

[**数 据 库 技 术 及 应 用 B**](javascript:setQuery('QueryName','%E6%95%B0%E6%8D%AE%E5%BA%93%E6%8A%80%E6%9C%AF%E5%8F%8A%E5%BA%94%E7%94%A8B'))

**第一次作业**

学院： 交通运输与物流学院

姓名： 刘欣豪

学号： 2020112921

专业： 交通工程

|  |
| --- |
| 作业得分： |
|  |

班级： 交通四班

指导老师： 葛乾

1. **作业目的**
2. 熟悉PyQGIS的各种函数操作，对读入与写入进行代码编写。
3. 对地理信息系统知识进行有效复习。
4. 融会贯通本课程的知识与操作。
5. **作业环境**
   1. 硬件环境
6. 计算机：OMEN by HP Laptop 15
7. CPU: Intel(R) Core(TM) i5-9300H CPU @ 2.40GHz
8. RAM：16GB
   1. 软件环境
9. 操作系统：Windows10家庭中文版
10. 开发工具：DataGrip 2022.2.5
11. **作业内容**

**必选题**

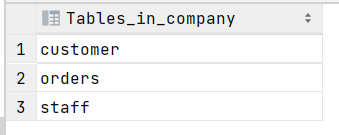
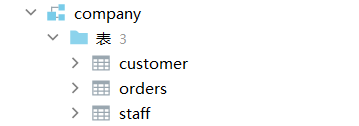
**1、数据库创建与关系创建：**

代码：

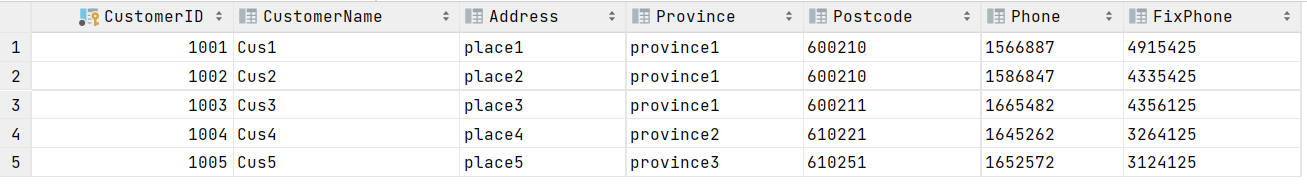
1. **create** **schema** company;
2. use company;
4. **create** **table** customer(CustomerID **int**(10) not null **primary** **key**,
5. CustomerName **char**(10),
6. Address **char**(20),
7. Province **char**(10),
8. Postcode **char**(10),
9. Phone **char**(10),
10. FixPhone **char**(10));
12. **create** **table** staff(StaffID **int**(10) not null **primary** **key**,
13. StaffName **char**(10),
14. WorkType **char**(10),
15. Department **char**(10),
16. Address **char**(20),
17. Phone **char**(10),
18. Hiredate **date**,
19. Salary **float**,
20. Accept996 bool);
22. -- 建立订单
23. **create** **table** Orders(OrderID **char**(10) not null **primary** **key**,
24. CustomerID **int**(10),
25. StaffID **int**(10),
26. OrderAmount **float**,
27. DeliveryDate **date**,
28. **constraint** order\_cus **foreign** **key**(CustomerID) **references** customer(CustomerID),
29. **constraint** order\_staff **foreign** **key**(StaffID) **references** staff(StaffID));

结果：

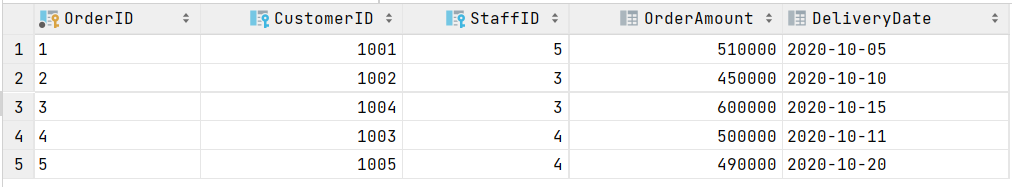
运行show tables；

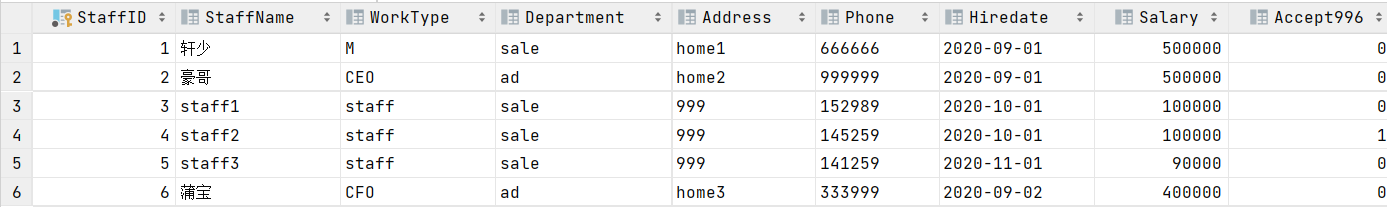
同时对表进行数据填充得到：



**customer**



**orders**



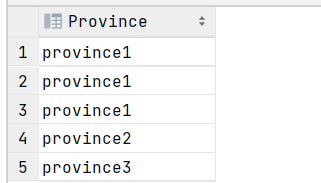
**staff**

**2、查询公司客户的分布省份：**

代码：

1. **select** Province **from** customer;

结果：

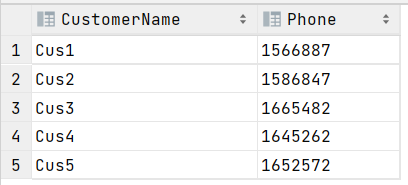


**3、查询所有不位于四川的客户的姓名和手机号码：**

代码：

1. **select** CustomerName,Phone **from** customer **where** Province!='成都';

结果：

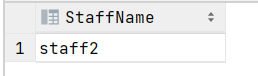


**4、查询所有接受996 工作制的员工的姓名：**

代码：

1. **select** StaffName **from** staff **where** Accept996=1;

结果：

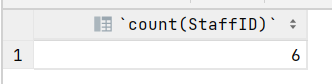


**5、查询员工总人数（提示：使用count 聚集函数）：**

代码：

1. **select** count(StaffID) **from** staff;

结果：

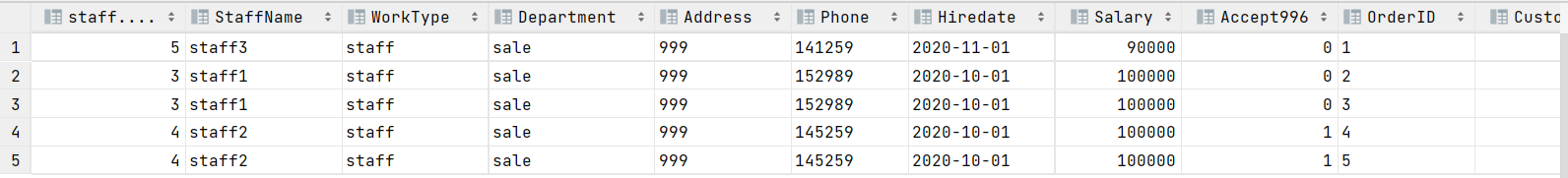


**6、查询每个员工以及其负责订单的情况：**

代码：

1. **select** staff.\*,orders.\* **from** staff,orders **where** staff.StaffID=orders.StaffID;

结果：

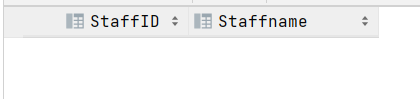


**7、查询负责订单金额在100 万以上的所有员工编号与姓名：**

代码：

1. **select** staff.StaffID,staff.Staffname
2. **from** staff,orders
3. **where** staff.StaffID=orders.StaffID and orders.OrderAmount>=1000000;

结果：

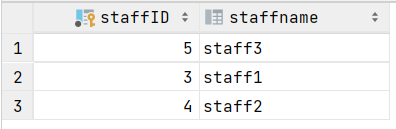


**8、查询负责的每个订单金额都在10 万以上的员工编号与姓名：**

代码：

1. **select** staff.staffID,staff.staffname
2. **from** staff,orders
3. **where** staff.StaffID=orders.StaffID
4. **group** **by** StaffID
5. **having** **min**(OrderAmount)>=100000;

结果：

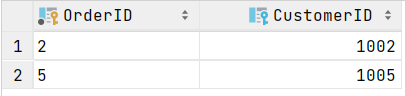


**9、查询出每个员工低于自己负责订单的平均金额的订单号与客户编号：**

代码：

1. **select** OrderID,CustomerID
2. **from** orders x
3. **where** OrderAmount<(**select** avg(OrderAmount)
4. **from** orders y
5. **where** x.staffID = y.staffID);

结果：

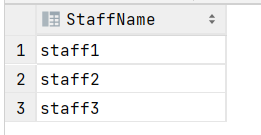


**10、假设公司有自动驾驶组、智能网联组、地图数据与数据可视化三个业务部门，查询非自动驾驶组中比自动驾驶组员工的平均薪水高的员工姓名与年龄：**

代码：

1. **select** StaffName
2. **from** staff x
3. **where** Salary<(**select** avg(Salary)
4. **from** staff y
5. **where** y.Department = 'ad') and x.Department = 'sale';

结果：

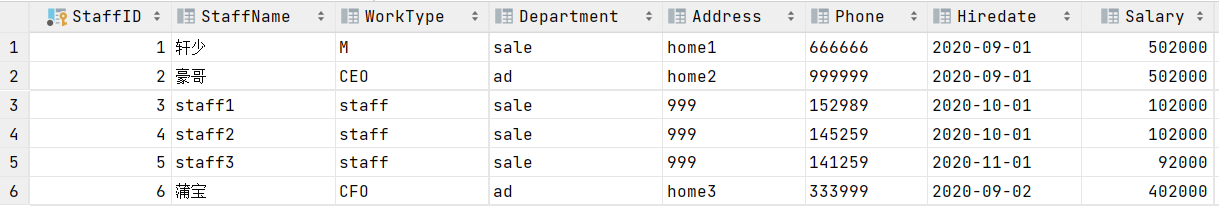


**11、由于国庆节提前上班，将所有员工的薪水修改为增加2000 元后的金额：**

代码：

1. **update** staff
2. **set** Salary = Salary+2000;

结果：

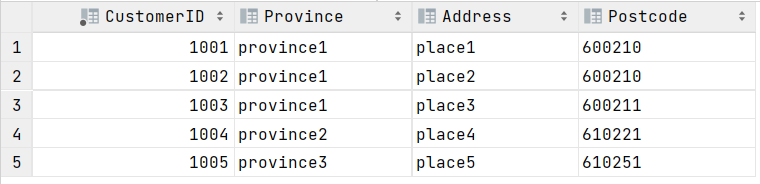


**12、定义一个客户详细地址（编号、省市区、住址、邮编）的视图：**

代码：

1. **create** **view** Address(CustomerID,Province,Address,Postcode)
2. **as**
3. **select** CustomerID,Province,Address,Postcode
4. **from** customer;

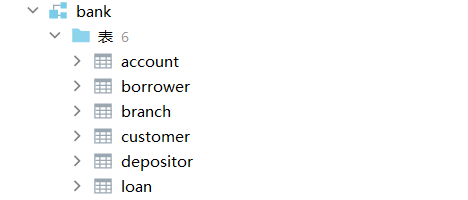
结果：



**加分题**

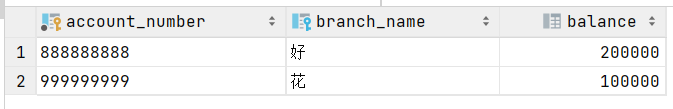
1. **建表**
2. **create** **table** branch(
3. branch\_name **char**(10) not null **primary** **key**,
4. branch\_city **char**(10),
5. assets **float**
6. );
8. **create** **table** customer(
9. ID **char**(10) not null **primary** **key**,
10. customer\_name **char**(10),
11. customer\_street **char**(20),
12. customer\_city **char**(10)
13. );
15. **create** **table** loan(
16. loan\_number **char**(10) not null **primary** **key**,
17. branch\_name **char**(10),
18. amount **float**,
19. **CONSTRAINT** branch\_loan
20. **FOREIGN** **KEY**(branch\_name) **REFERENCES** branch(branch\_name)
21. );
23. **create** **table** borrower(
24. ID **char**(10) not null ,
25. loan\_number **char**(10) not null ,
26. **primary** **key**(ID,loan\_number)
27. );
29. **create** **table** account(
30. account\_number **char**(10) not null **primary** **key**,
31. branch\_name **char**(10),
32. balance **double**,
33. **CONSTRAINT** account\_branch
34. **FOREIGN** **KEY**(branch\_name) **REFERENCES** branch(branch\_name)
35. );
37. **create** **table** depositor(
38. ID **char**(10),
39. account\_number **char**(10),
40. **primary** **key** (ID,account\_number)
41. );

结果：

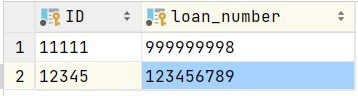


插入数据：

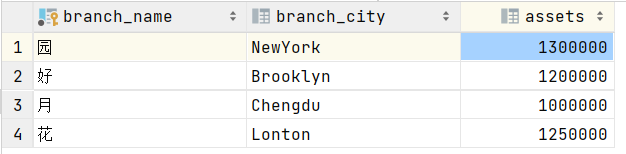
account：



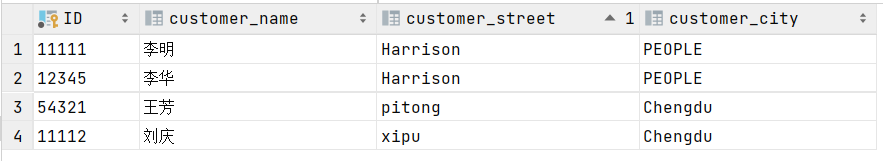
borrower：



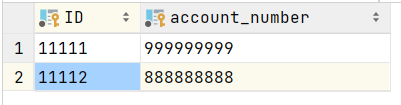
branch：



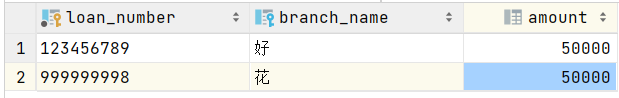
customer：



depositor：



loan：

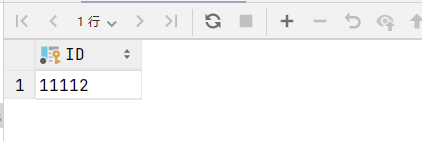


1. **Find the ID of each customer of the bank who has an account but not a loan.**

代码：

1. **select** customer.ID
2. **from** customer
3. **where** customer.ID in
4. (**select** ID
5. **from** depositor)
6. and customer.ID not in
7. (**select** ID
8. **from** borrower);

结果：



11112为有账户没贷款的账户，查询成功。

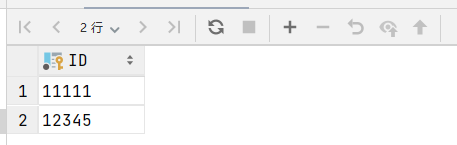
其中

1. **Find the ID of each customer who lives on the same street and in the same city as customer ‘12345’.**

代码：

1. **select** ID **from** customer
2. **where** customer\_street=(**select** customer\_street
3. **from** customer
4. **where** ID = '12345' )
5. and customer\_city=(**select** customer\_city
6. **from** customer
7. **where** ID = '12345');

结果：



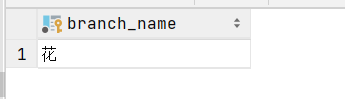
11111与12345都位于同一城市的同一街道，结果输出成功。

1. **Find the name of each branch that has at least one customer who has an account in the bank and who lives in “Harrison”.**

代码：

1. **select** branch.branch\_name **from** branch,account
2. **where** (**select** count(customer.ID) **from** customer,depositor
3. **where** customer.ID = depositor.ID
4. and customer\_street = 'Harrison'
5. and account.account\_number = depositor.account\_number)>=1
6. and branch.branch\_name = account.branch\_name;

结果：



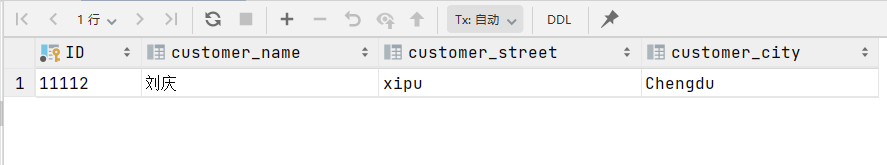
符合结果。

1. **Find each customer who has an account at every branch located in“Brooklyn”.**

代码：

1. **select** \*
2. **from** customer
3. **where** ID in
4. (**select** ID
5. **from** depositor
6. **where** account\_number in
7. (**select** account\_number
8. **from** account
9. **where** branch\_name in
10. (**select** branch\_name
11. **from** branch
12. **where** branch\_city = 'Brooklyn')));

结果：



刘庆的账户所在银行位于Brooklyn，结果正确

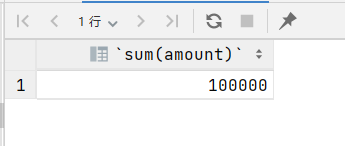
1. **Find the sum of all loan amounts in the bank.**

（全部银行的总和）

代码：

1. **select** sum(amount) **from** loan;

结果：

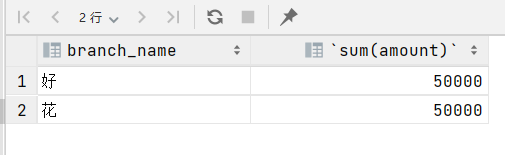


（单个银行的名字与总和）

代码：

1. **select** branch\_name, sum(amount) **from** loan
2. **group** **by** branch\_name;

**结果：**



1. **Find the names of all branches that have assets greater than those of at least one branch located in “Brooklyn”.**

代码：

1. **select** branch\_name **from** branch
2. **where** assets>any(**select** assets **from** branch
3. **where** branch\_city = 'Brooklyn');

结果：

