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：

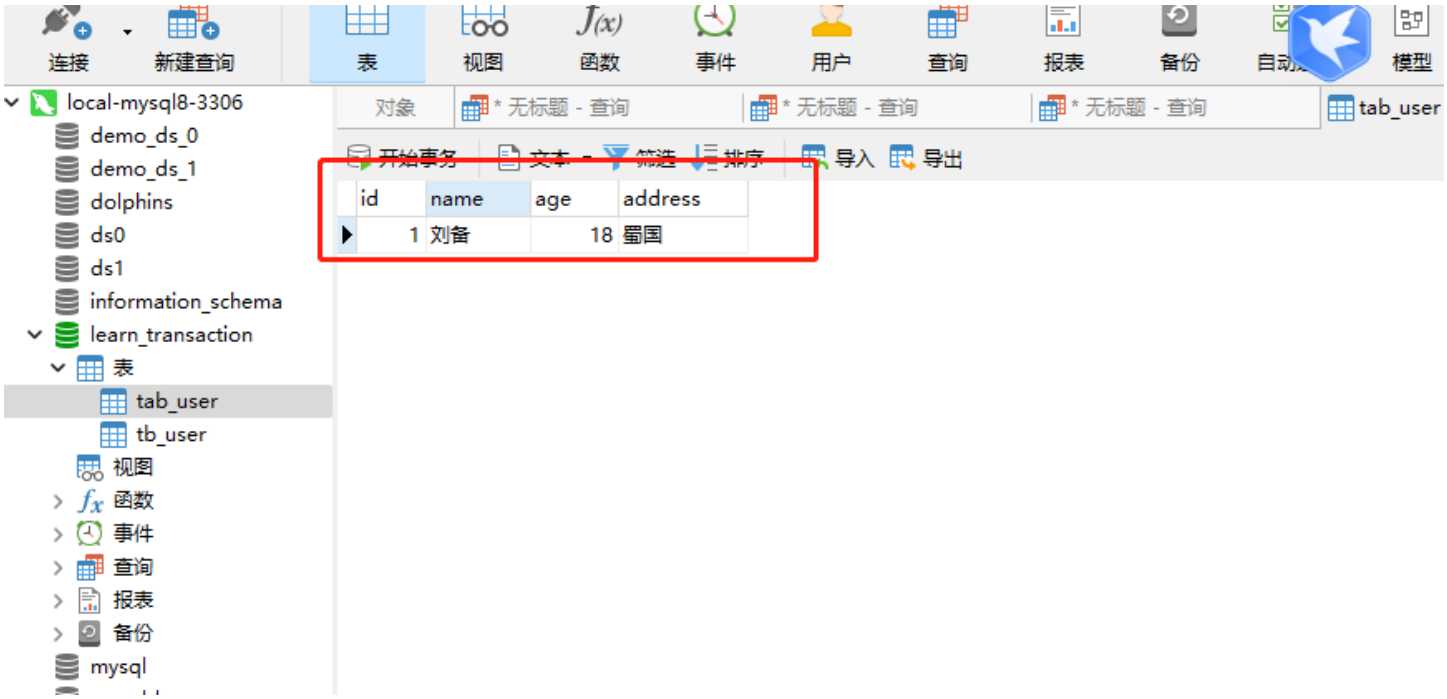
1. 案例 01- 读已提交 RC 隔离级别下的可见性分析

    - 目标

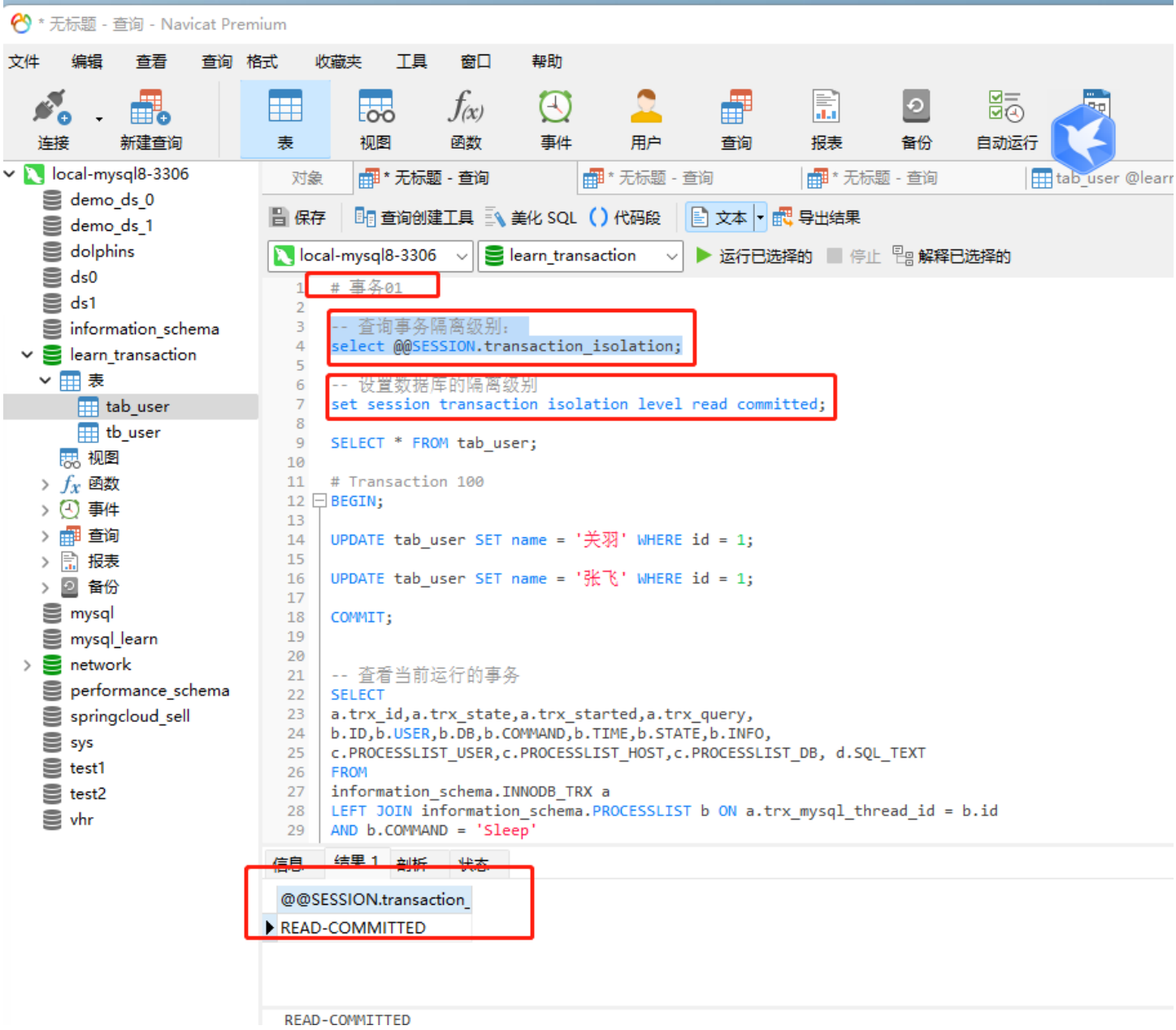
      通过实验证明RC隔离级别下，事务间数据的可见性

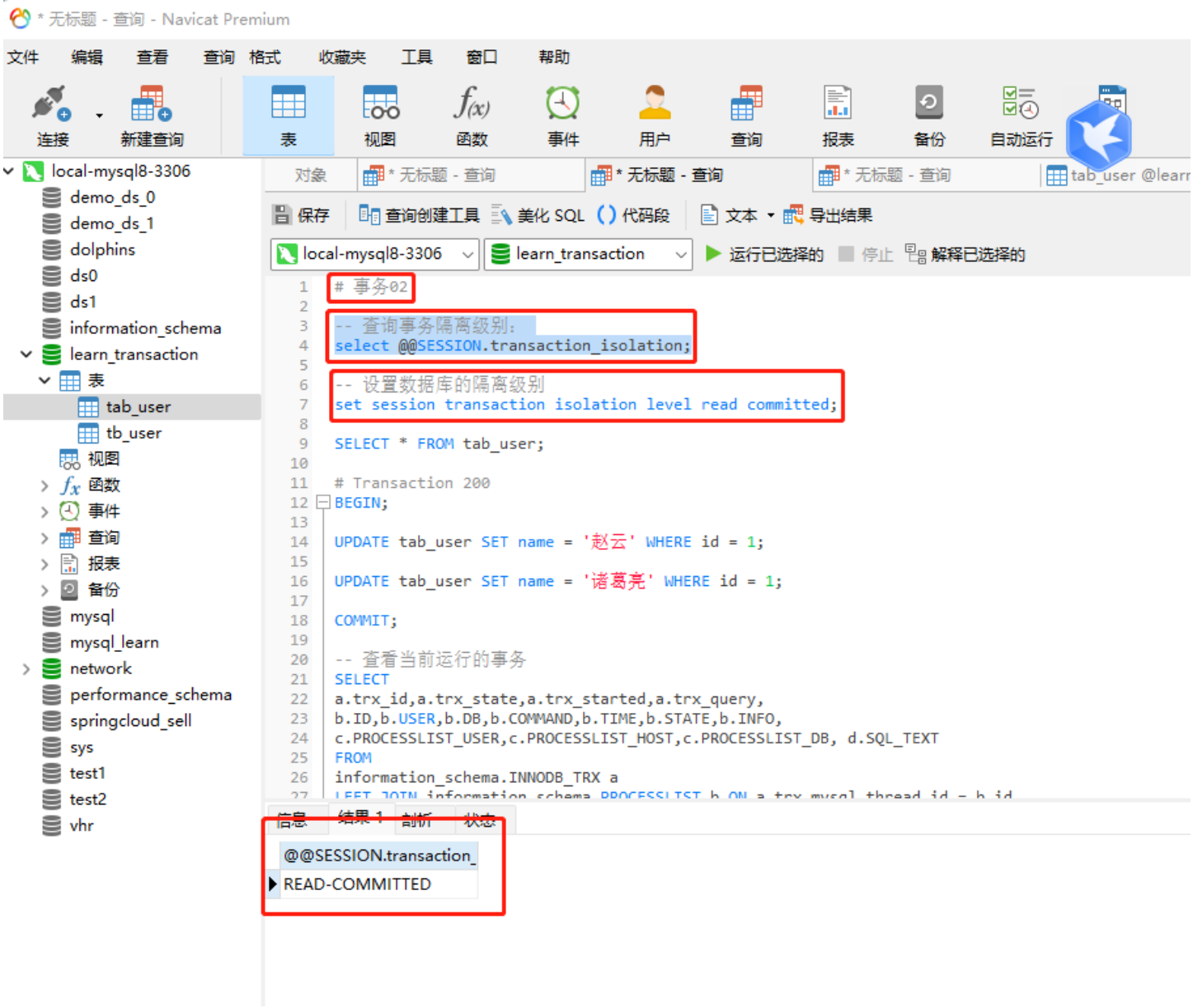
    - 操作步骤

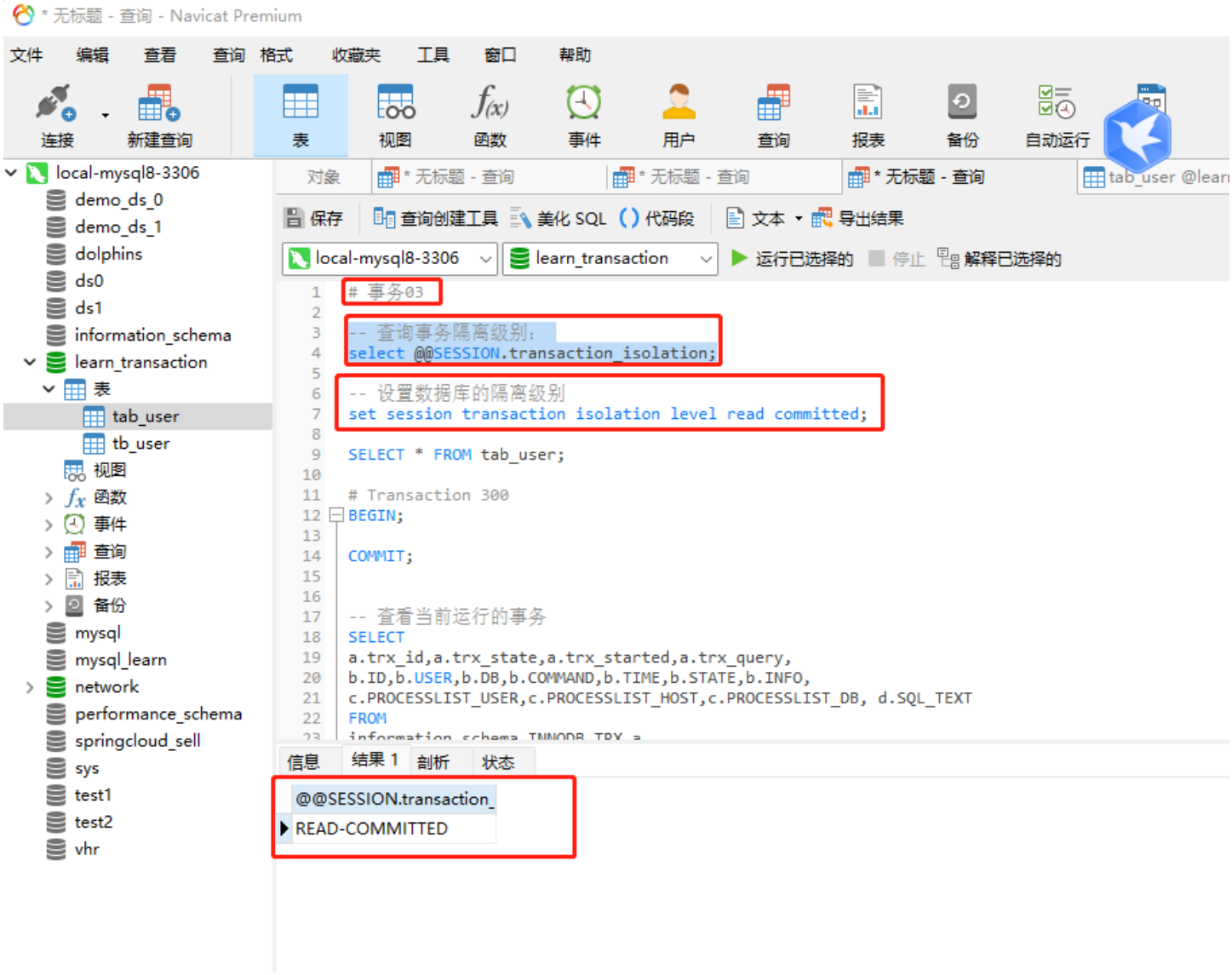
      1.创建用户表tab_user

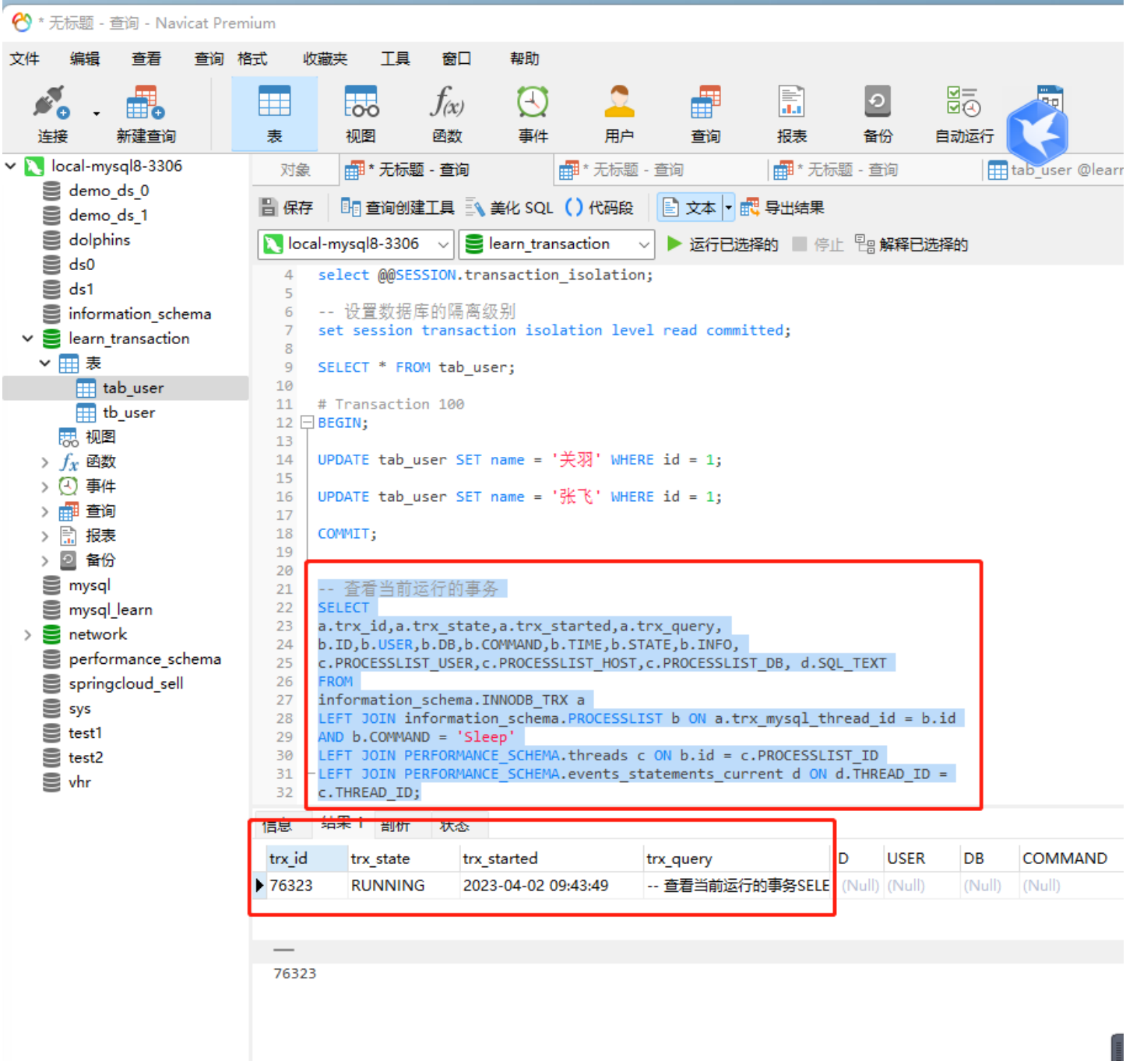
    - 2.通过navicat了解数据库，并打开三个查询窗口，都设置事务隔离级别为RC,并都开启事务。


* 无标题 - 查询 - Navicat Premium
文件 编辑 查看 查询 格式 收藏夹 工具 窗口 帮助
连接 新建查询 表 视图 函数 事件 用户 查询 报表 备份 自动运行
local-mysql8-3306
demo_ds_0
demo_ds_1
dolphins
ds0
ds1
information_schema
learn_transaction
表
tab_user
tb_user
视图
函数
事件
查询
报表
备份
mysql
mysql_learn
network
performance_schema
springcloud_sell
sys
test1
test2
vhr
对象
* 无标题 - 查询
* 无标题 - 查询
* 无标题 - 查询
tab_user @learn
保存 查询创建工具 美化 SQL 代码段 文本 导出结果
local-mysql8-3306 learn_transaction 运行已选择的 停止 解释已选择的
事务01
-- 查询事务隔离级别:
select @@SESSION.transaction_isolation;
-- 设置数据库的隔离级别
set session transaction isolation level read committed;
SELECT * FROM tab_user;
# Transaction 100
BEGIN;
UPDATE tab_user SET name = '关羽' WHERE id = 1;
UPDATE tab_user SET name = '张飞' WHERE id = 1;
COMMIT;
-- 查看当前运行的事务
SELECT
a.trx_id,a.trx_state,a.trx_started,a.trx_query,
b.ID,b.USER,b.DB,b.COMMAND,b.TIME,b.STATE,b.INFO,
c.PROCESSLIST_USER,c.PROCESSLIST_HOST,c.PROCESSLIST_DB, d.SQL_TEXT
FROM
information_schema.INNODB_TRX a
LEFT JOIN information_schema.PROCESSLIST b ON a.trx_mysql_thread_id = b.id
AND b.COMMAND = 'Sleep'
信息 结果 1 剖析 状态
@@SESSION.transaction_
READ-COMMITTED
READ-COMMITTED

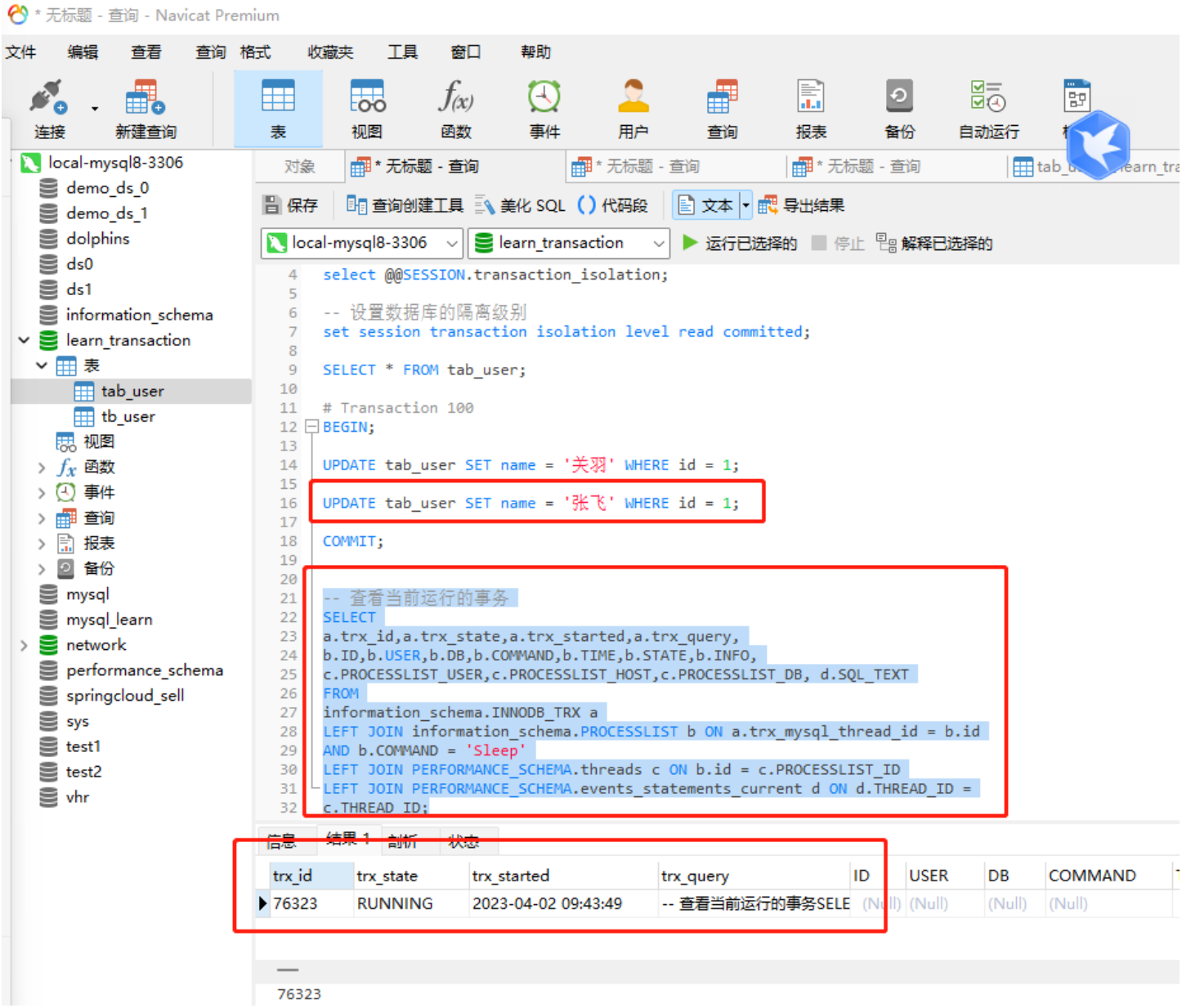

* 无标题 - 查询 - Navicat Premium
文件 编辑 查看 查询 格式 收藏夹 工具 窗口 帮助
连接 新建查询 表 视图 函数 事件 用户 查询 报表 备份 自动运行
local-mysql8-3306
demo_ds_0
demo_ds_1
dolphins
ds0
ds1
information_schema
learn_transaction
表
tab_user
tb_user
视图
函数
事件
查询
报表
备份
mysql
mysql_learn
network
performance_schema
springcloud_sell
sys
test1
test2
vhr
对象
* 无标题 - 查询
* 无标题 - 查询
* 无标题 - 查询
tab_user @learn
保存 查询创建工具 美化 SQL 代码段 文本 导出结果
local-mysql8-3306
learn_transaction
运行已选择的 停止 解释已选择的
1 # 事务02
2
3 -- 查询事务隔离级别:
4 select @@SESSION.transaction_isolation;
5
6 -- 设置数据库的隔离级别
7 set session transaction isolation level read committed;
8
9 SELECT * FROM tab_user;
10
11 # Transaction 200
12 BEGIN;
13
14 UPDATE tab_user SET name = '赵云' WHERE id = 1;
15
16 UPDATE tab_user SET name = '诸葛亮' WHERE id = 1;
17
18 COMMIT;
19
20 -- 查看当前运行的事务
21 SELECT
22 a.trx_id,a.trx_state,a.trx_started,a.trx_query,
23 b.ID,b.USER,b.DB,b.COMMAND,b.TIME,b.STATE,b.INFO,
24 c.PROCESSLIST_USER,c.PROCESSLIST_HOST,c.PROCESSLIST_DB, d.SQL_TEXT
25 FROM
26 information_schema.INNODB_TRX a
信息 结果 1 剖析 状态
@@SESSION.transaction_
READ-COMMITTED



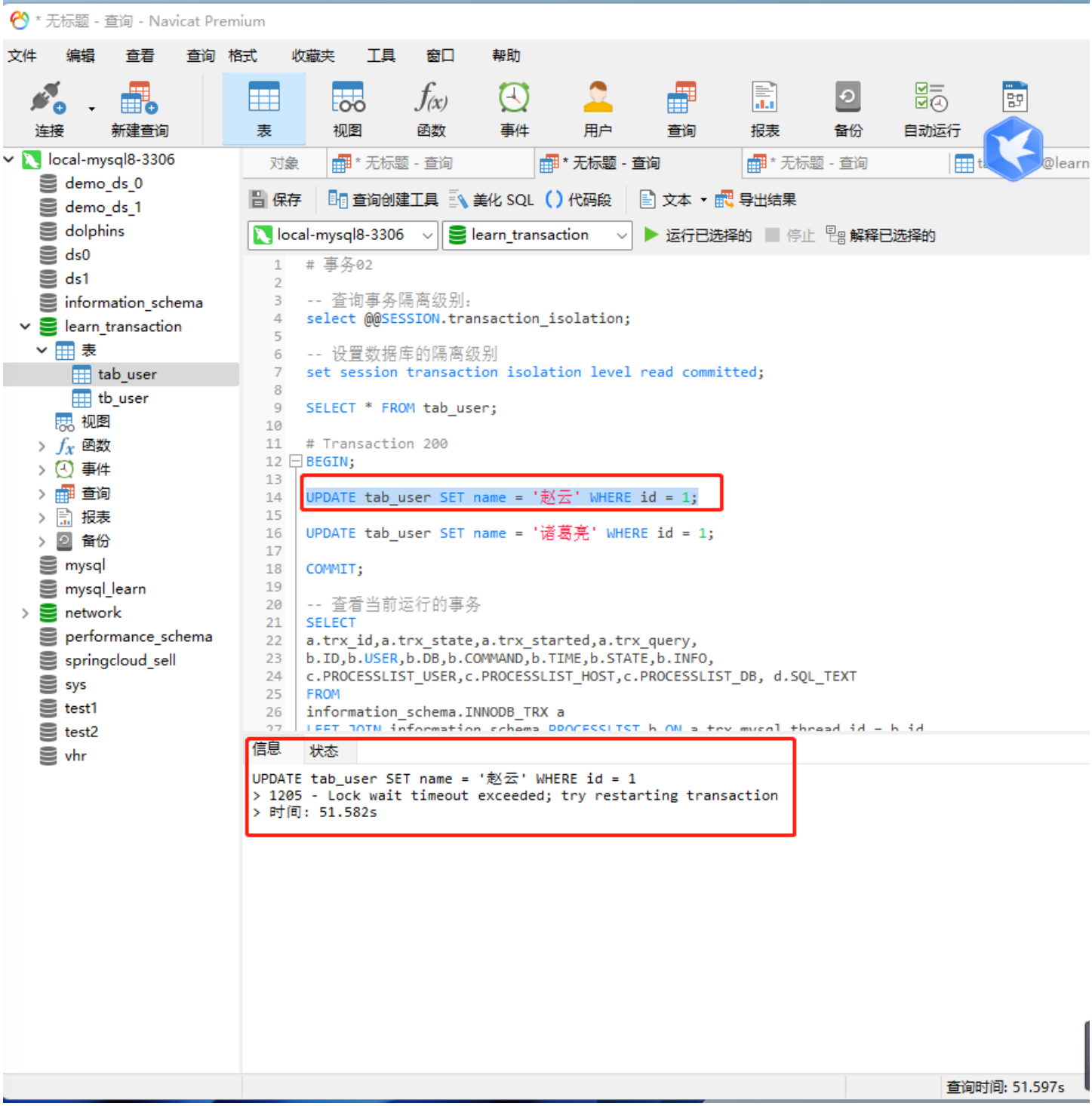
3.更新事务1数据，并查询事务1的信息



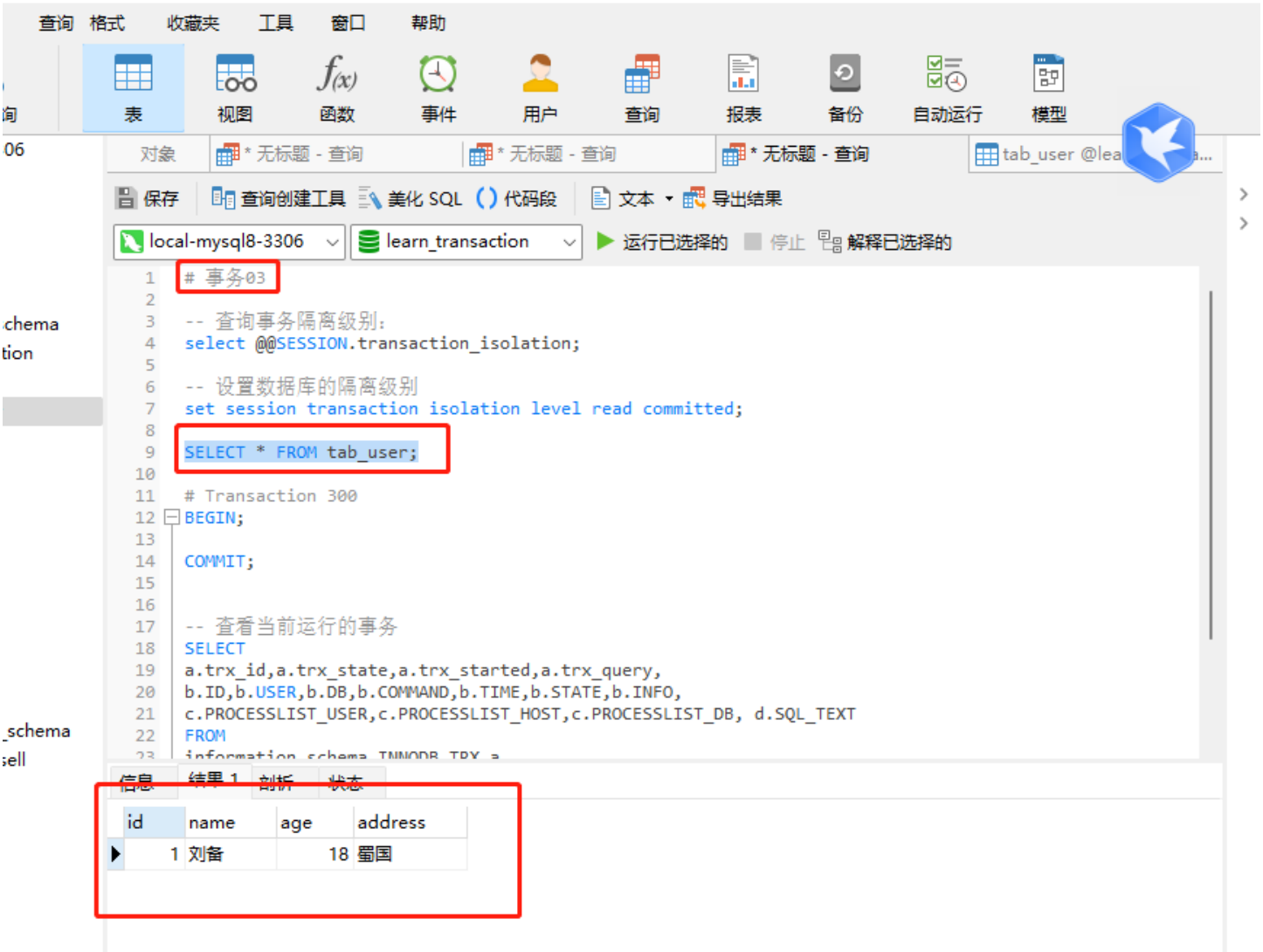
4.继续更新事务1数据，并查询事务1的详情



5.更新事务2的数据，发现数据被锁，事务等待中

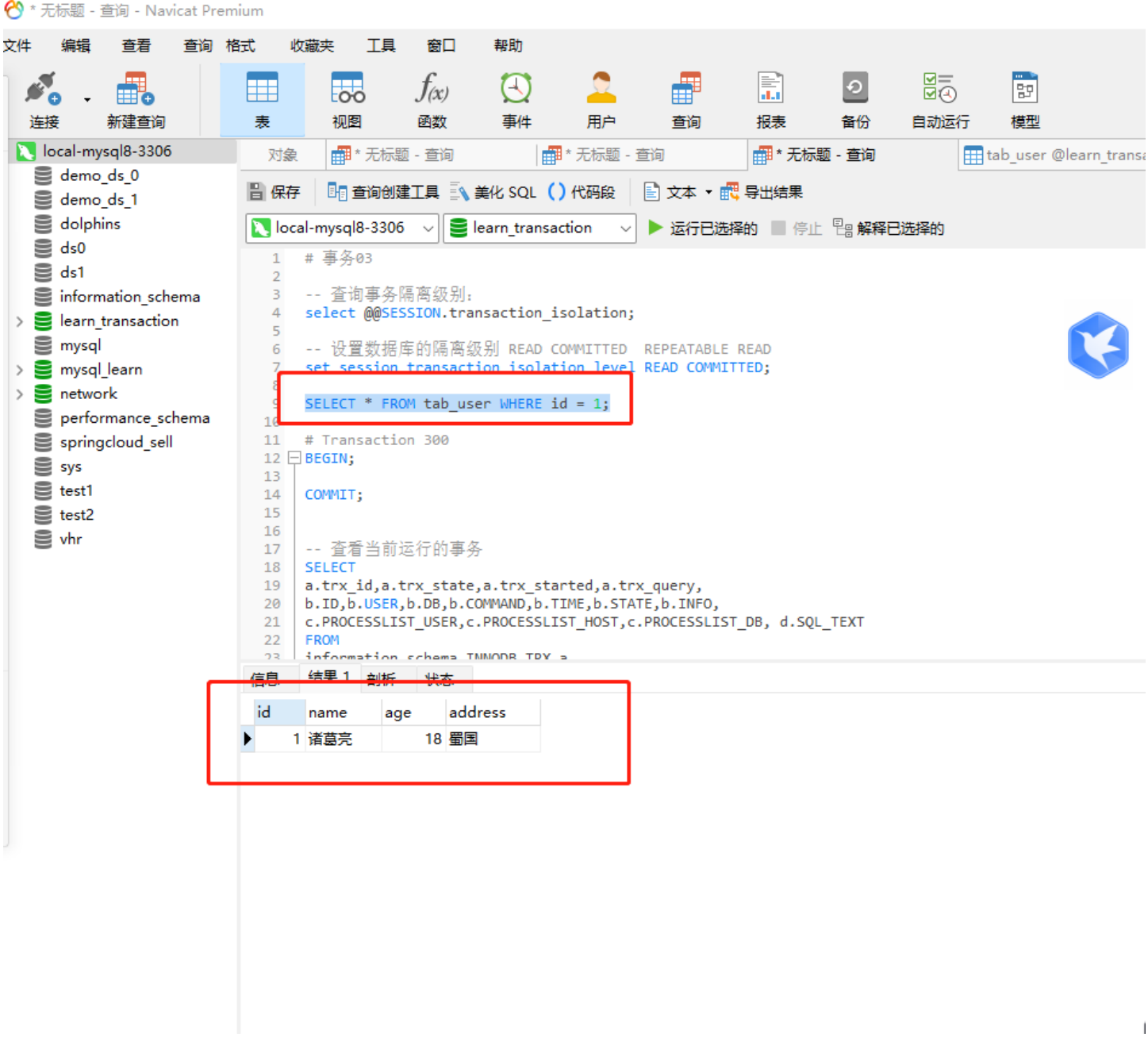


6.在事务三中查询数据，发现数据未变，查询出刘备。



8.提交事务1，然后通过事务3查询数据，得到数据是张飞，也是就事7.提交事务1后，查询事务3的数据是张飞，也就是事务1最后更改的数据。同时释放锁，事务2更新成功

- 8.继续更新事务2的数据，更新完后，提交事务2。然后查询事务3数据，得到的是诸葛亮



9.提交事务2，通过事务三查询数据，得到数据是张飞。

10.最终提交事务3.结束实践

- 结论

读已提交的事务隔离级别，会造成不可重复读的问题，脏读、幻读。

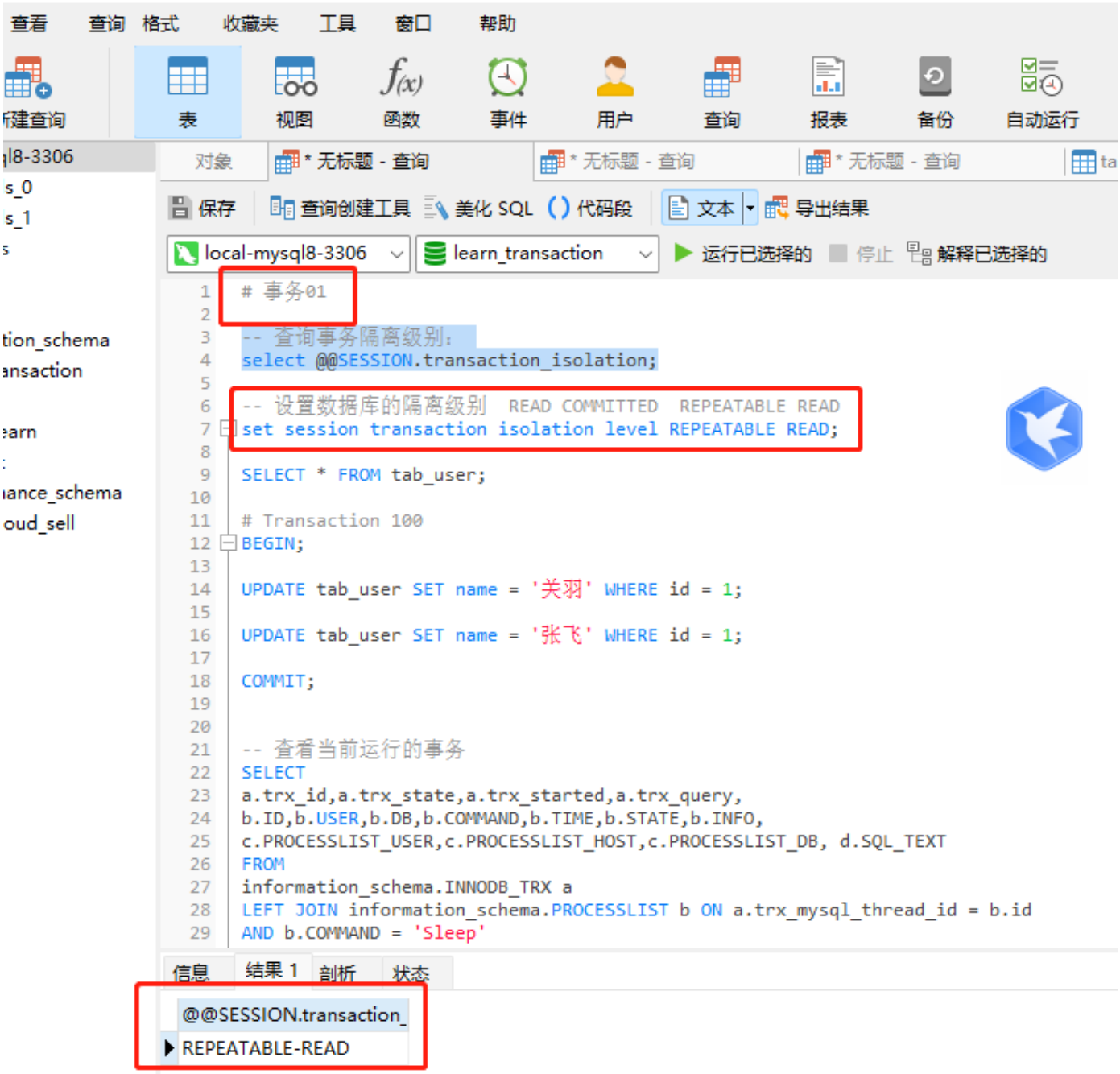
2. 案例 02- 可重复读 RR 隔离级别下的可见性分析

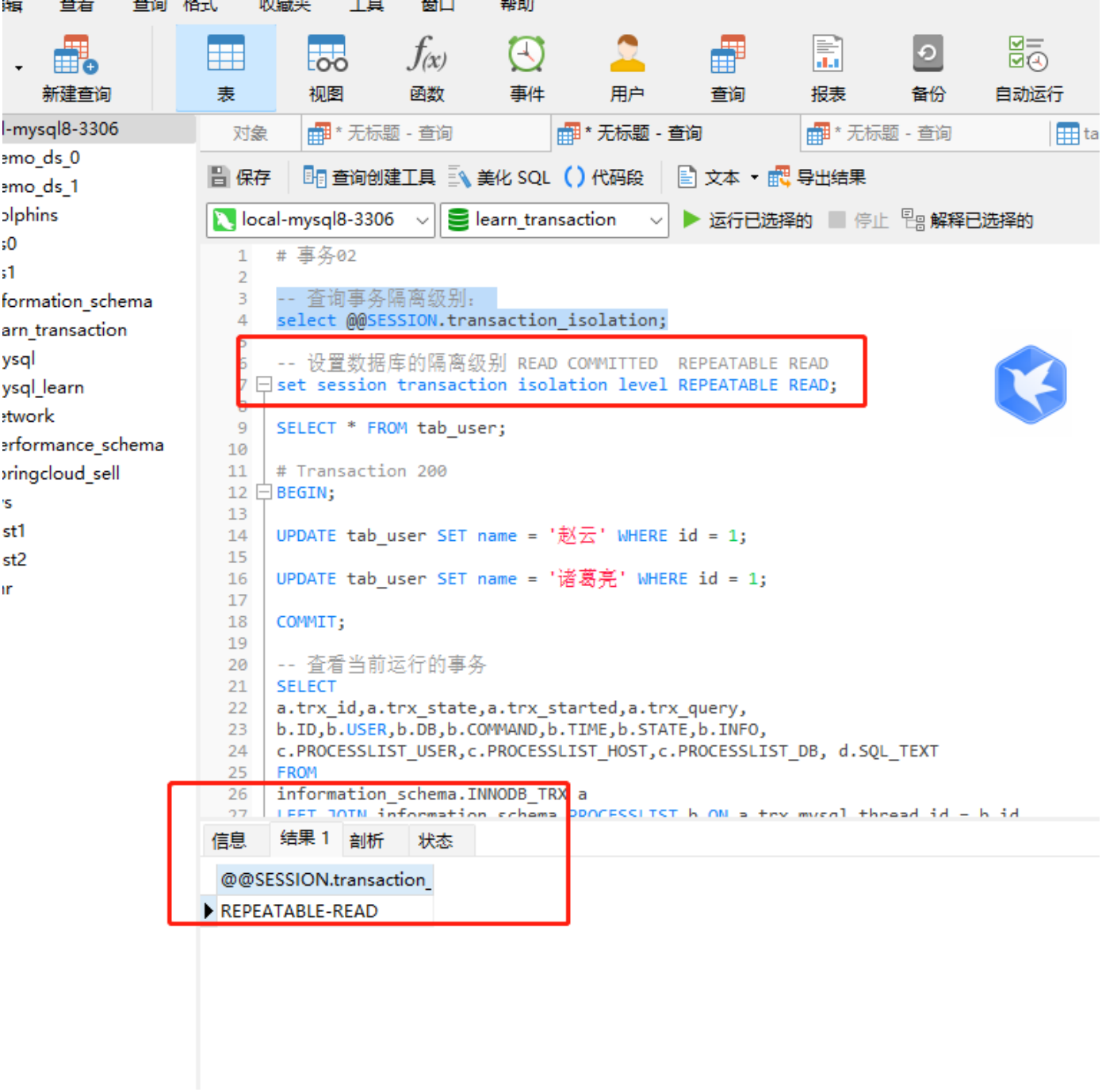
- 目标

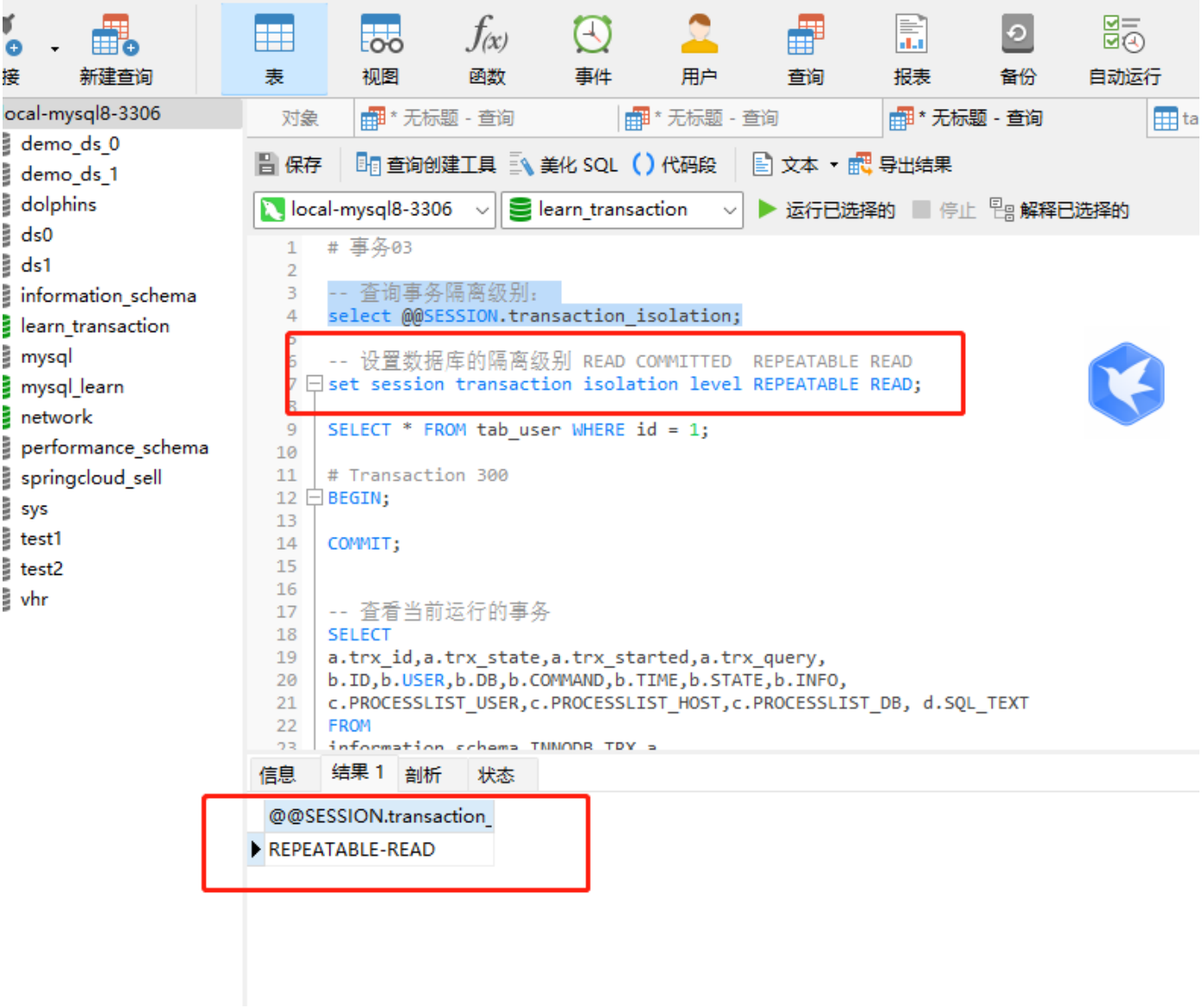
通过实验证明RR隔离级别下，事务间数据的可见性

  - 操作步骤

1.开启三个事务，并分别设置事务隔离级别为RR

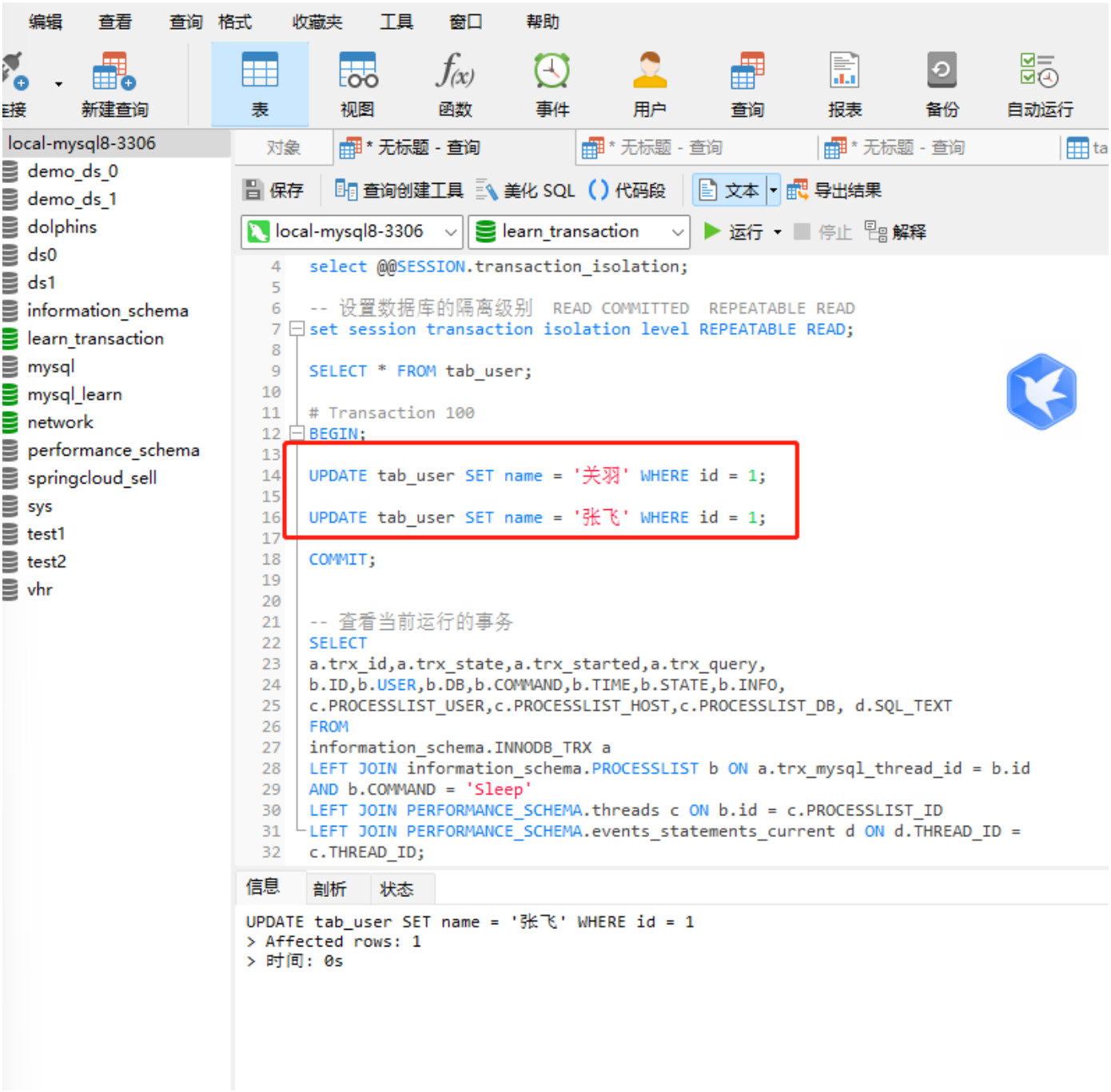

查看 查询 格式 收藏夹 工具 窗口 帮助
新建查询 表 视图 函数 事件 用户 查询 报表 备份 自动运行
对象 * 无标题 - 查询 * 无标题 - 查询 * 无标题 - 查询
保存 查询创建工具 美化 SQL 代码段 文本 导出结果
local-mysql8-3306 learn_transaction 运行已选择的 停止 解释已选择的
事务01
-- 查询事务隔离级别:
select @@SESSION.transaction_isolation;
-- 设置数据库的隔离级别 READ COMMITTED REPEATABLE READ
set session transaction isolation level REPEATABLE READ;
SELECT * FROM tab_user;
# Transaction 100
BEGIN;
UPDATE tab_user SET name = '关羽' WHERE id = 1;
UPDATE tab_user SET name = '张飞' WHERE id = 1;
COMMIT;
-- 查看当前运行的事务
SELECT
a.trx_id,a.trx_state,a.trx_started,a.trx_query,
b.ID,b.USER,b.DB,b.COMMAND,b.TIME,b.STATE,b.INFO,
c.PROCESSLIST_USER,c.PROCESSLIST_HOST,c.PROCESSLIST_DB, d.SQL_TEXT
FROM
information_schema.INNODB_TRX a
LEFT JOIN information_schema.PROCESSLIST b ON a.trx_mysql_thread_id = b.id
AND b.COMMAND = 'Sleep'
信息 结果 1 剖析 状态
@@SESSION.transaction_
REPEATABLE-READ


新建查询
表
视图
函数
事件
用户
查询
报表
备份
自动运行
-mysql8-3306
emo_ds_0
emo_ds_1
olphins
;0
;1
formation_schema
arn_transaction
ysql
ysql_learn
etwork
erformance_schema
pringcloud_sell
's
st1
st2
ir
对象
* 无标题 - 查询
* 无标题 - 查询
* 无标题 - 查询
保存
查询创建工具
美化 SQL
代码段
文本
导出结果
local-mysql8-3306
learn_transaction
运行已选择的
停止
解释已选择的
1 # 事务02
2
3 -- 查询事务隔离级别：
4 select @@SESSION.transaction_isolation;
5
6 -- 设置数据库的隔离级别 READ COMMITTED REPEATABLE READ
7 set session transaction isolation level REPEATABLE READ;
8
9 SELECT * FROM tab_user;
10
11 # Transaction 200
12 BEGIN;
13
14 UPDATE tab_user SET name = '赵云' WHERE id = 1;
15
16 UPDATE tab_user SET name = '诸葛亮' WHERE id = 1;
17
18 COMMIT;
19
20 -- 查看当前运行的事务
21 SELECT
22 a.trx_id,a.trx_state,a.trx_started,a.trx_query,
23 b.ID,b.USER,b.DB,b.COMMAND,b.TIME,b.STATE,b.INFO,
24 c.PROCESSLIST_USER,c.PROCESSLIST_HOST,c.PROCESSLIST_DB, d.SQL_TEXT
25 FROM
26 information_schema.INNODB_TRX a
信息
结果 1
剖析
状态
@@SESSION.transaction_
REPEATABLE-READ

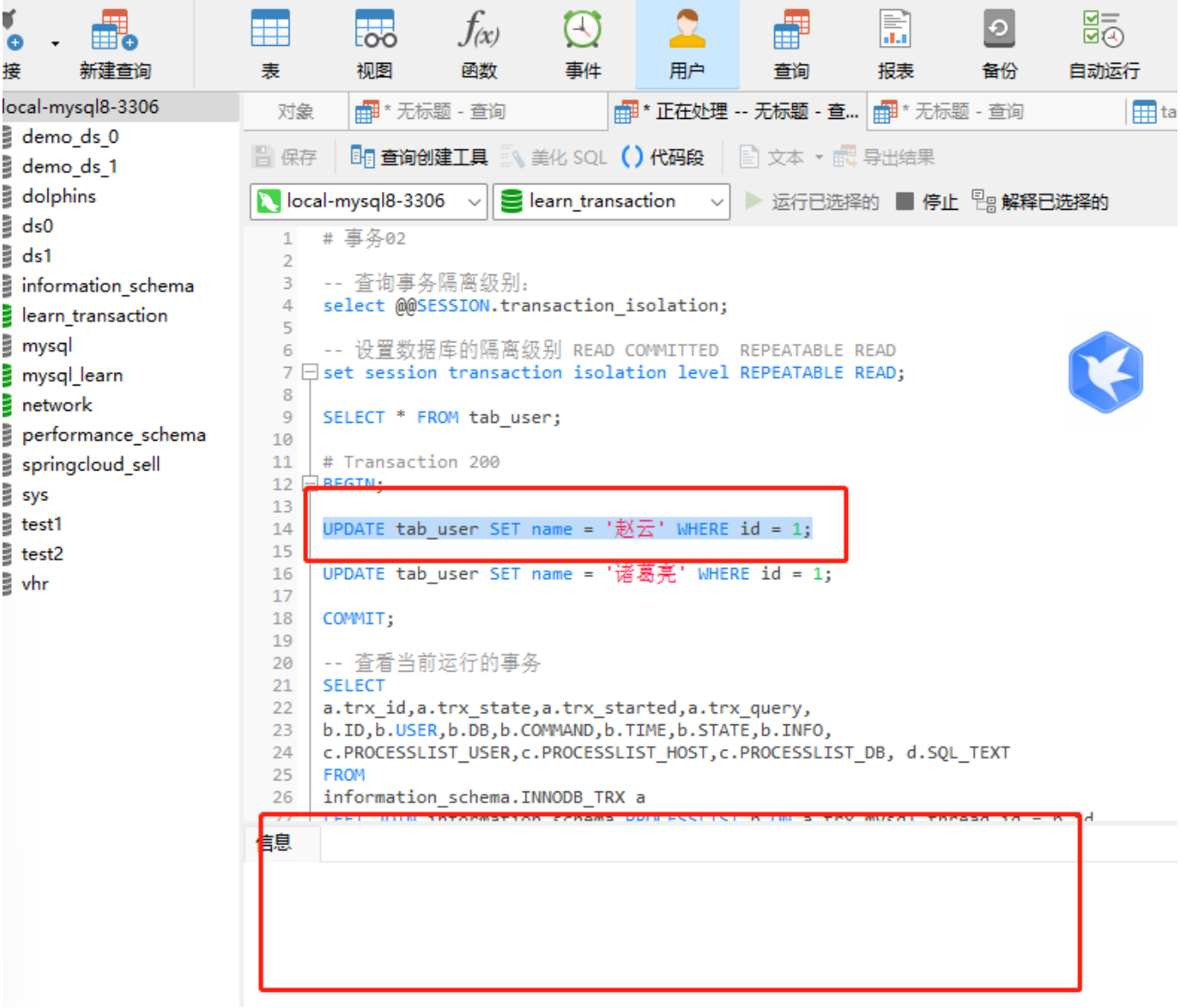


2.将三个事务都开启

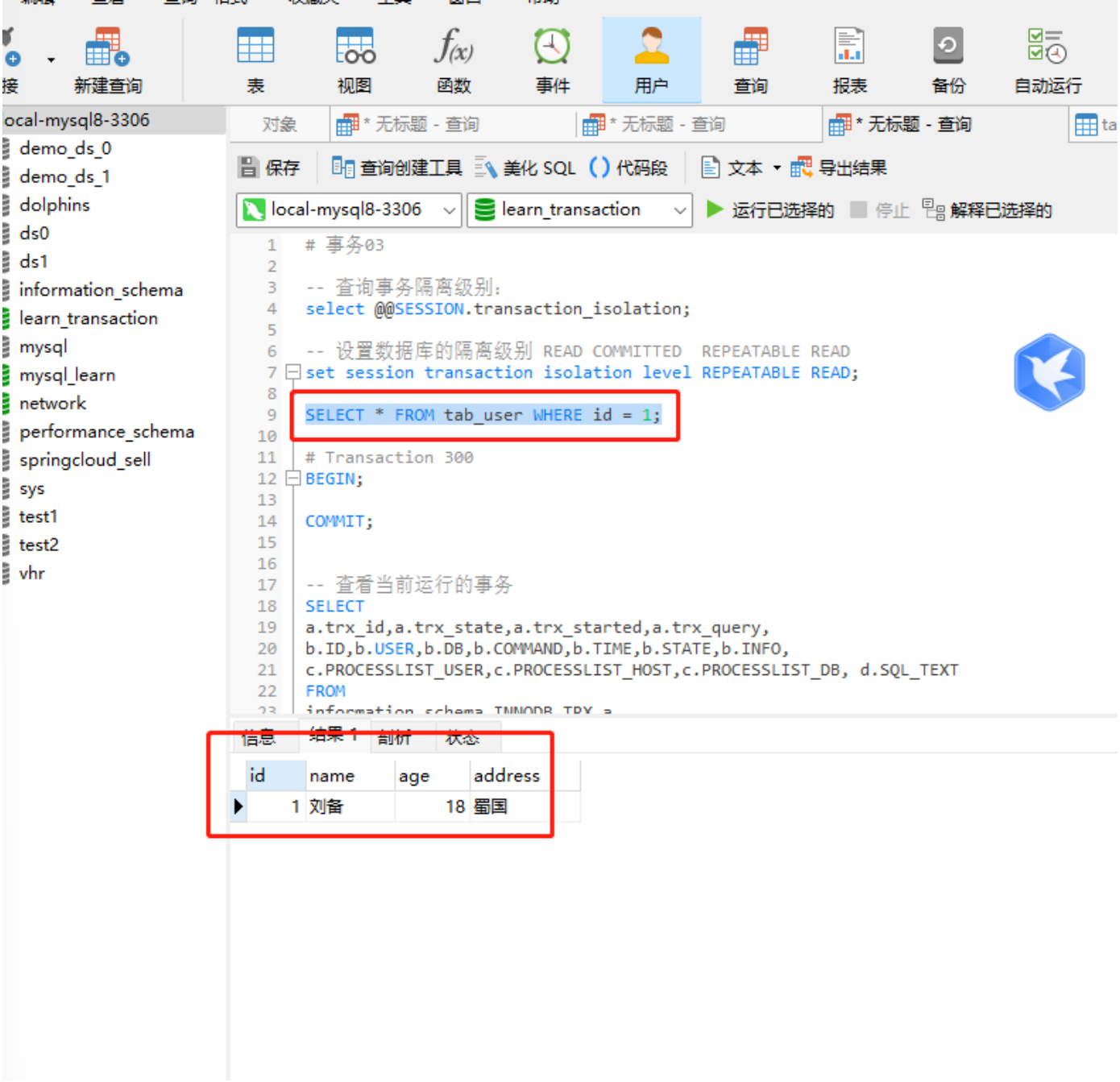
3.将事务1的数据进行更新。



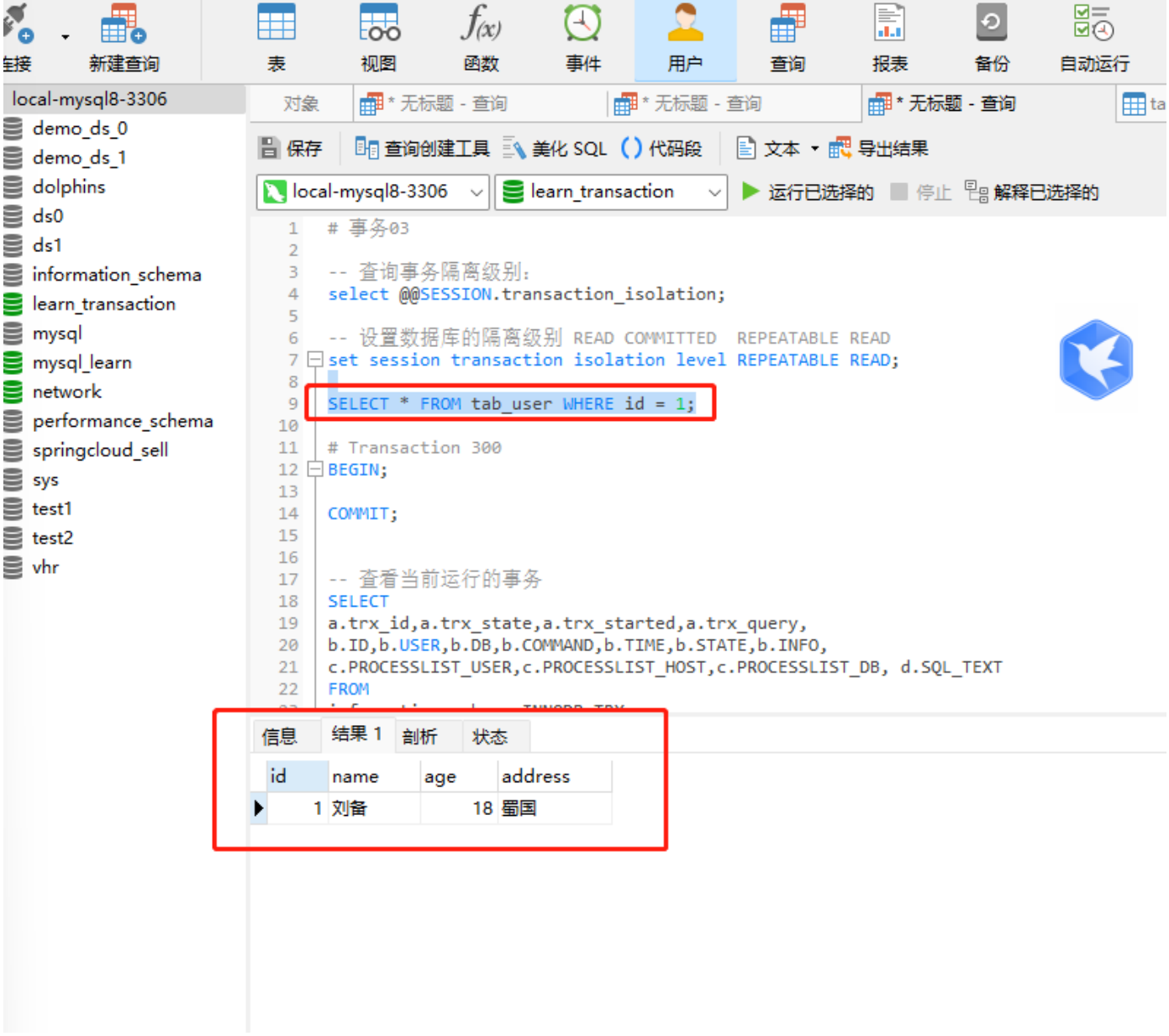
4.执行事务2中的更新语句，数据锁定，事务挂起



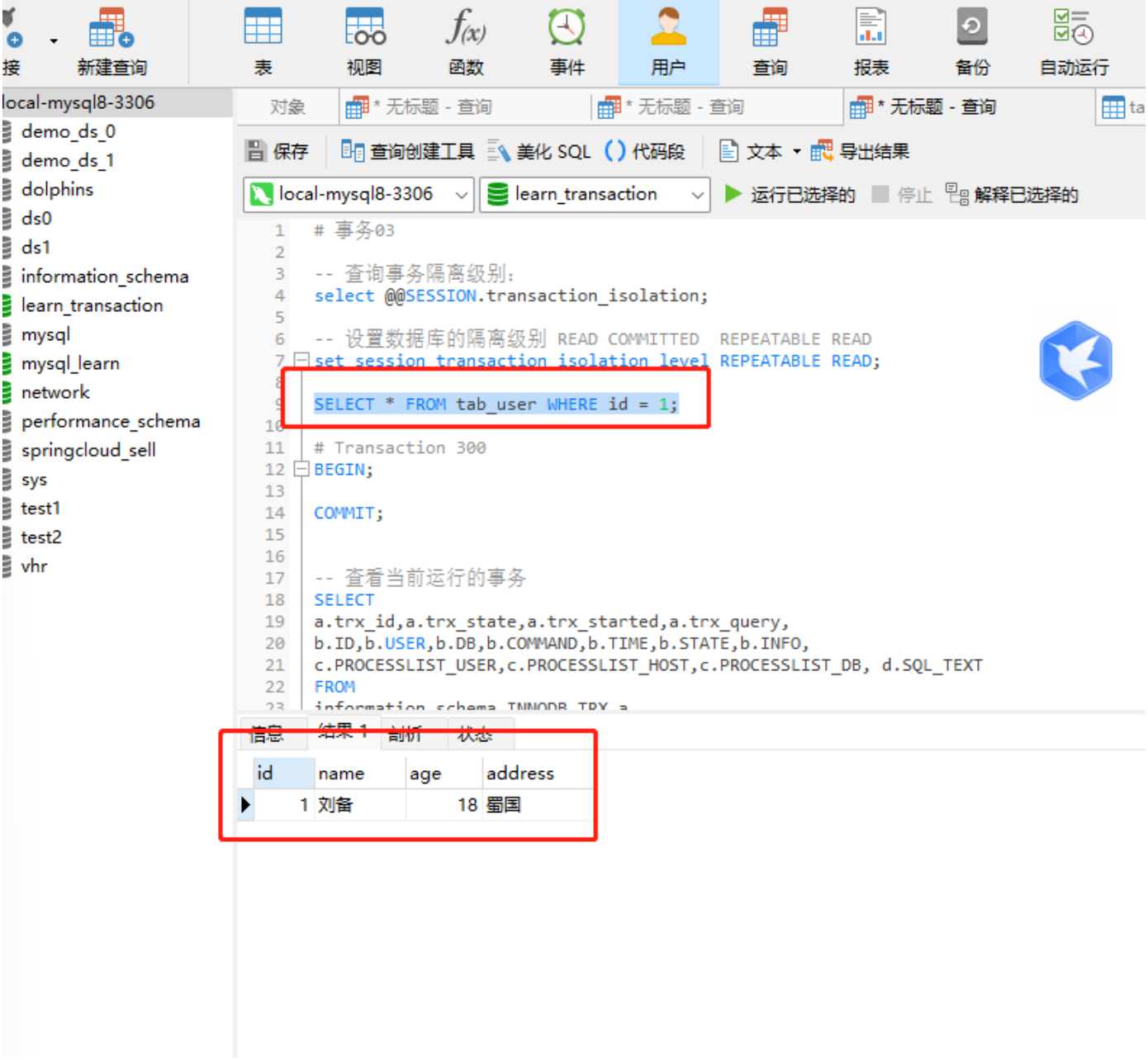
5.查询事务3的数据，查询的是刘备



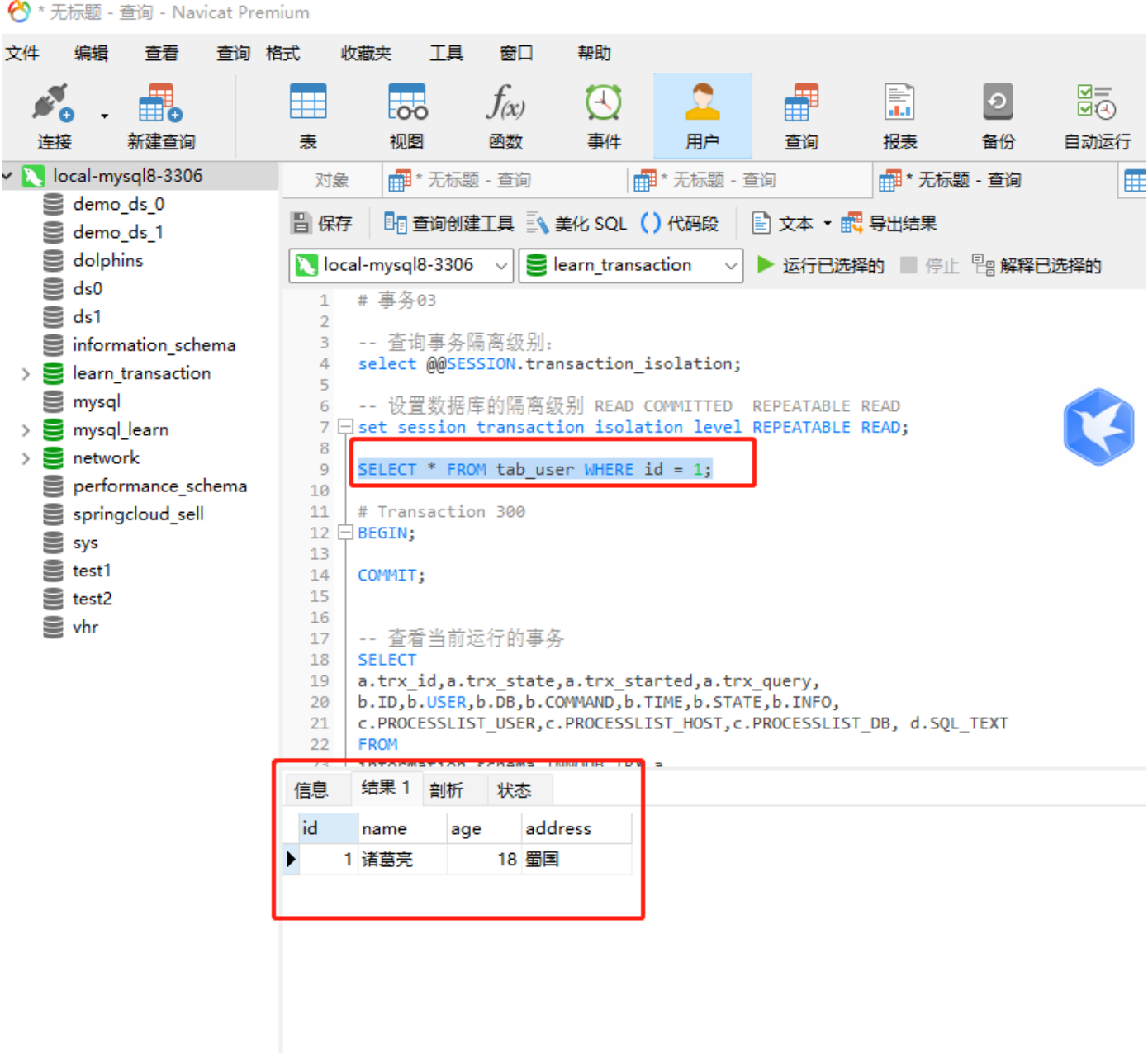
接着提交事务1，在查询事务三的数据，查询的仍然是刘备



接着更新事务2的数据，并提交书屋2，然后去查询事务3的数据，仍然查询的是刘备。



6.提交事务3，最后在查询事务3的数据，查询到诸葛亮，也就是事务2最后更新的结果。



- 结论

  RR级别的事务隔离级别，会解决不可重读和脏读的问题，但是扔存在幻读的问题

实践总结：

RC和RR的隔离级别的区别在于生成ReadView的时机不同。

RC是在select第一次查询的时候创建ReadView

RR在第一次进行普通select操作前生成一个ReadView，之后的操作重复使用该视图。

题目 02- 什么是索引?

要点：

1. 优点是什么?

   减少磁盘IO，提高查询效率

2. 缺点是什么?

   占用磁盘空间，增加、更新、删除数据的时候，会增加了额外的维护索引开销成本

3. 索引分类有哪些？特点是什么?

   主键索引：特殊的唯一索引，一个表只能有一个主键，不允许有空值，不允许重复，如果没有指明表主键，会默认生成一个主键

   辅助（二级）索引：

   1.唯一索引：不允许索引列的值重复，但是允许空值

   2.组合索引：在多个字段上组成一个索引，用于组合搜索，效率高于多个索引。

   3.全文索引：只用于文本数据类型的字段

   4.覆盖索引：select中列数据如果可以直接在辅助索引树上全部获取，也就是说索引树已经“覆盖”了我们的查询需求， 这时MySQL就不会白费力气的回表查询，这中现象就是覆盖索引。

4. 索引创建的原则是什么?

   1. 频繁出现在where 条件字段，order排序，group by分组字段

   2. select 频繁查询的列，考虑是否需要创建联合索引（覆盖索引，不回表）

   3. 多表join关联查询，on字段两边的字段都要创建索引

5. 有哪些使用索引的注意事项?

   1. 全值匹配我最爱

   2. 最左前缀匹配原则

   3. 不在索引列上做任何操作【计算、函数、类型转换】，会导致索引失效，转而使用全表扫描

   4. 存储引擎不能使用索引中范围条件右边的列

   5. 尽量使用覆盖索引【只访问索引的查询，索引列和查询列一致】，减少使用select *

   6. 不等于【!= 或 <>】，索引会失效

   7. is null，is not null，索引会失效

   8. like以通配符开头，索引会失效

   9. 字符串不加单引号，索引会失效

   10. 少用or，用它来连接时，索引会失效

6. 如何知道 SQL 是否用到了索引?

   explain分析sql,根据索引相关字段。

7. 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

  B+Tree只有叶子节点才会存储数据，非叶子节点只存储键的值，叶子节点底层是双向链表的数据结构。

  这样B+Tree可以存储更多的键的情况下，拥有更低的树高，保证等值和范围的快速查找

题目 03- 什么是 MVCC?

要点：

1. Redo 日志

   首先说明，redo日志是为了让已经提交的事务对数据库中数据的更改都能永久生效，即时服务器挂了，也能恢复数据。为了达到这个效果，在事务提交的时候都会将该事务中对内存中数据的修改都记录到redo日志中，所以也称重做日志。

2. ReadView

   它是一张存储事务id的表，主要包含当前系统中有哪些活跃的读写事务，结合undo日志中的字段事务trx_id，控制哪个版本的undo日志是否能被其他事务访问。

3. 如何判断可见性

   开启事务后，执行第一次查询的时候，首先生成ReadView,然后根据Undo日志和ReadView判断可见性。

   1.如果被访问版本的 trx_id 属性值，小于ReadView中的事务下限id，表明生成该版本的事务在生 成 ReadView 前已经提交，所以该版本可以被当前事务访问。

   2.如果被访问版本的 trx_id 属性值，等于ReadView中的 m_creator_trx_id ，可以被访问。

   3.如果被访问版本的 trx_id 属性值，大于等于ReadView中的事务上限id，在生成 ReadView 后才产 生的数据，所以该版本不可以被当前事务访问。

   4.如果被访问版本的 trx_id 属性值，在事务下限id和事务上限id之间，那就需要判断是不是在 m_ids 列表中。如果在，说明创建 ReadView 时生成该版本的事务还是活跃的，该版本不可以被访问；如果不在，说明创建 ReadView 时生成该版本的事务已经被提交，该版本可以被访问。