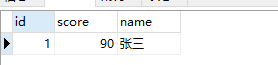
题目 01- 完成 ReadView 案例，解释为什么 RR 和 RC 隔离级别下看到查询结果不一致  
要求：

* 完成****案例 01- 读已提交 RC 隔离级别下的可见性分析****
* 完成****案例 02- 可重复读 RR 隔离级别下的可见性分析****
* 用通俗易懂的方式记录整个案例过程，可以画图与截图
* 做完案例给出结论，并对结论进行分析

回答范式：

准备一张表t,里面数据如图：



1. 案例 01- 读已提交 RC 隔离级别下的可见性分析
   * 目标

通过两个会话对同一数据行进行更新操作后查询，来判断结果是否符合RC的可见性判断。

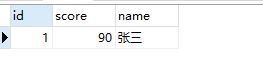
* + 操作步骤

|  |  |
| --- | --- |
| 会话1-txd=100 | 会话2-txd=110 |
| t1时刻 |  |
| begin;  select \* from t where id = 1;  update t set name = '李四' where id = 1; |  |
|  | t2时刻 |
|  | begin;  select \* from t where id = 1; |
| t3时刻 |  |
| commit; |  |
|  | t4时刻 |
|  | select \* from t where id = 1; |
|  | commit; |
|  |  |

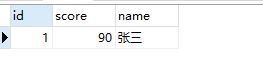
准备会话1，会话2，隔离级别均设置为RC

* + 实践过程

1.t1时会话1查询结果，并更新



2.t2时刻会话2查询结果



3.t4时刻会话2查询结果



* + 结论

在t1时刻，会话1更新’张三’为’李四’，但是事务没有提交。此时会话2在t2时刻查询时，生成的ReadView中，txd=100的会话1事务处于活动状态，对于会话2来说不可见，在undo日志中会追溯但最近可见版本’张三’。

在t3时刻，会话1提交更新后，t4时刻会话2查询时会生成新的ReadView,此时txd=100属于过去的事务，可以查询到更新，因此读取到’李四’。

1. 案例 02- 可重复读 RR 隔离级别下的可见性分析
   * 目标

通过两个会话对同一数据行进行更新操作后查询，来判断结果是否符合RR的可见性判断。

* + 操作步骤

准备会话1，会话2，隔离级别均设置为RC，后续操作步骤如案例1

* + 实践过程

1. t1时刻会话1查询



1. t2时刻会话2查询



1. t4时刻会话2查询



* + 结论

在t1时刻，会话1更新’张三’为’李四’，但是事务没有提交。此时会话2在t2时刻查询时，生成的ReadView中，txd=100的会话1事务处于活动状态，对于会话2来说不可见，在undo日志中会追溯但最近可见版本’张三’。

在t3时刻，会话1提交更新后，t4时刻会话2查询时仍然使用t2会话第一次查询的ReadView,txd=100的会话1事务处于活动状态，对于会话2来说不可见，在undo日志中会追溯但最近可见版本’张三’。

1. 结论分析

A:mysql中的innodb引擎对于rc与rr隔离级别的支持是基于undo日志版本链和readview的视图来实现的。rc与rr的主要区别是：rc每次查询都会生产一个新的readview,而rr只会在事务第一次读生成readview,后续都基于这个readview来判断数据可见性。

## **题目 02- 什么是索引？**

****要点：****

1. 优点是什么？

A:加快数据的检索效率，减少磁盘IO,提高数据库的查询速度。

1. 缺点是什么？

A:索引树会占用一定的磁盘空间；数据更新需要维护索引树。

1. 索引分类有哪些？特点是什么？

A:

根据数量分：

单列索引：单个列上建立的索引。

组合索引：在两个及以上的列上建立的索引。

根据是否主键分：

聚集索引：主键上建立的索引。

辅助索引：在非单个主键上建立的索引。

1. 索引创建的原则是什么？

频繁出现在where，order by ,group by上的字段建立组合索引；

频繁出现在select后的字段建立组合索引；

多表join，on条件上的字段建立索引；

1. 有哪些使用索引的注意事项？

表记录少不需要建立索引；

一个表的索引不要太多；

尽量建立组合索引；

频繁更新的字段不建议建立索引；

区分度低的字段不建议建立索引；

很长的字段不建议建立索引；

无序的字段不建议建立索引；

1. 如何知道 SQL 是否用到了索引？

可以通过explain分析sql语句的执行计划，看是否使用了索引。

1. 请你解释一下索引的原理是什么？「重点」
   * 说清楚为什么要用 B+Tree

A：索引的主要目的是通过空间换时间的理念，尽可能减少磁盘io，提高数据的检索效率。

索引的数据结构一般需要满足两个基本的需求：等值查询与范围查询；

使用b+树建立索引，查询的时间效率是logn,而且因为树的层次不高，磁盘io的次数会减少。

B+树的叶子结点形成的有序双向链表结构，也能极大提高范围查询的效率。并且每个表的数据 都在主键索引的叶子结点上，减少了磁盘空间的消耗。

## **题目 03- 什么是 MVCC？**

****要点：****

1. Redo 日志

A:redo日志是mysql的重做日志，能够降低磁盘IO,提供高性能的并发能力。在数据库崩溃时也能用于数据恢复，保证数据的完整性和一致性。

1. ReadView

A:ReadView是Innodb引擎中用来实现RC和RR事务隔离级别的基础。总的来说，就是在查询的时候，会生成对当前活动的事务Id、当前活动事务Id的最小值、最大值、生成本ReadView的事务id的视图。

1. 如何判断可见性

A:mysql中的RC和RR的可见性是通过Undo日志与ReadView来实现的。

总体上讲：

如果undo版本链中的记录的事务Id<ReadView中活动事务Id的最小值，表明这是过去已经提交的事务，可见。

如果undo版本链中的记录的事务Id>ReadView中活动事务Id的最大值，表明这是未来的事务，不可见。

如果undo版本链中的记录的事务Id=ReadView中本事务Id，则可见。

如果undo版本链中的记录的事务Id在ReadView活动事务Id的最小值与最大值之间，则判断是否是当前活动的事务Id，是则不可见；不是则可见。