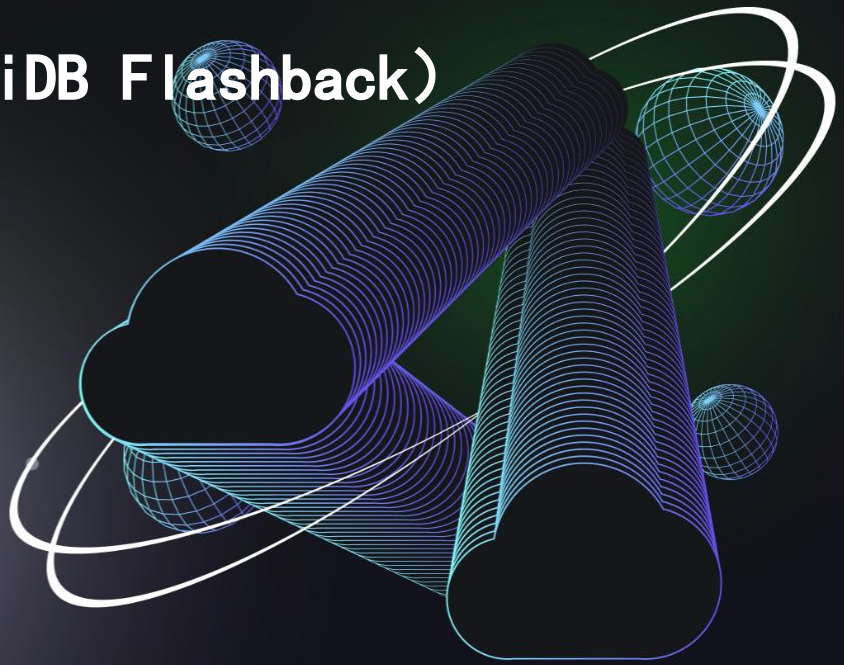
项目名称：MVCC 时光机（原 TiDB Flashback）

主讲人：耿海直



团队介绍

PingCAP TiDB

队长：何哲宇（@RinChanNOWWW）

队员：耿海直（@JmPotato）、黄梦龙（@disksing）

TiDB HACKATHON 2021

PingCAP TiDB

项目背景

TiDB HACKATHON 2021

理论上的灾难恢复 v. s. 实际上的灾难恢复

PingCAP TiDB

- 硬件故障
- 断网断电
- 洪水
- 海啸
- 地震
- 太阳黑子异常活动
- 核战争
- 彗星撞地球

- 出 bug 写坏数据
- 有漏洞被黑客攻击
- 把生产环境当成了测试环境
- drop 错数据库
- delete 忘加 where
- 删库跑路
- ~~宇宙射线导致比特位翻转~~

TiDB HACKATHON 2021

TiDB 的现状

“天灾易躲，人祸难防”，现有的功能和计划中的功能对 DML 造成的数据问题处理都太弱了：

- 对于需要排查数据损坏的情况，查看某条记录的变化历史太麻烦
- RECOVER TABLE 只能恢复 DROP/TRUNCATE 这种 DDL 操作，对 DML 没招
- GC SafePoint 之前的数据恢复不了
- 备份恢复需要经历 Dump 再写入的过程

PingCAP TiDB

项目解决了什么问题？

TiDB HACKATHON 2021

对于 MVCC 的思考

- 一段时间内的旧版本都在，只要数据没有被 GC，理论上可以进行快速恢复
- MVCC 数据是非常“结构化”的，完全有可能使用 SQL 的方式来进行读写操作
- MVCC 不只是可以用来暂时性地处理事务隔离，也完全可以做为冷备，相比于外部的备份，其优点是可以更省空间，恢复数据也更方便更快
- 如果需要回滚某段时间的写入，并不一定要真的删除或覆盖实际数据，可以使用轻量级“标记删除”的方式，跳过特定时间段的 MVCC 版本就行了

PingCAP TiDB

Make MVCC Great Again!

TiDB HACKATHON 2021

方案对比

	TiDB	TiDB + MVCC 时光机
排查数据变更历史	用 debug 接口查看 raw 数据，或者设置 snapshot_ts 查询某个时间点的数据	使用 SQL 查询所有版本，并可以进行各种条件过滤
误删/误写坏数据	recover table 只能处理 drop/truncate table	支持恢复各种 delete 或 update
回滚数据	只能先 dump 出来再重新覆盖写入，操作复杂，恢复速度慢	使用一条 flashback 命令，在亚秒级时间完成恢复（一次DDL同步）
长时间保留历史数据	更新频繁时占用大量磁盘空间，还影响性能	通过设置 save point，使用最少的空间保留多份 Point-in-Time 数据
外部数据备份历史数据	消耗额外资源同步数据，恢复数据耗时长	直接通过 save point 备份数据，充分利用 TiDB 的高可用等特性，而且恢复数据又快又方便

PingCAP TiDB

Hackathon 上实现了哪些内容？

TiDB HACKATHON 2021

Hackathon 上实现了哪些内容？

谁掌握了过去，谁就掌握了未来；谁掌握了现在，谁就掌握过去。

——乔治·奥威尔《1984》

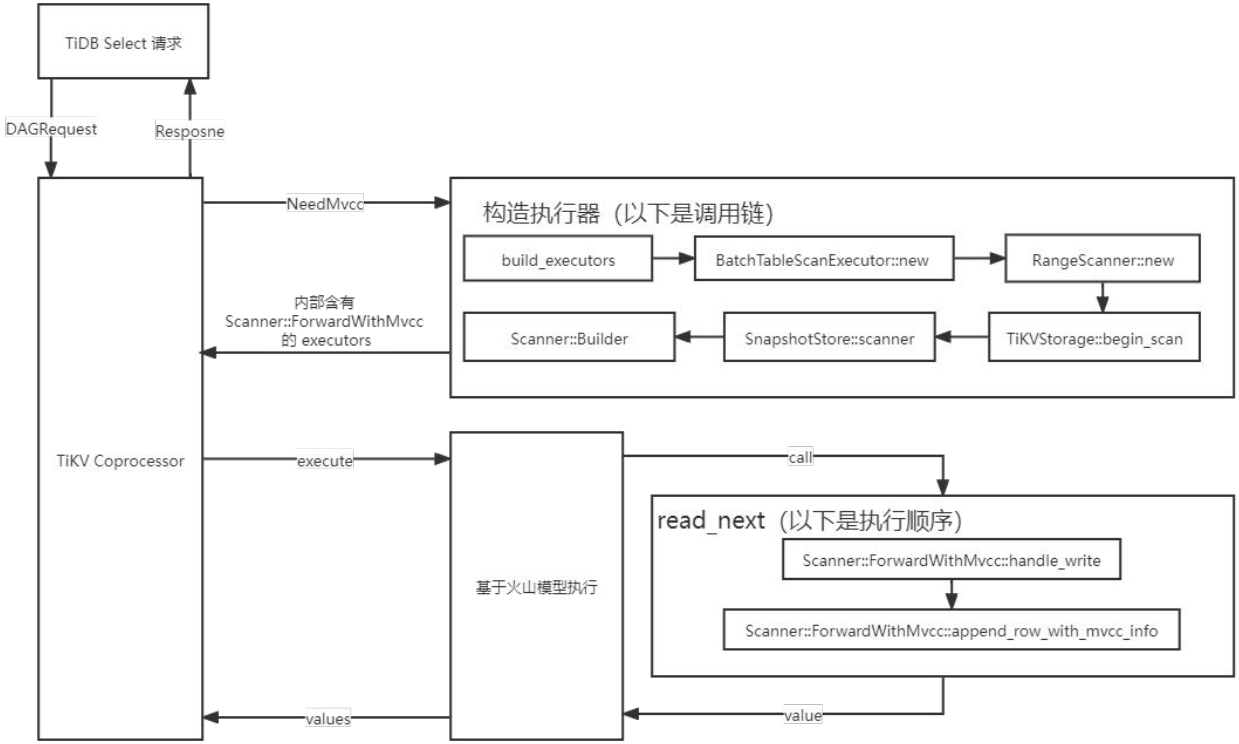
- 直接通过 SQL 增删改查数据的 MVCC 记录 -> 操纵过去
- 瞬间 Flashback 一张表到任意时刻 -> 掌控现在
- 保留表的指定版本，有备而无患 -> 着眼未来

MVCC Query in SQL

- 带上虚拟列（_tidb_mvcc_op _tidb_mvcc_ts）查询，可以 MVCC 版本
- 可以在查询条件中添加针对 的过滤条件，只显示符合条件
- 可以使用子查询的方式还原某
- 带上虚拟列的 UPDATE 操作直 MVCC 记录，不会新增 PUT 记

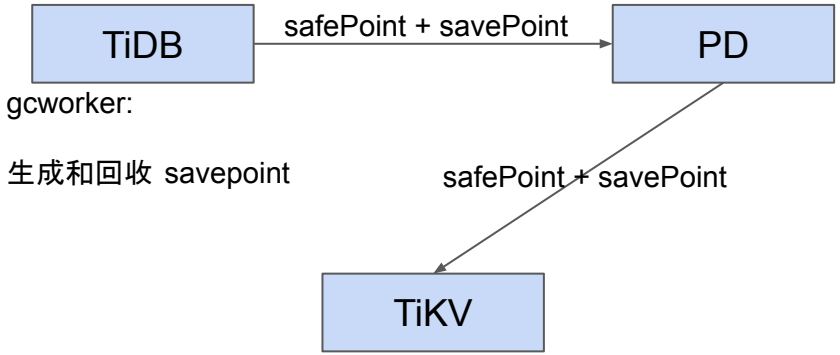
```
mysql> select * from t;
+------+
| a    |
+------+
|    2 |
|    3 |
+------+
2 rows in set (0.01 sec)

mysql> select _tidb_rowid, a, _tidb_mvcc_ts, _tidb_mvcc_op from t;
+-------------+------+--------------------+---------------+
| _tidb_rowid | a    | _tidb_mvcc_ts      | _tidb_mvcc_op |
+-------------+------+--------------------+---------------+
|           1 | NULL | 430325081203539970 | Delete        |
|           1 |    1 | 430325069158285315 | Put           |
|           2 |    2 | 430325069158285315 | Put           |
|           3 |    3 | 430325069158285315 | Put           |
+-------------+------+--------------------+---------------+
4 rows in set (0.01 sec)

mysql> select _tidb_rowid, a, _tidb_mvcc_ts, _tidb_mvcc_op from t where _tidb_mvcc_op = 'Delete';
+-------------+------+--------------------+---------------+
| _tidb_rowid | a    | _tidb_mvcc_ts      | _tidb_mvcc_op |
+-------------+------+--------------------+---------------+
|           1 | NULL | 430325081203539970 | Delete        |
+-------------+------+--------------------+---------------+
1 row in set (0.01 sec)

mysql> select _tidb_rowid, a, _tidb_mvcc_ts, _tidb_mvcc_op from t where _tidb_mvcc_ts < 430325081203539970;
+-------------+------+--------------------+---------------+
| _tidb_rowid | a    | _tidb_mvcc_ts      | _tidb_mvcc_op |
+-------------+------+--------------------+---------------+
|           1 |    1 | 430325069158285315 | Put           |
|           2 |    2 | 430325069158285315 | Put           |
|           3 |    3 | 430325069158285315 | Put           |
+-------------+------+--------------------+---------------+
3 rows in set (0.01 sec)
```

MVCC Query in SQL



GC SavePoint

- 新增新系统表 *gc_save_point*，并可以通过 SQL 进行管理
- 新增新 GC 配置项 gc_save_point_interval 和 gc_save_point_lifetime，用于系统自动创建 save point
- GC 之后，每个 SavePoint 前的最后一个版本数据会被保留
- 使用历史读或者 MVCC Query in SQL 查询时，可以展示 GC 按预期运行



Subsecond Flashback

- 添加 *FLASH TABLE TO TIMESTAMP TS* SQL 语句，用于指定表进行数据还原，意义是将表还原至不超过某个时间戳指定的版本
  - 快+轻量——并不是真的删除新数据
  - 表信息里加入了一个区间：[flashback_ts, now_ts]
  - DDL 操作进行更新
  - TiKV 扫描时跳过对应 TS 区间的 Key
  - 缺点 & 改进
- 还原后启动新事务进行查询，看到旧版本的数据

PingCAP TiDB

Demo

TiDB HACKATHON 2021

测试结果 - MVCC Query

PingCAP TiDB

```
mysql> select * from t;
+------+
| a    |
+------+
|    2 |
|    3 |
+------+
2 rows in set (0.01 sec)

mysql> select _tidb_rowid, a, _tidb_mvcc_ts, _tidb_mvcc_op from t;
+-------------+------+--------------------+---------------+
| _tidb_rowid | a    | _tidb_mvcc_ts      | _tidb_mvcc_op |
+-------------+------+--------------------+---------------+
|           1 | NULL | 430325081203539970 | Delete        |
|           1 |    1 | 430325069158285315 | Put           |
|           2 |    2 | 430325069158285315 | Put           |
|           3 |    3 | 430325069158285315 | Put           |
+-------------+------+--------------------+---------------+
4 rows in set (0.01 sec)

mysql> select _tidb_rowid, a, _tidb_mvcc_ts, _tidb_mvcc_op from t where _tidb_mvcc_op = 'Delete';
+-------------+------+--------------------+---------------+
| _tidb_rowid | a    | _tidb_mvcc_ts      | _tidb_mvcc_op |
+-------------+------+--------------------+---------------+
|           1 | NULL | 430325081203539970 | Delete        |
+-------------+------+--------------------+---------------+
1 row in set (0.01 sec)

mysql> select _tidb_rowid, a, _tidb_mvcc_ts, _tidb_mvcc_op from t where _tidb_mvcc_ts < 430325081203539970;
+-------------+------+--------------------+---------------+
| _tidb_rowid | a    | _tidb_mvcc_ts      | _tidb_mvcc_op |
+-------------+------+--------------------+---------------+
|           1 |    1 | 430325069158285315 | Put           |
|           2 |    2 | 430325069158285315 | Put           |
|           3 |    3 | 430325069158285315 | Put           |
+-------------+------+--------------------+---------------+
3 rows in set (0.01 sec)
```

Select

测试结果 - MVCC Query

```
mysql> select * from t;
+------+
| a    |
+------+
|    2 |
|    3 |
+------+
2 rows in set (0.00 sec)

mysql> update t set a = 4 where a = 2;
Query OK, 1 row affected (0.01 sec)
Rows matched: 1  Changed: 1  Warnings: 0

mysql> select * from t;
+------+
| a    |
+------+
|    4 |
|    3 |
+------+
2 rows in set (0.00 sec)

mysql> select _tidb_rowid, a, _tidb_mvcc_ts, _tidb_mvcc_op from t;
+-------------+------+--------------------+---------------+
| _tidb_rowid | a    | _tidb_mvcc_ts      | _tidb_mvcc_op |
+-------------+------+--------------------+---------------+
|           1 | NULL | 430325081203539970 | Delete        |
|           1 |    1 | 430325069158285315 | Put           |
|           2 |    4 | 430325117799104514 | Put           |
|           2 |    2 | 430325069158285315 | Put           |
|           3 |    3 | 430325069158285315 | Put           |
+-------------+------+--------------------+---------------+
5 rows in set (0.01 sec)
```

Update

TiDB HACKATHON 2021

测试结果 - MVCC Query

PingCAP TiDB

```
mysql> select * from t;
+------+
| a    |
+------+
|    4 |
|    3 |
+------+
2 rows in set (0.01 sec)

mysql> delete from t where a = 3;
Query OK, 1 row affected (0.02 sec)

mysql> select * from t;
+------+
| a    |
+------+
|    4 |
+------+
1 row in set (0.01 sec)

mysql> select _tidb_rowid, a, _tidb_mvcc_ts, _tidb_mvcc_op from t;
+-------------+------+--------------------+----------------+
| _tidb_rowid | a    | _tidb_mvcc_ts      | _tidb_mvcc_op  |
+-------------+------+--------------------+----------------+
|           1 | NULL | 430325081203539970 | Delete         |
|           1 |    1 | 430325069158285315 | Put            |
|           2 |    4 | 430325117799104514 | Put            |
|           2 |    2 | 430325069158285315 | Put            |
|           3 | NULL | 430325134458355715 | Delete         |
|           3 |    3 | 430325069158285315 | Put            |
+-------------+------+--------------------+----------------+
6 rows in set (0.00 sec)
```

Delete

TiDB HACKATHON 2021

测试结果 - MVCC Query

```
mysql> select * From  t;
+------+
| a    |
+------+
|    1 |
+------+
1 row in set (0.00 sec)

mysql> update t set a = 2 where _tidb_mvcc_ts = 430318915147268099;
Query OK, 0 rows affected (0.01 sec)
Rows matched: 0  Changed: 0  Warnings: 0

mysql> select _tidb_rowid, a, _tidb_mvcc_ts, _tidb_mvcc_op from t;
+-------------+------+--------------------+---------------+
| _tidb_rowid | a    | _tidb_mvcc_ts      | _tidb_mvcc_op |
+-------------+------+--------------------+---------------+
|           1 |    1 | 430319471658008579 | Put           |
+-------------+------+--------------------+---------------+
1 row in set (0.01 sec)

mysql> update t set a = 2 where _tidb_mvcc_ts = 430319471658008579;
Query OK, 1 row affected (0.02 sec)
Rows matched: 1  Changed: 1  Warnings: 0

mysql> select _tidb_rowid, a, _tidb_mvcc_ts, _tidb_mvcc_op from t;
+-------------+------+--------------------+---------------+
| _tidb_rowid | a    | _tidb_mvcc_ts      | _tidb_mvcc_op |
+-------------+------+--------------------+---------------+
|           1 |    2 | 430319471658008579 | Put           |
+-------------+------+--------------------+---------------+
1 row in set (0.01 sec)

mysql> select * from t;
+------+
| a    |
+------+
|    2 |
+------+
1 row in set (0.01 sec)
```

Update By MVCC

测试结果 - MVCC Query

```
mysql> select * from t;
+------+
| a    |
+------+
|    2 |
+------+
1 row in set (0.00 sec)

mysql> update t set a = 3 where a = 2;
Query OK, 1 row affected (0.07 sec)
Rows matched: 1  Changed: 1  Warnings: 0

mysql> select * from t;
+------+
| a    |
+------+
|    3 |
+------+
1 row in set (0.01 sec)

mysql> select _tidb_rowid, a, _tidb_mvcc_ts, _tidb_mvcc_op from t;
+-------------+------+--------------------+---------------+
| _tidb_rowid | a    | _tidb_mvcc_ts      | _tidb_mvcc_op |
+-------------+------+--------------------+---------------+
|           1 |    3 | 430320878286274562 | Put           |
|           1 |    2 | 430319471658008579 | Put           |
+-------------+------+--------------------+---------------+
2 rows in set (0.00 sec)

mysql> update t set a = 4 where _tidb_mvcc_ts = 430319471658008579;
Query OK, 1 row affected (0.08 sec)
Rows matched: 1  Changed: 1  Warnings: 0

mysql> select _tidb_rowid, a, _tidb_mvcc_ts, _tidb_mvcc_op from t;
+-------------+------+--------------------+---------------+
| _tidb_rowid | a    | _tidb_mvcc_ts      | _tidb_mvcc_op |
+-------------+------+--------------------+---------------+
|           1 |    3 | 430320878286274562 | Put           |
|           1 |    4 | 430319471658008579 | Put           |
+-------------+------+--------------------+---------------+
2 rows in set (0.00 sec)

mysql> select * from t;
+------+
| a    |
+------+
|    3 |
+------+
1 row in set (0.01 sec)
```

Update By MVCC

TiDB HACKATHON 2021

测试结果 - GC SavePoint

```
mysql> select VARIABLE_NAME,VARIABLE_VALUE from tidb where VARIABLE_NAME like '%sa
ve_point%';
+------------------------------+----------------+
| VARIABLE_NAME                | VARIABLE_VALUE |
+------------------------------+----------------+
| tikv_gc_save_point_interval  | 1h0m0s         |
| tikv_gc_save_point_life_time | 24h0m0s        |
+------------------------------+----------------+
2 rows in set (0.01 sec)

mysql> select * from gc_save_point;
+---------------------+
| save_point          |
+---------------------+
| 2022-01-08 06:00:00 |
| 2022-01-08 07:00:00 |
| 2022-01-08 08:00:00 |
| 2022-01-08 09:00:00 |
| 2022-01-08 10:00:00 |
| 2022-01-08 11:00:00 |
+---------------------+
6 rows in set (0.00 sec)
```

新增系统表 gc_save_point 记录 gc_safe_point 之外额外的多个存档点

新增 save_point_interval 和 save_point_life_time 配置用于系统定时自动创建和回收 save point

测试结果 - GC SavePoint

```
mysql> mysql> select _tidb_rowid, TIDB_PARSE_TSO(_tidb_mvcc_ts), _tidb_mvcc_op, a from t
by _tidb_mvcc_ts limit 20;
+-------------+------------------------------+---------------+---------+
| _tidb_rowid | TIDB_PARSE_TSO(_tidb_mvcc_ts) | _tidb_mvcc_op | a       |
+-------------+------------------------------+---------------+---------+
|           1 | 2022-01-08 11:09:54.941000    | Put           | hello1  |
|           1 | 2022-01-08 11:14:54.504000    | Put           | hello14 |
|           1 | 2022-01-08 11:19:54.055000    | Put           | hello27 |
|           1 | 2022-01-08 11:24:53.504000    | Put           | hello40 |
|           1 | 2022-01-08 11:29:53.005000    | Put           | hello53 |
|           1 | 2022-01-08 11:34:52.606000    | Put           | hello66 |
|           1 | 2022-01-08 11:36:47.856000    | Put           | hello71 |
|           1 | 2022-01-08 11:37:10.906000    | Put           | hello72 |
|           1 | 2022-01-08 11:37:33.907000    | Put           | hello73 |
|           1 | 2022-01-08 11:37:56.957000    | Put           | hello74 |
|           1 | 2022-01-08 11:38:20.006000    | Put           | hello75 |
|           1 | 2022-01-08 11:38:43.057000    | Put           | hello76 |
|           1 | 2022-01-08 11:39:06.106000    | Put           | hello77 |
|           1 | 2022-01-08 11:39:29.157000    | Put           | hello78 |
|           1 | 2022-01-08 11:39:52.157000    | Put           | hello79 |
|           1 | 2022-01-08 11:40:15.206000    | Put           | hello80 |
|           1 | 2022-01-08 11:40:38.256000    | Put           | hello81 |
|           1 | 2022-01-08 11:41:01.307000    | Put           | hello82 |
|           1 | 2022-01-08 11:41:24.357000    | Put           | hello83 |
|           1 | 2022-01-08 11:41:47.407000    | Put           | hello84 |
+-------------+------------------------------+---------------+---------+
20 rows in set (0.00 sec)
```

gc_safe_point

设置 gc_save_point_interval = ‘5m’ 后
在 gc_safe_point 之前，本来会被回收 mvcc 记录每 5 分钟保留一个版本

测试结果 - Subsecond Flashback

PingCAP TiDB

```
mysql> select * from dodo_list;
Empty set (0.00 sec)

mysql> select NOW();
+---------------------+
| NOW()               |
+---------------------+
| 2022-01-07 23:57:09 |
+---------------------+
1 row in set (0.00 sec)

mysql> insert into dodo_list values ("金色渡渡鸟");
Query OK, 1 row affected (0.00 sec)

mysql> select * from dodo_list;
+-----------------+
| dodo_name       |
+-----------------+
| 金色渡渡鸟      |
+-----------------+
1 row in set (0.00 sec)

mysql> flashback table dodo_list to timestamp '2022-01-07 23:57:09';
Query OK, 0 rows affected (0.06 sec)

mysql> select * from dodo_list;
Empty set (0.00 sec)
```

Flashback

测试结果 - Subsecond Flashback

```
mysql> select _tidb_rowid, dodo_name from dodo_list;
+-------------+--------------------------------+
| _tidb_rowid | dodo_name                      |
+-------------+--------------------------------+
|           1 | 元初渡渡鸟                     |
|           2 | 变化渡渡鸟                     |
|           3 | 不愿透露姓名的渡渡鸟           |
+-------------+--------------------------------+
3 rows in set (0.00 sec)

mysql> select _tidb_rowid, dodo_name, _tidb_mvcc_ts, _tidb_mvcc_op from dodo_list order by _tidb_mvcc_ts desc;
+-------------+--------------------------------+--------------------+---------------+
| _tidb_rowid | dodo_name                      | _tidb_mvcc_ts      | _tidb_mvcc_op |
+-------------+--------------------------------+--------------------+---------------+
|           2 | 变化渡渡鸟                     | 430328379040333826 | Put           |
|           3 | 不愿透露姓名的渡渡鸟           | 430328363639635970 | Put           |
|           2 | 时间渡渡鸟                     | 430328362577690626 | Put           |
|           1 | 元初渡渡鸟                     | 430328361215066115 | Put           |
+-------------+--------------------------------+--------------------+---------------+
4 rows in set (0.00 sec)

mysql> select TIDB_PARSE_TSO(430328379040333826);
+-----------------------------------+
| TIDB_PARSE_TSO(430328379040333826) |
+-----------------------------------+
| 2022-01-08 00:21:30.846000        |
+-----------------------------------+
1 row in set (0.00 sec)

mysql> flashback table dodo_list to timestamp '2022-01-08 00:21:29';
Query OK, 0 rows affected (0.09 sec)

mysql> select _tidb_rowid, dodo_name from dodo_list;
+-------------+--------------------------------+
| _tidb_rowid | dodo_name                      |
+-------------+--------------------------------+
|           1 | 元初渡渡鸟                     |
|           2 | 时间渡渡鸟                     |
|           3 | 不愿透露姓名的渡渡鸟           |
+-------------+--------------------------------+
3 rows in set (0.00 sec)
```

根据 MVCC 记录 Flashback

TiDB HACKATHON 2021

PingCAP TiDB

未来展望

TiDB HACKATHON 2021

未来展望

- 目前仅基于 TableScan 进行了 Demo，要想完全适配还有一些工作要做
- 与生态工具的兼容性问题
- Flashback 操作和 MVCC Query 的组合拳
  - 查看 Flashback 记录
  - 撤销 Flashback 操作
  - 修改 Flashback 记录

PingCAP TiDB

Thanks

TiDB HACKATHON 2021