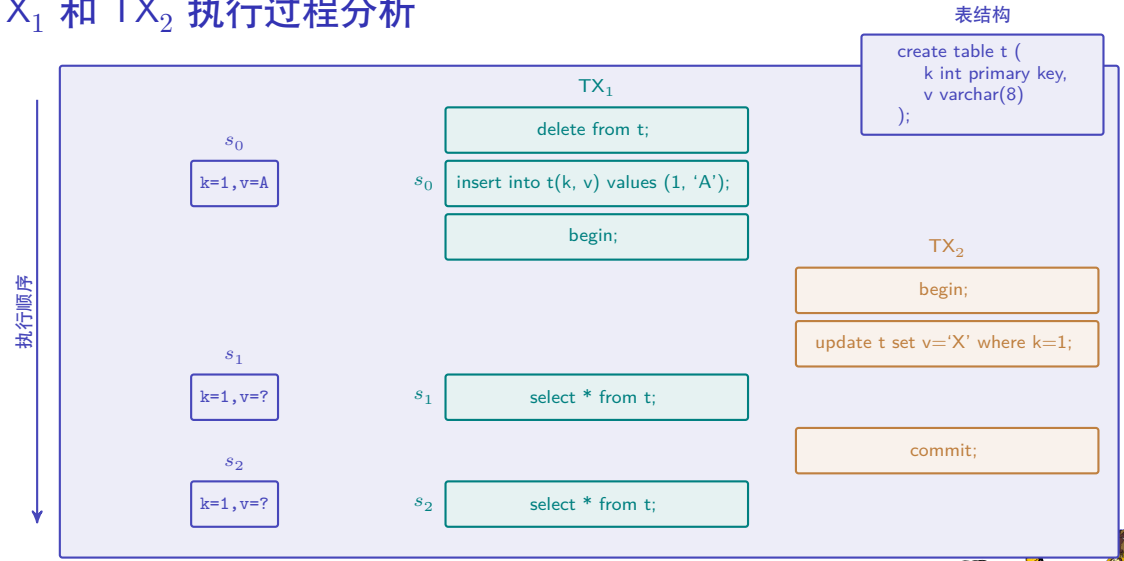




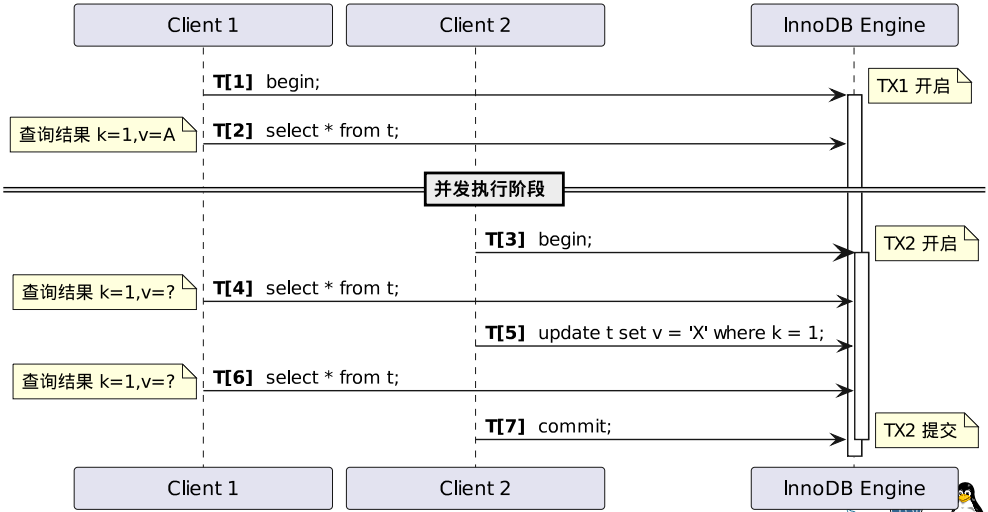
2

事务隔离级别

TX_1 和 TX_2 执行过程分析



多事务执行时序图


Client 1
Client 2
InnoDB Engine
T[1] begin;
TX1 开启
查询结果 k=1,v=A
T[2] select * from t;
并发执行阶段
T[3] begin;
TX2 开启
查询结果 k=1,v=?
T[4] select * from t;
T[5] update t set v = 'X' where k = 1;
查询结果 k=1,v=?
T[6] select * from t;
T[7] commit;
TX2 提交
Client 1
Client 2
InnoDB Engine

RR 可重复读

INIT	TX1	TX2
mysql> show variables like '%iso%'\G	mysql> begin;	mysql>
********** 1. row **********	mysql>	mysql> begin;
Variable_name: transaction_isolation	mysql>	mysql> update t set v = 'X' where k = 1;
Value: REPEATABLE-READ	mysql>	Query OK, 1 row affected (0.00 sec)
1 row in set (0.02 sec)	mysql>	Rows matched: 1 Changed: 1 Warnings: 0
	mysql> select * from t;	
mysql> select * from t;	+---+------+	mysql>
+---+------+	| k | v |	mysql>
| k | v |	+---+------+	mysql>
+---+------+	| 1 | A |	mysql>
| 1 | A |	+---+------+	mysql>
+---+------+	1 row in set (0.00 sec)	mysql> commit;
1 row in set (0.00 sec)		Query OK, 0 rows affected (0.00 sec)
	mysql>	
mysql>	mysql> select * from t;	mysql>
	+---+------+	mysql>
	| k | v |	mysql>
	+---+------+	mysql>
	| 1 | A |	mysql>
	+---+------+	mysql>
	1 row in set (0.00 sec)	mysql>
		mysql>
	mysql> commit;	mysql>
	Query OK, 0 rows affected (0.00 sec)	mysql>

RC 读已提交

INIT	TX1	TX2
mysql> show variables like '%iso%'\G	mysql> begin;	mysql>
*************** 1. row ***************	mysql>	mysql> begin;
Variable_name: transaction_isolation	mysql>	mysql> update t set v = 'X' where k = 1;
Value: READ-COMMITTED	mysql>	Query OK, 1 row affected (0.00 sec)
1 row in set (0.02 sec)	mysql>	Rows matched: 1 Changed: 1 Warnings: 0
	mysql> select * from t;	
mysql> select * from t;	+---+------+	mysql>
+---+------+	| k | v |	mysql>
| k | v |	+---+------+	mysql>
+---+------+	| 1 | A |	mysql>
| 1 | A |	+---+------+	mysql>
+---+------+	1 row in set (0.00 sec)	mysql> commit;
1 row in set (0.00 sec)		Query OK, 0 rows affected (0.00 sec)
	mysql>	
mysql>	mysql> select * from t;	mysql>
	+---+------+	mysql>
	| k | v |	mysql>
	+---+------+	mysql>
	| 1 | X |	mysql>
	+---+------+	mysql>
	1 row in set (0.00 sec)	mysql>
		mysql>
	mysql> commit;	mysql>
	Query OK, 0 rows affected (0.00 sec)	mysql>

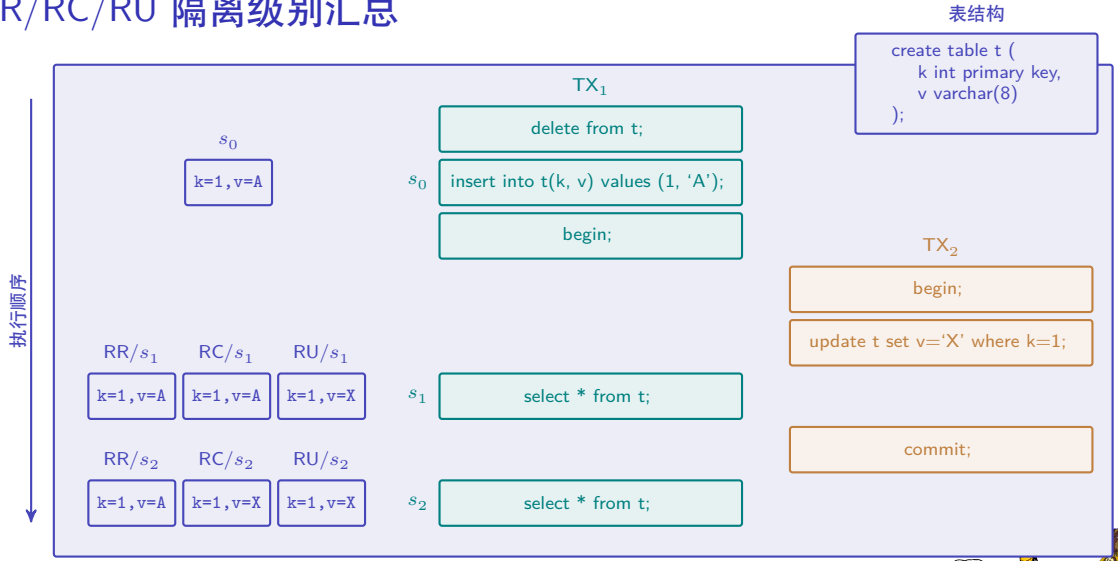
RU 读未提交

INIT	TX1	TX2
mysql> show variables like '%iso%'\G	mysql> begin;	mysql>
********** 1. row **********	mysql>	mysql> begin;
Variable_name: transaction_isolation	mysql>	mysql> update t set v = 'X' where k = 1;
Value: READ-UNCOMMITTED	mysql>	Query OK, 1 row affected (0.00 sec)
1 row in set (0.02 sec)	mysql>	Rows matched: 1 Changed: 1 Warnings: 0
	mysql> select * from t;	
mysql> select * from t;	+---+------+	mysql>
+---+------+	| k | v |	mysql>
| k | v |	+---+------+	mysql>
+---+------+	| 1 | X |	mysql>
| 1 | A |	+---+------+	mysql>
+---+------+	1 row in set (0.00 sec)	mysql> commit;
1 row in set (0.00 sec)		Query OK, 0 rows affected (0.00 sec)
	mysql>	
mysql>	mysql> select * from t;	mysql>
	+---+------+	mysql>
	| k | v |	mysql>
	+---+------+	mysql>
	| 1 | X |	mysql>
	+---+------+	mysql>
	1 row in set (0.00 sec)	mysql>
		mysql>
	mysql> commit;	mysql>
	Query OK, 0 rows affected (0.00 sec)	mysql>

S 串行化

```
-------------    INIT   --------------+--------------   TX1   --------------+---------------   TX2   -----------------
mysql> show variables like '%iso%'\G |mysql> begin;                        |mysql>
*************** 1. row ***************|mysql>                              |mysql> begin;
Variable_name: transaction_isolation |mysql>                              |mysql> update t set v = 'X' where k = 1;
        Value: SERIALIZABLE          |mysql>                              |Query OK, 1 row affected (0.00 sec)
1 row in set (0.02 sec)              |mysql>                              |Rows matched: 1  Changed: 1  Warnings: 0
                                     |mysql> select * from t;             |
mysql> select * from t;              |阻塞                                |mysql>
+---+------+                         |                                    |
| k | v    |                         |                                    |
+---+------+                         |                                    |
| 1 | A    |                         |                                    |
+---+------+                         |                                    |
1 row in set (0.00 sec)              |                                    |
                                     |                                    |
mysql>                               |                                    |
                                     |                                    |
                                     |                                    |
                                     |                                    |
                                     |                                    |
                                     |                                    |
                                     |                                    |
                                     |                                    |
                                     |                                    |
                                     |                                    |
-------------------------------------+------------------------------------+---------------
```

RR/RC/RU 隔离级别汇总



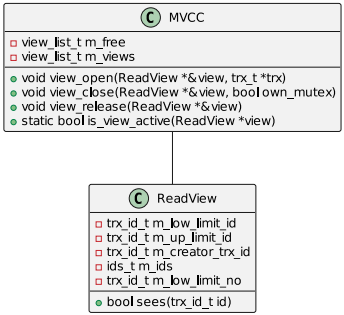
3

实现代码分析

ReadView

- ReadView 类重要成员变量定义 ☞ … /include/read0types.h

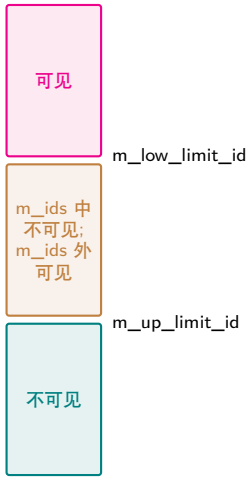
```
282  /** The read should not see any transaction with trx id >= this
283  value. In other words, this is the "high water mark". */
284  trx_id_t m_low_limit_id;
285
286  /** The read should see all trx ids which are strictly
287  smaller (<) than this value.  In other words, this is the
288  low water mark". */
289  trx_id_t m_up_limit_id;
290
291  /** trx id of creating transaction, set to TRX_ID_MAX for free
292  views. */
293  trx_id_t m_creator_trx_id;
294
295  /** Set of RW transactions that was active when this snapshot
296  was taken */
297  ids_t m_ids;
298
299  /** The view does not need to see the undo logs for transactions
300  whose transaction number is strictly smaller (<) than this value:
301  they can be removed in purge if not needed by other views */
302  trx_id_t m_low_limit_no;
```



RC 和 RR 下的 ReadView

- RC 和 RR 下生成 ReadView 的时机是有所差异的
  - RC 每次 SELECT 数据前都生成一个 ReadView
  - RR 只在第一次读取数据时生成一个 ReadView，后面会复用第一次生成的
  - RU 和 S 是两个特殊的隔离级别，它们不需要使用 ReadView
- 函数调用点

```
gdb) bt 3
#0  MVCC::view_open (this=0x7fffe4323800, view=@0x7fffe95f2098: 0x0, trx=0x7fffe95f1ff8)
    at /opt/src/mysql-server/storage/innobase/read/read0read.cc:530
#1  0x000055555aa1ac98 in trx_assign_read_view (trx=0x7fffe95f1ff8)
    at /opt/src/mysql-server/storage/innobase/trx/trx0trx.cc:2324
#2  0x000055555a952747 in row_search_mvcc (buf=0x7fff38b52b80 "\377", mode=PAGE_CUR_G,
        prebuilt=0x7fff380165a8, match_mode=0, direction=0)
    at /opt/src/mysql-server/storage/innobase/row/row0sel.cc:4822
(More stack frames follow...)
(gdb)
```



结束