**数据库实验报告**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **实 验 人：** | **伍建霖** | | **学 号：** | **20337251** | **日 期：** |  |
| **院（系）：** | **计算机学院** | | | **专业（班级）：** | **网络空间安全** | |
| **联系方式：** | QQ 773542531 | | |  |  | |
| **实验题目：** | | | **1.4 数据更新实验** | | | | | |

* + - 1. **实验目的**

熟悉数据库的数据更新操作，能够使用 SQL 语句对数据库进行数据的插入、修改、 删除操作。

* + - 1. **实验内容和要求**

针对 TPC-H 数据库设计单个元组插入、批量数据插入、修改数据和删除数据等 SQL 语句。理解和掌握 insert、update 和 delete 语法结构的各个组成成分，结合嵌套 SQL 子查询，分别设计几种不同形式的插入、修改和删除数据的语句并调试成功。

* + - 1. **实验重点和难点**

实验重点：插入、修改和删除数据的 SQL。

实验难点：与嵌套 SQL 子查询相结合的插入、修改和删除数据的 SQL 语句；利用一 个表的数据来插入、修改和删除另外一个表的数据。

* + - 1. **实验工具**

MySQL、SQL Server、Navicat

* + - 1. **实验过程**

（1）insert 基本语句（插入全部列的数据）

插入一条顾客记录，要求每列都给一个合理的值。如：values(30303030, '张三', '北京市',40,'010-51001199',0.00, 'Northeast', 'VIP Customer')

INSERT INTO customer

values(30303030, '张三', '北京市',40,'010-51001199',0.00, 'Northeast', 'VIP Customer')

将其结果查询后截屏如下：

SELECT \* FROM customer WHERE custkey= 30303030;

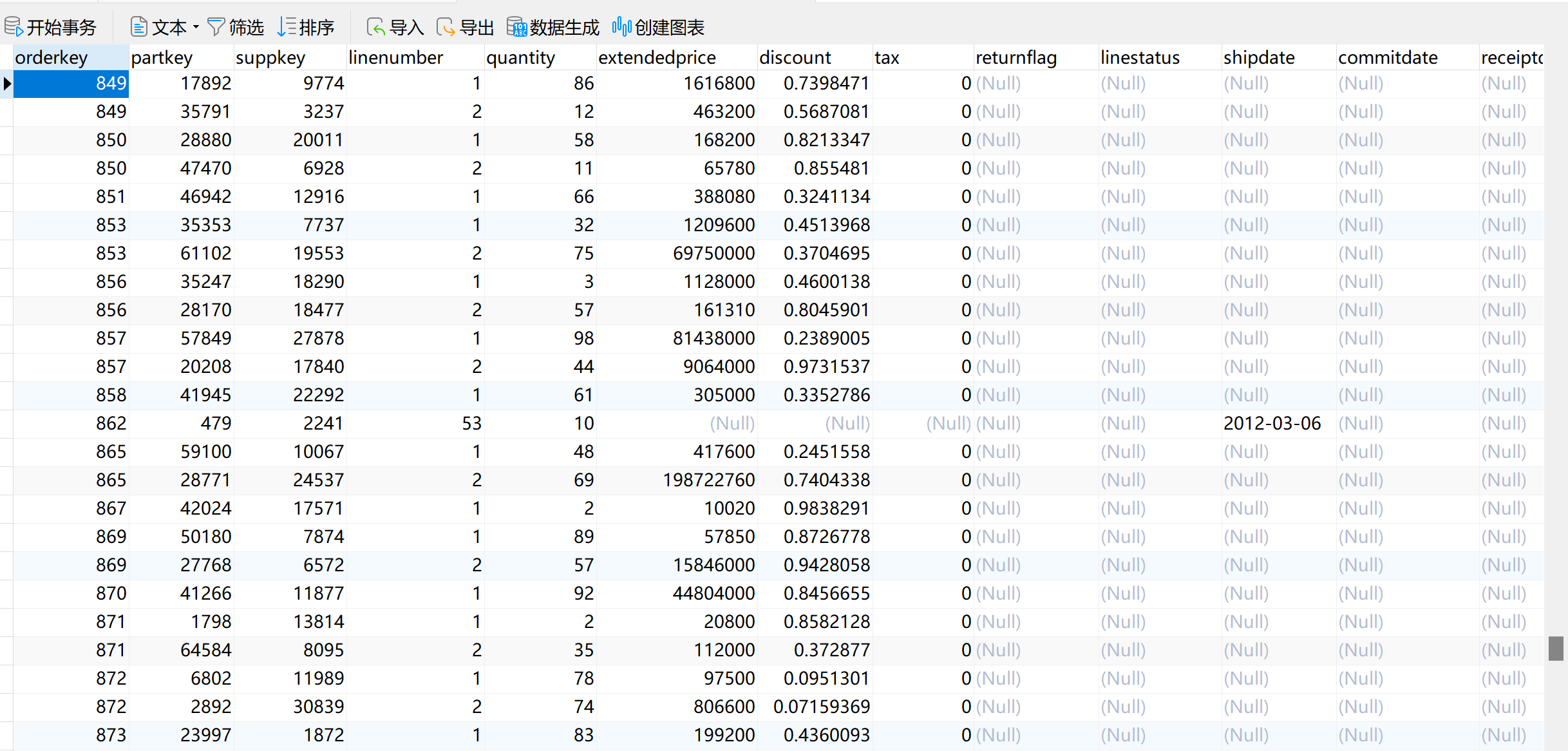


（2）insert 基本语句（插入部分列的数据） lineitem

插入一条订单记录，给出必要的几个字段值。如：(ORDERKEY,LINENUMBER,PARTKEY,SUPPKEY,QUANTITY,SHIPDATE)values(862,ROUND(RAND()\*100,0),479,2241,10,'2012-3-6')

INSERT INTO lineitem (ORDERKEY,LINENUMBER,PARTKEY,SUPPKEY,QUANTITY,SHIPDATE)values(862,ROUND(RAND()\*100,0),479,2241,10,'2012-3-6')

将其结果查询后截屏如下：



说明：由于在创建八个基本表的时候，已将所有列添加了约束：非空（不能为 NULL 值），因此在给 lineitem 插入部分列的时候会失败。

（3）批量数据 insert 语句

1. 创建一个新的顾客表NewCustomer，把所有来自“中国”的顾客插入到这个表NewCustomer

CREATE TABLE NewCustomer LIKE Customer;

INSERT INTO newcustomer

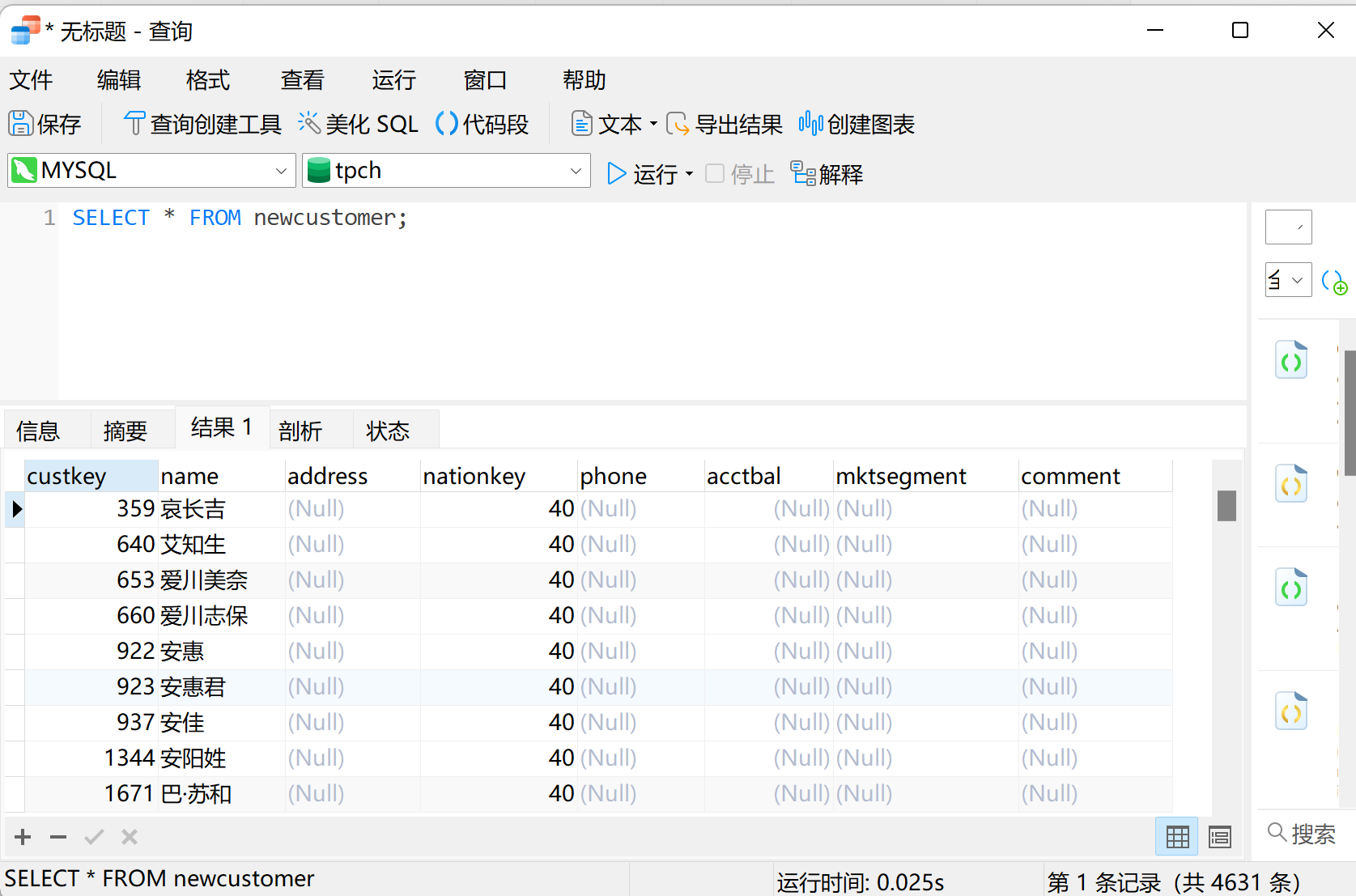
SELECT \*

FROM customer

WHERE customer.nationkey = 40;

将其结果查询后部分截屏如下：

SELECT \* FROM newcustomer;



2. 创建一个顾客购物统计表，记录每个顾客及其购物总数和总价等信息。

CREATE TABLE ShoppingStat(custkey INTEGER, quantity REAL, totalprice REAL);

INSERT INTO shoppingstat

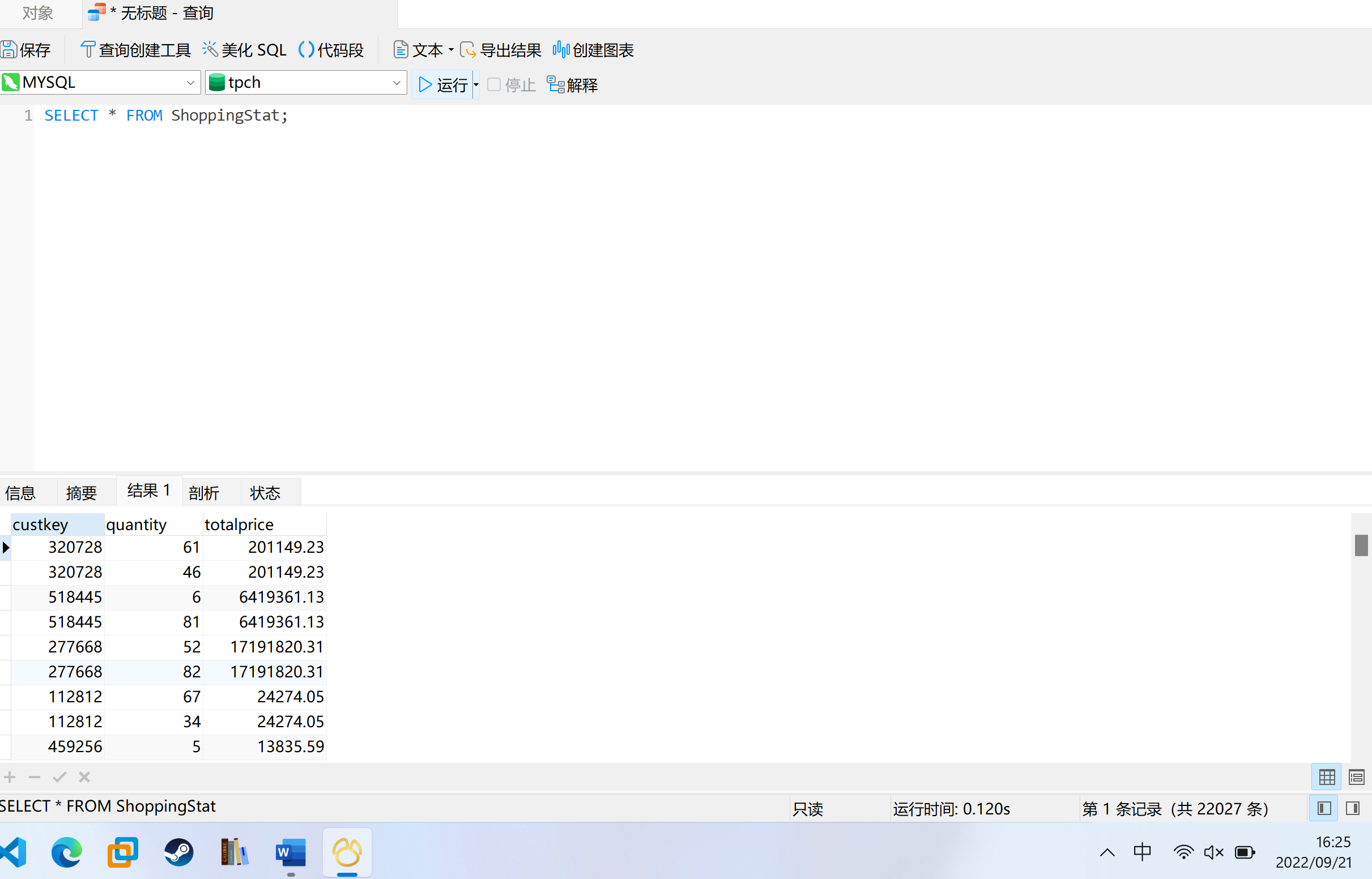
SELECT orders.custkey, lineitem.quantity, orders.totalprice

FROM orders, lineitem

WHERE orders.orderkey = lineitem.orderkey;

将其结果查询后截屏如下：

SELECT \* FROM ShoppingStat;



3. 倍增零件表的数据，多次重复执行，直到总记录数达到 50 万为止。

insert into PART

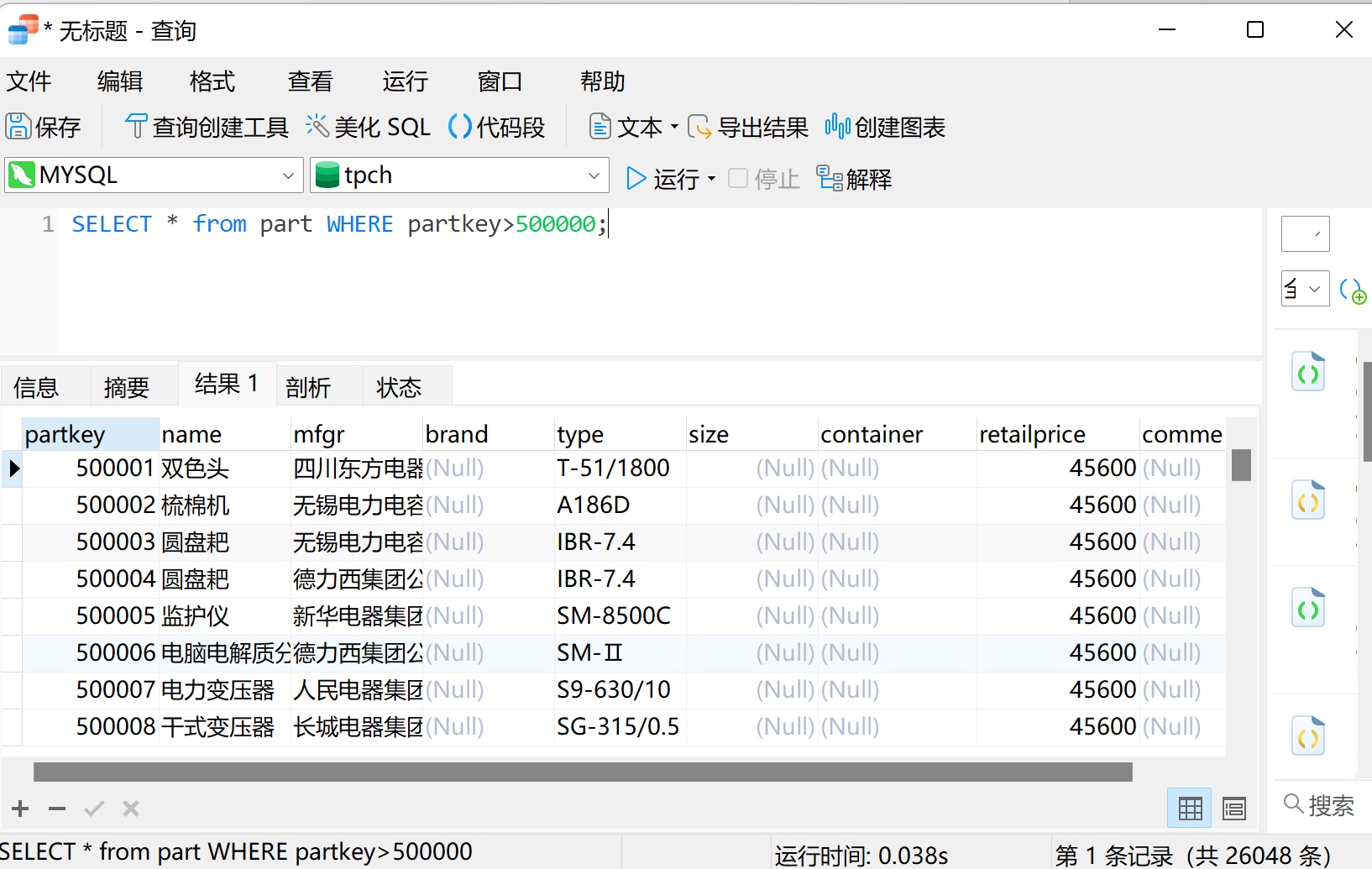
select PARTKEY+(select max(partkey) from PART), NAME, MFGR, BRAND, TYPE, SIZE, CONTAINER, RETAILPRICE, COMMENT

from PART;

多次执行上述语句后，观察表格可以发现，其最终结果满足要求。

将其结果查询后截屏如下：

SELECT \* from part WHERE partkey>500000;



（4）update 语句（修改部分记录的部分列值）

将“双思 集团”供应的所有零件的供应成本价下降10%。

UPDATE partsupp

SET supplycost = supplycost \* 0.9

WHERE suppkey in (SELECT suppkey FROM Supplier WHERE name='双思 集团') ;

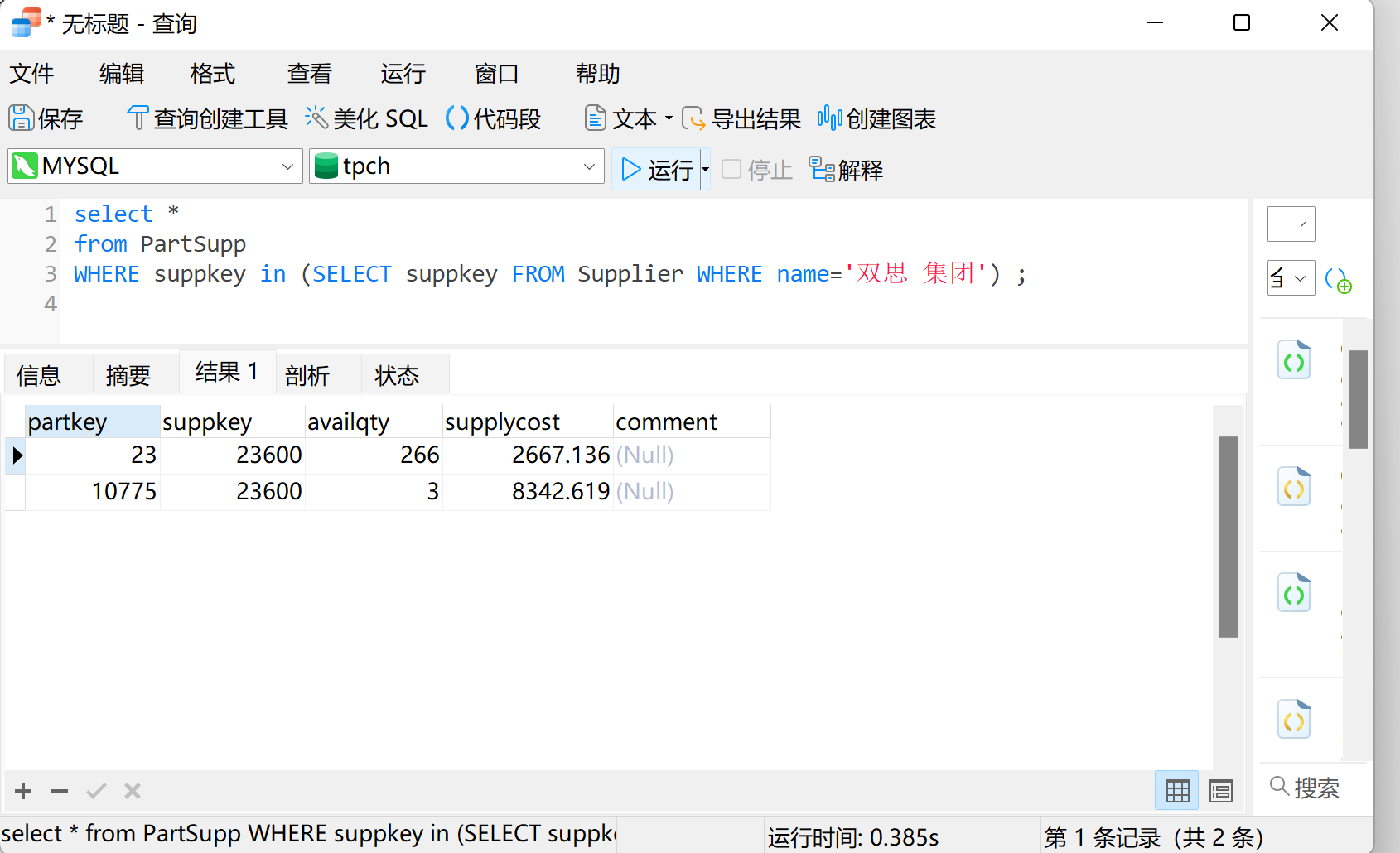
部分查询结果截屏如下：

select \*

from PartSupp

WHERE suppkey in (SELECT suppkey FROM Supplier WHERE name='双思 集团') ;

1 更新前：



2 更新后：



比较两者的 SUPPLYCOST 可以发现， SUPPLYCOST 的供应成本价较 update 前确实下降了 10%，其最终结果满足要求。

（5）update 语句（利用一个表的数据修改另外一个表的数据）

利 用 part 表 中 的 零 售 价 格 来 修 改 lineitem 中 的 extendedprice ， 其 中 extendedprice = part.retailprice\*quantity。

update LINEITEM L INNER JOIN PART P USING(partkey)

set L.EXTENDEDPRICE = P.RETAILPRICE \* L.QUANTITY

或者

UPDATE Lineitem

SET extendedprice = quantity \* (select retailprice from part where partkey=Lineitem.partkey);

（6）delete 基本语句（删除给定条件的所有记录）

删除顾客“阿波罗”的所有订单记录。

DELETE FROM lineitem

WHERE orderkey

IN (SELECT orderkey

FROM orders WHERE custkey

IN(SELECT custkey

FROM customer

WHERE name='阿波罗'));

DELETE FROM orders

WHERE orders.custkey

IN (SELECT custkey

FROM customer

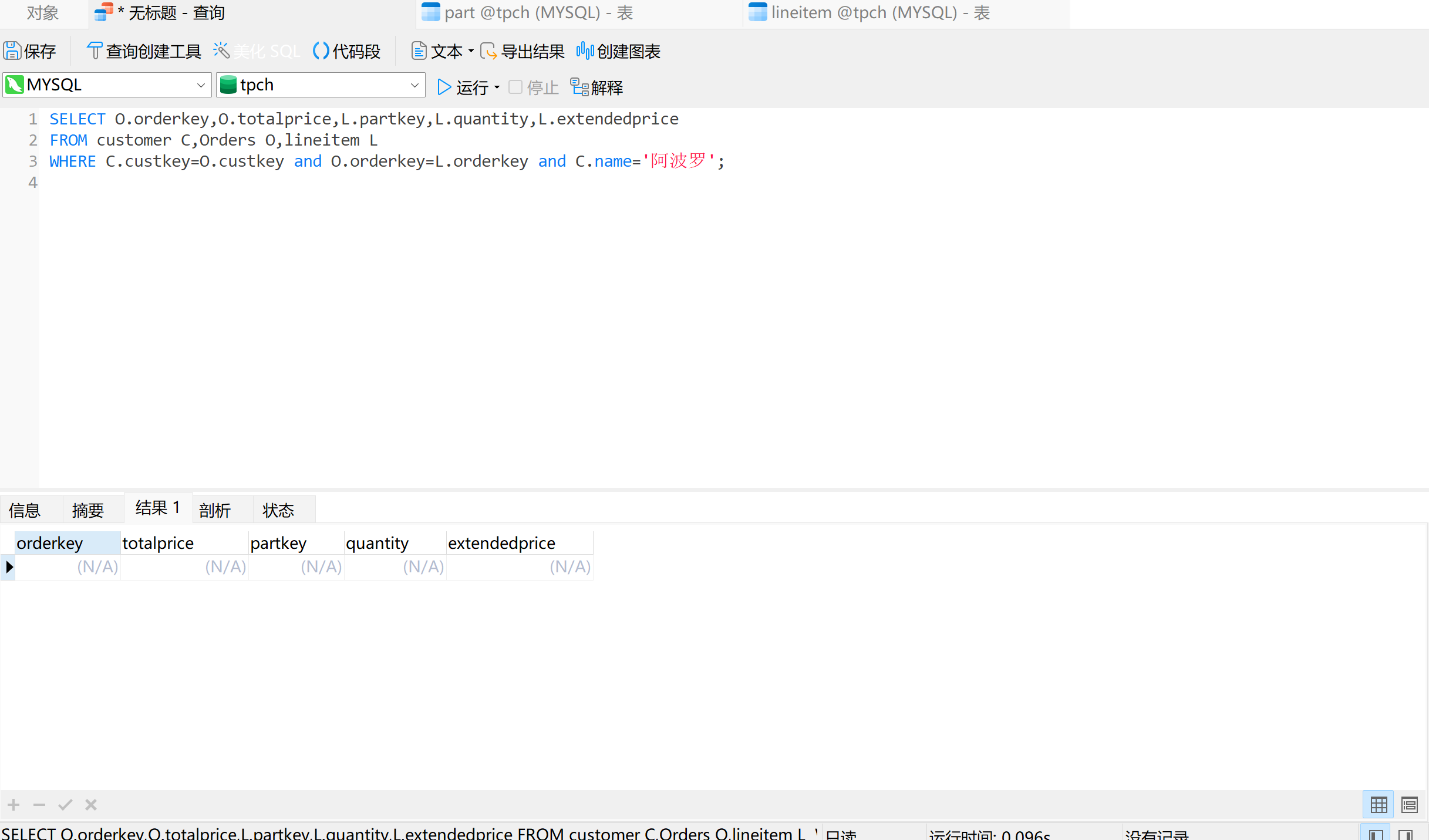
WHERE name='阿波罗');

再查询‘阿波罗’的订单记录，结果如下：

SELECT O.orderkey,O.totalprice,L.partkey,L.quantity,L.extendedprice

FROM customer C,Orders O,lineitem L

WHERE C.custkey=O.custkey and O.orderkey=L.orderkey and C.name='阿波罗';



说明：结果表明，已经找不到该顾客的订单记录，说明删除该顾客的订单记录成功。

* + - 1. **与实验结果相关的文件**

无

* + - 1. **实验总结**