

数据库系统课程实验报告

|  |  |
| --- | --- |
| 实验名称： | 数据基本查询 |
| 实验日期： | 4.11 |
| 实验地点： | 西片4号楼105 |
| 提交日期： | 4.19 |
|  | |
| 学号： | 37220232203786 |
| 姓名： | 潘腾凯 |
| 专业年级： | 软工2023级 |
| 学年学期： | 2024-2025学年第二学期 |

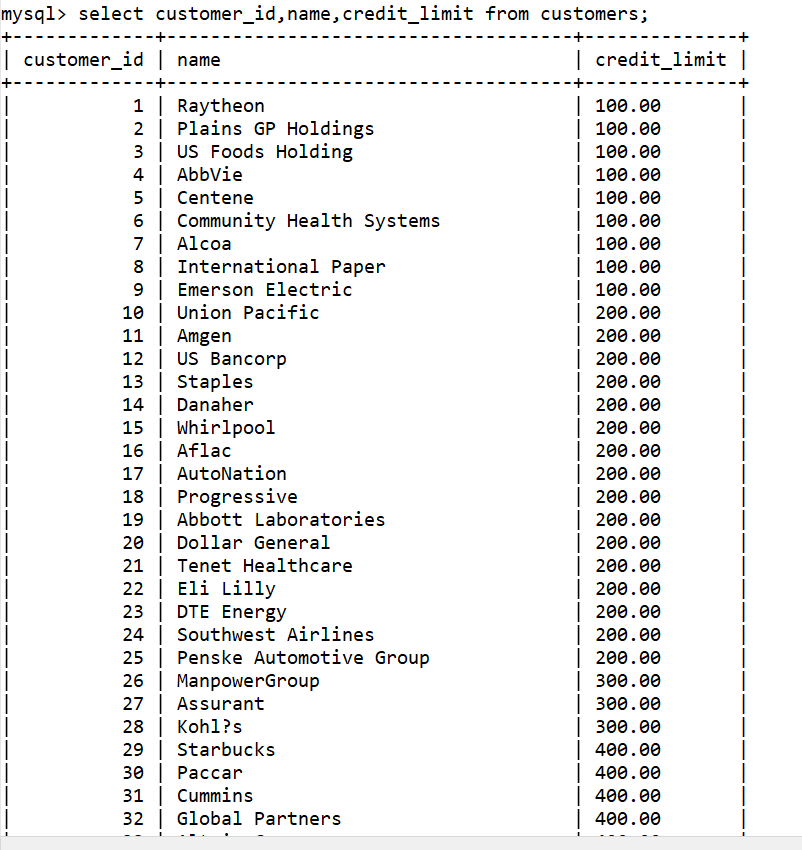
1. 实验目的

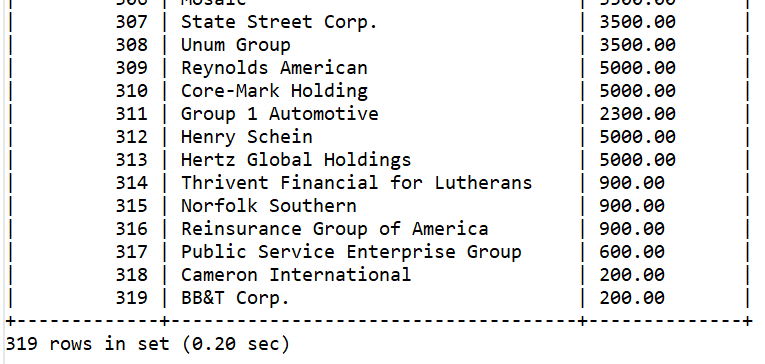
* 熟练掌握 mysql 单表查询的语法结构及其使用方法
* 掌握设计正确查询语句以实现查询要求的方法
* ─ 简单单表查询（此处指不涉及模糊、集合、聚集、分组、排序的查询）
* ─ 模糊查询、 聚集函数、分组统计和排序
* 掌握 Group by 的使用
* 正确区分元组过滤条件（WHERE 子句）和分组过滤条件（HAVING 短语）的异同
* 掌握 Order by 的使用
* 掌握使用 DISTINCT 实现查询结果去重的方法
* 掌握空值 NULL 的使用方法
* 掌握表别名的使用场合及方法
* 掌握自身连接的使用方法
* 掌握使用 LIMIT m [offset n]命令来控制输出行数的方法

1. 实验内容和步骤

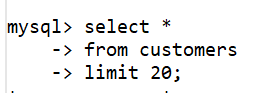
1.查询顾客表中的顾客号（customer\_id）、顾客名（name）和信用卡额度（credit\_limit）

步骤如下：



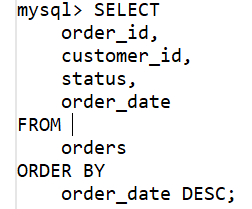


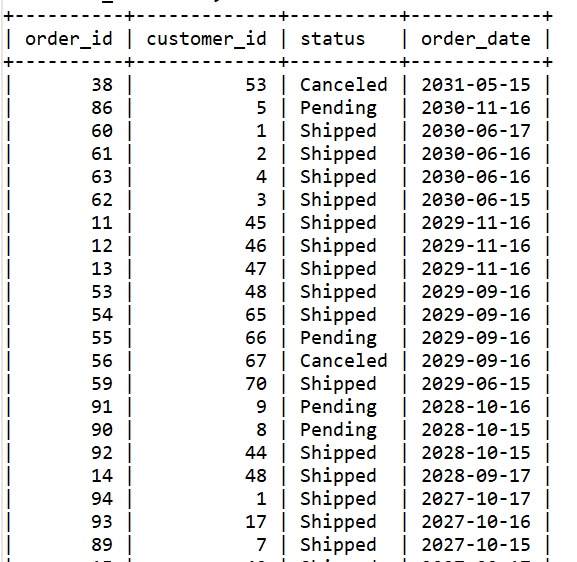
2. 查询顾客的所有信息，且只显示前 20 条记录

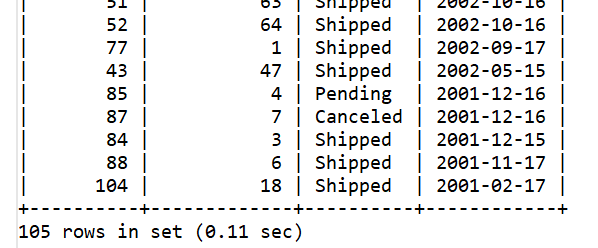




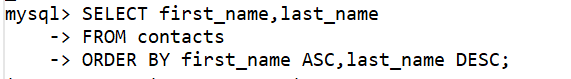
3. 查询订单表中的订单号，顾客号，状态，订单日期，并按订单日期降序显示结果

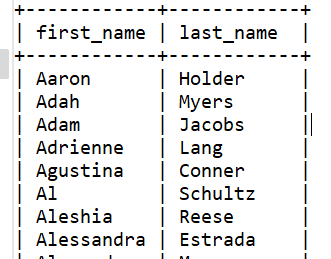


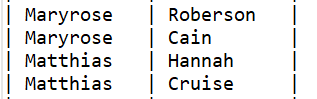


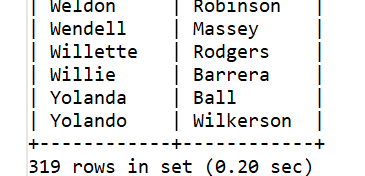


4. 查询联系表中的名（first name）和姓（last name），并按名升序，姓降序显示









5. 执行以下语句并观察 state 列 NULL 值的显示位置，得出结论

SELECT country\_id, city, state FROM locations ORDER BY city, state; SELECT country\_id, city, state FROM locations ORDER BY state ASC; SELECT country\_id, city, state FROM locations ORDER BY state DESC;

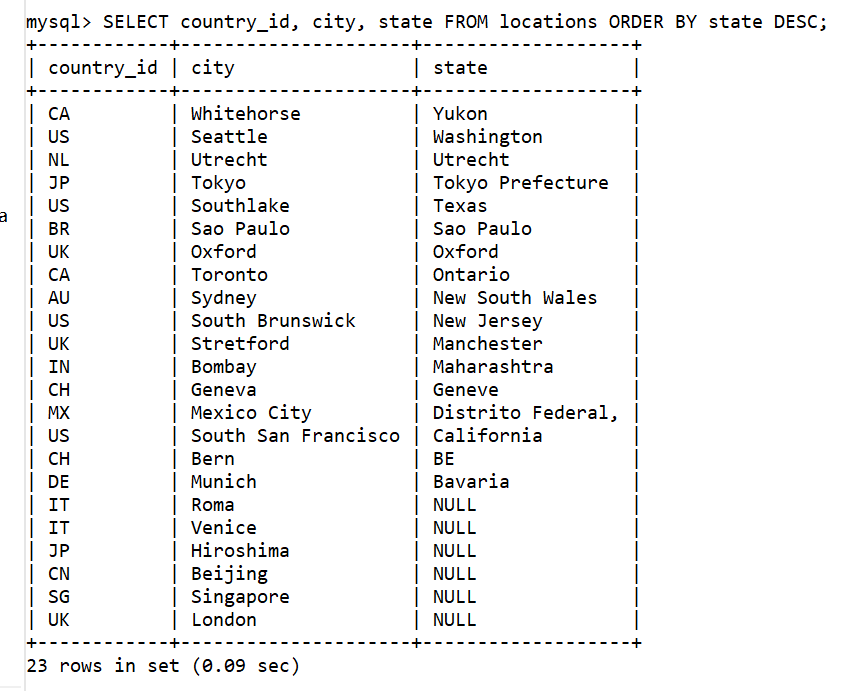
第一句



第二句



第三句



通过观察 state 列中 NULL 值的排序位置，可以得出以下结论：

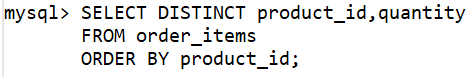
在升序排序中，NULL 值默认排在最前面。

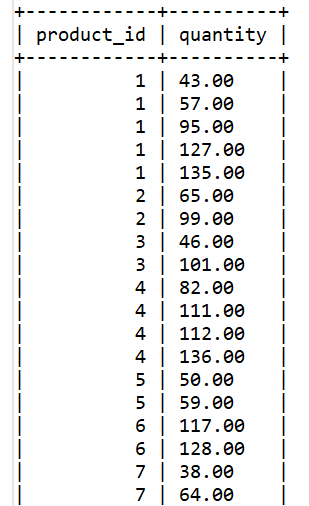
在降序排序中，NULL 值默认排在最后面。

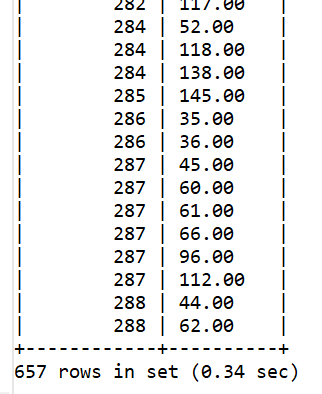
可以通过手动调整排序逻辑来改变 NULL 值的位置。

6. 查询订单细节表中（order\_items）的产品号和数量，查询结果应无重复元组

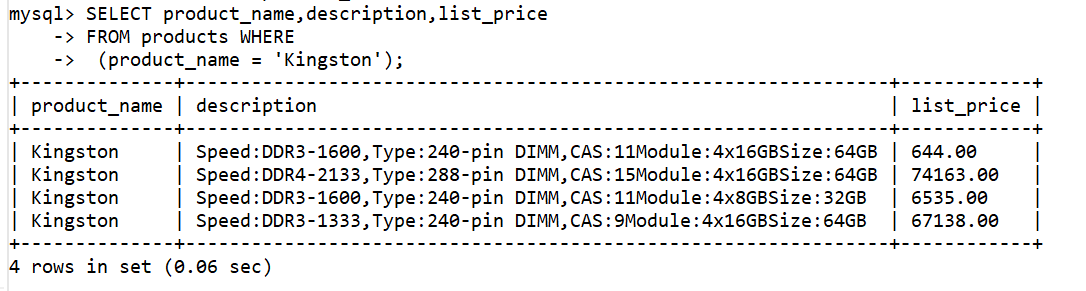
用DISTINCT避免重复，同时排序一下让结果看起来更直观



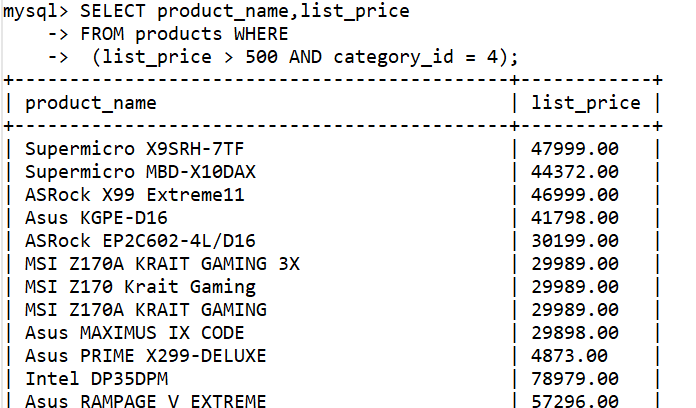


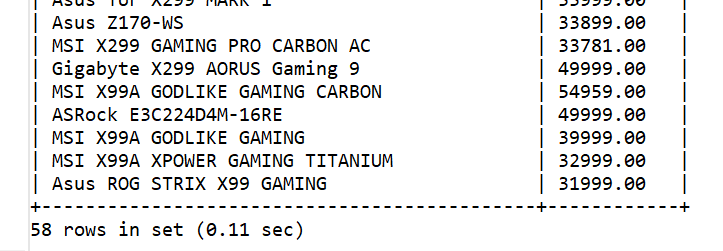


7. 查询产品表中的产品名为‘Kingston’的产品名，产品描述和价格

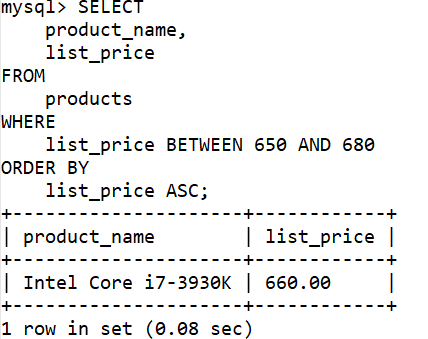


8. 查询产品表中所有价格大于 500 且 category\_id 为 4 的产品名和价格

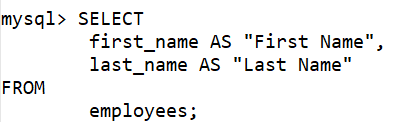




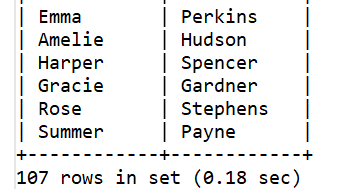
9. 查询产品表中所有价格在 650 和 680 之间的产品名和价格并按价格升序显示结果



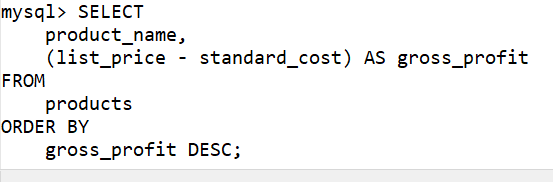
10. 查询雇员表中的名和姓，名和姓的字段分别显示为"First Name"和"Family Name"

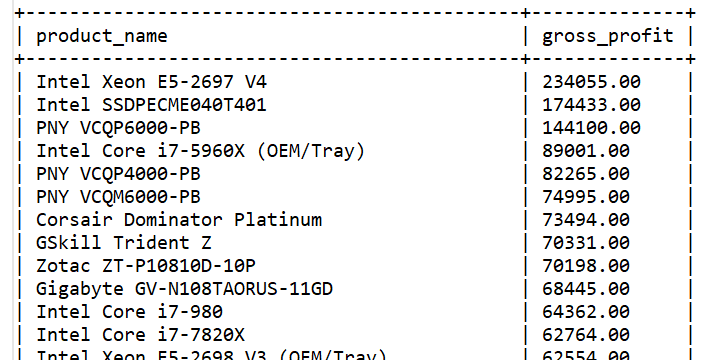


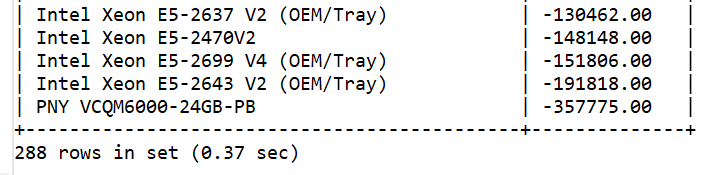




11. 查询产品表中的产品名及毛利，并按毛利结果降序显示，毛利名为 gross\_profit，毛利= list\_price -standard\_cost





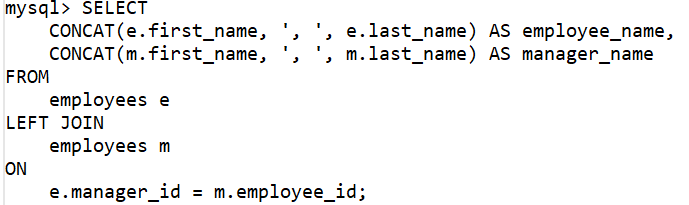


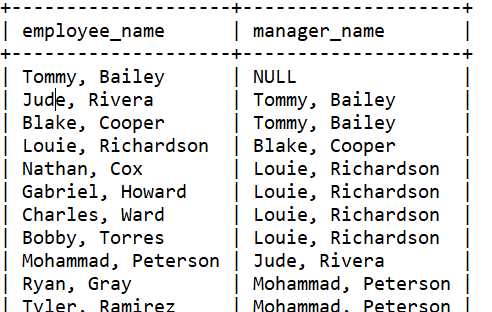
12. 查询雇员表中每个雇员对应的经理名，要求第一列字段名为 employee\_name，第二列字段名为manager\_name（雇员和经理的姓名同一格式为‘first\_name, last\_name’）

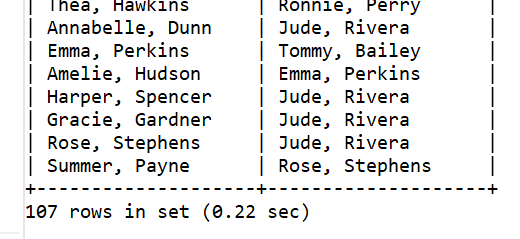
注： mysql 中常用字符串的连接可使用 concat()函数实现

SELECT CONCAT('Hello', ' ', 'World') AS result; -- 输出： Hello World

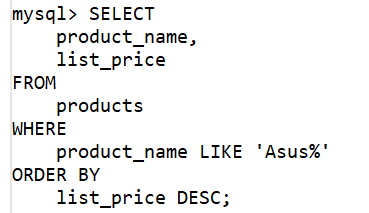
SELECT CONCAT(first\_name, ' ', last\_name) AS full\_name FROM users; -- 字段拼接

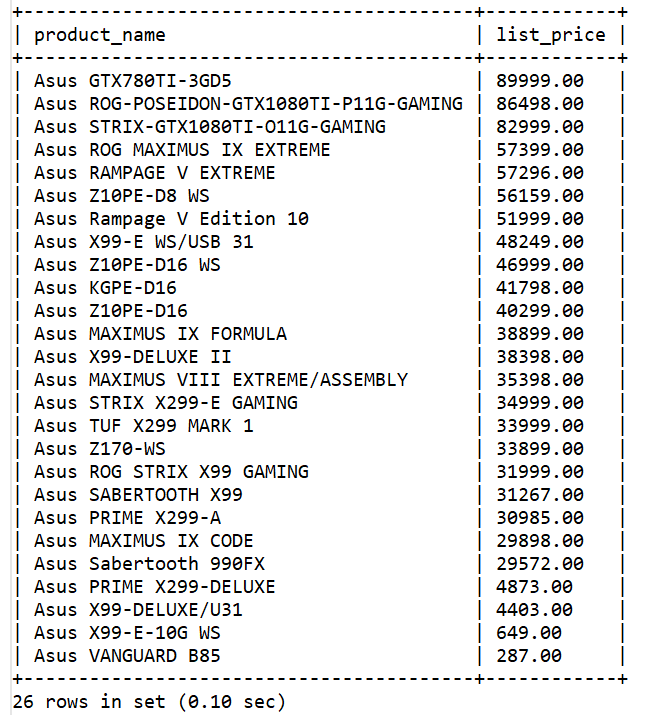




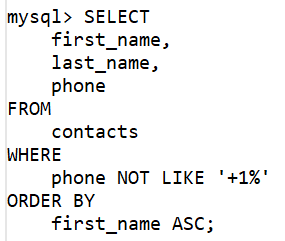


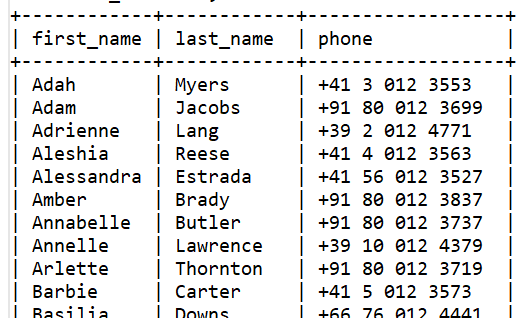
13. 查询产品表中所有以 Asus 开头的产品名和价格，并以价格降序显示

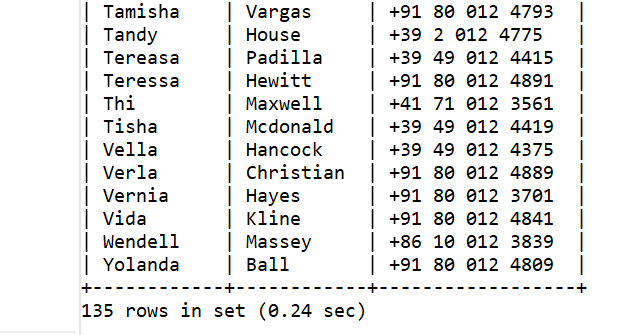




14. 查询联系表中电话号码不是以‘+1’开头的名、姓和电话号码，并以名升序显示

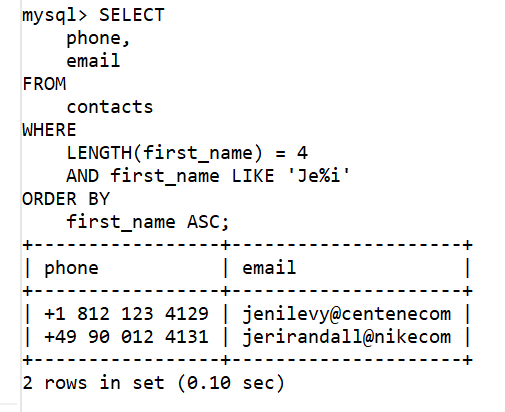




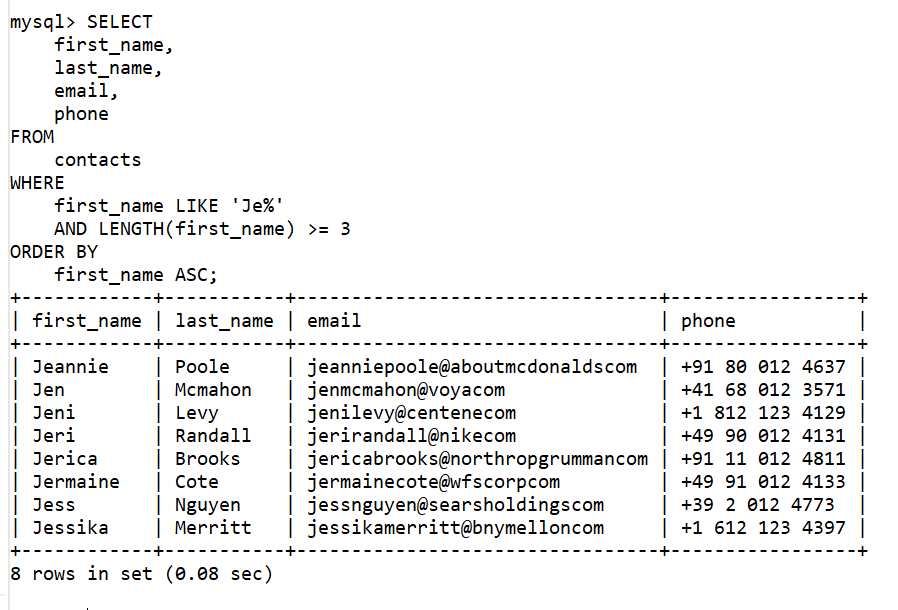


15. 查询联系表中的电话号码和电子邮件，要求名(first\_name) 的长度为 4 且以'Je'开头，以'i'结尾，按名

升序显示

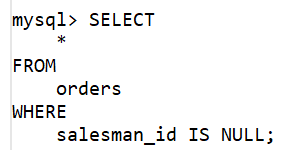


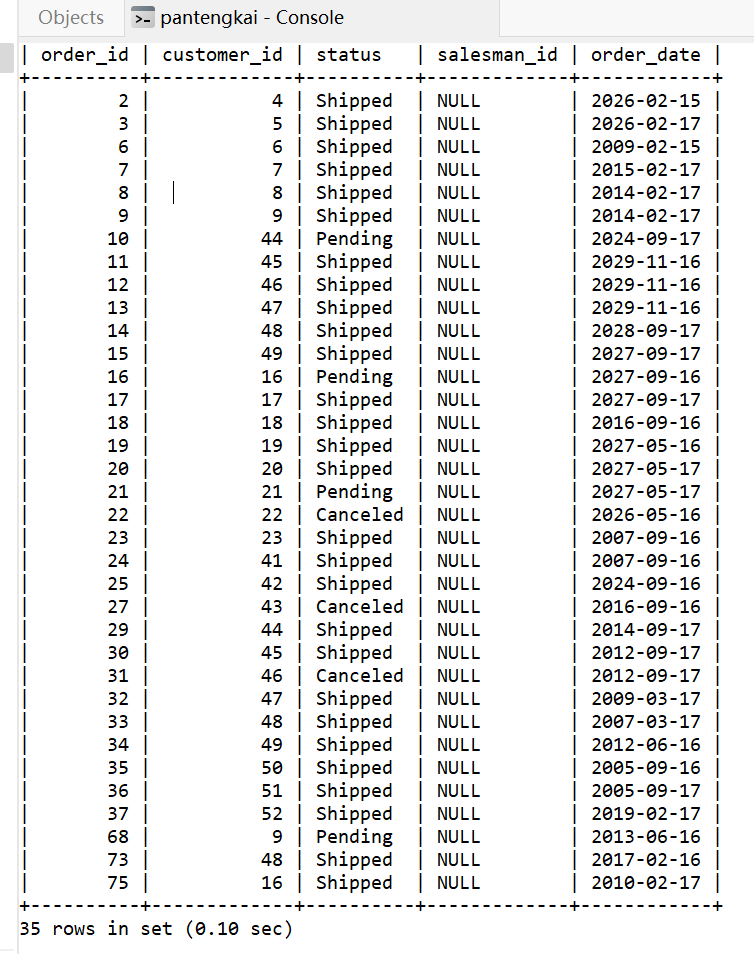
16. 查询联系表中所有以开头'Je'的名，且至少包含 3 个字符的名，姓，电子邮件和电话



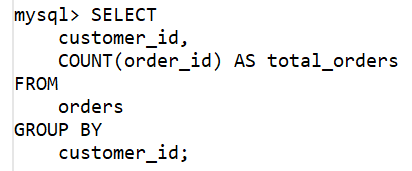
17. 查询订单表中所有没有销售员负责的订单（ i.e., query all sales orders that do not have a

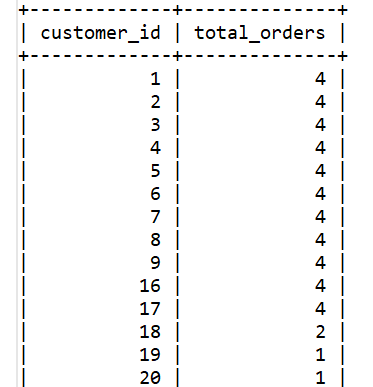
responsible salesman）

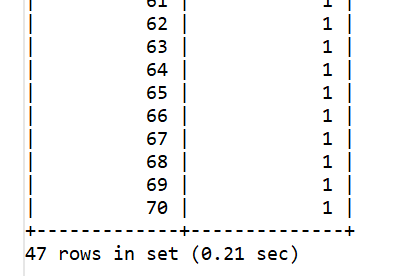




18. 统计每个顾客的订单总数（查询订单表）

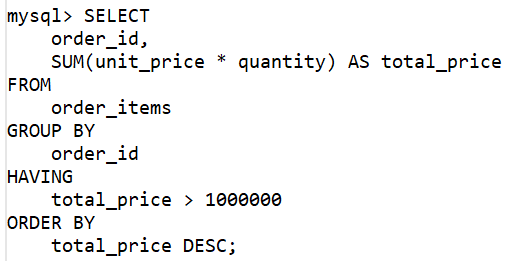


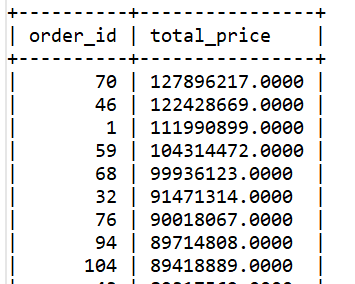


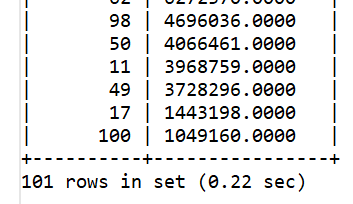


19. 统计每个订单的总价格大于 1000000 的订单号和总价格，并按总价格降序显示结果（通过查询订单细

节表 order\_items，总价格=unit\_price\*quantity）

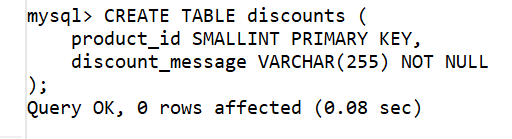


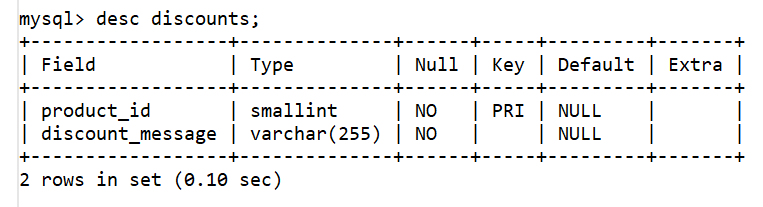




20. 创建一个折扣表 discounts，其表结构如下：

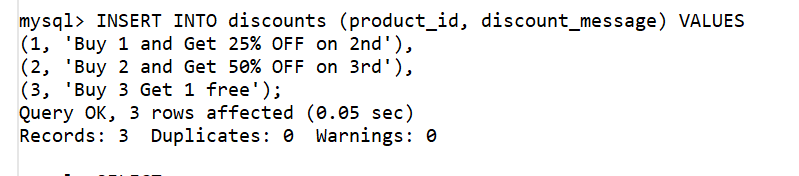
|  |  |  |  |
| --- | --- | --- | --- |
| 字段名 | 含义 | 数据类型 | 说明 |
| product\_id | 产品号 | smallint | 主码 |
| discount\_message | 折扣信息 | varchar(255) | 非空 |



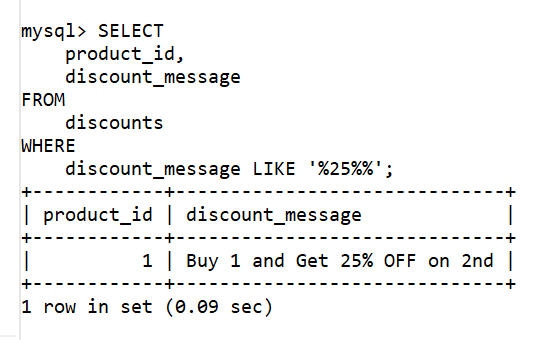


21. 给 discounts 表插入 3 条数据： (1, 'Buy 1 and Get 25% OFF on 2nd ')， (2, 'Buy 2 and Get

50% OFF on 3rd '), (3, 'Buy 3 Get 1 free'), 查询折扣表中折扣信息出现“25%”的产品号和折扣信息

插入：  


查询：



1. 实验总结

3.1 完成的工作

1.查询与显示数据：

查询雇员表中的名和姓，并为字段指定别名。

根据特定条件（如毛利计算、电话号码格式）查询并排序产品表和联系人表的数据。

2.自定义查询条件：

使用LIKE关键字筛选以特定字符串开头或结尾的数据（例如，以'Je'开头且以'i'结尾的名字）。

通过连接（JOIN）同一张表（自连接）来查询每个雇员对应的经理名。

3.聚合函数的应用：

统计每个顾客的订单总数，以及每个订单的总价格，并根据条件（如总价格大于100万）过滤结果。

4.创建表和插入数据：

创建了一个名为discounts的表，并向其中插入了三条记录。

查询了折扣信息中包含"25%"的产品号及其折扣信息。

3.2 对实验的认识

1.SQL语言的基础与应用：通过一系列具体的例子，加深了对SQL基本命令的理解，包括SELECT, FROM, WHERE, GROUP BY, HAVING, ORDER BY, JOIN, 和LIKE等。

2.数据处理技巧：学习了如何有效地进行数据筛选、分组、聚合及排序，这对数据分析至关重要。

3.实践中的注意事项：

理解了在SQL查询中处理NULL值的重要性。

学习了关于大小写敏感性的问题，特别是在使用LIKE进行模式匹配时。

探讨了性能优化的方法，比如使用索引提高查询效率。

3.3 遇到的困难及解决方法

无