

学号 WA2214014 专业 人工智能 姓名 杨跃浙  
实验日期 05.30 教师签字 成绩

# 实验报告

【实验名称】 实验 2-简单查询实验

## 【实验目的】

1. 熟练掌握 SQL 语言查询语句 SELECT 的基本语法;
2. 熟悉各种表达选择查询条件和展示结果目标列的方式;
3. 理解分组的概念，熟练掌握使用 GROUP BY 子句进行分组查询;
4. 熟练使用 ORDER BY 子句对结果进行排序;
5. 理解和掌握各种聚集函数的功能和应用。

## 【实验原理】

### 0.简单查询

语句格式

```
SELECT [ALL|DISTINCT|TOP n] <目标列表达式>
[, <目标列表达式>] ...
[ INTO <新表名>]
FROM <表名或视图名>[, <表名或视图名>] ...
[ WHERE <条件表达式 1> ]
[ GROUP BY <列名 1> [ HAVING <条件表达式 2> ] ]
[ ORDER BY <列名 2> [ ASC|DESC ][,<列名 3> [ ASC|DESC ]... ]];
```

语法格式中参数的简要说明:

ALL|DISTINCT : 标识在查询结果集中是否显示相同行;

<目标列表达式>[,<目标列表达式>] … : 指定查询结果集中要显示的目标列;

INTO <新表名>: 将查询结果集插入一个新的数据表中;

FROM <表名或视图名>[,<表名或视图名>] : 指定查询操作的数据源, 可以是一个或多个基本表或视图, 或者查询结果;

WHERE <条件表达式 1> : 指定限定返回行的选择条件;

GROUP BY <列名 1>: 指定查询的分组条件;

HAVING <条件表达式 2>: 指定对分组或者聚合进行选择的条件, 必须与 GROUP BY 配合使用;

ORDER BY : <列名 2> [ ASC|DESC ][,<列名 3> [ ASC|DESC ]]…: 指定查询结果集的排序方式; ASC 升序, DESC 降序, 默认为 ASC 升序。

## 1. 选择表中的若干列

- (1) 查询指定列
- (2) 查询全部列

两种表达方式:

- 1) 在 SELECT 关键字后列出基本表中的所有的列名
- 2) 若顺序相同, 将<目标列表达式>指定为通配符“\*”。

- (3) 取消重复元组

两个原本不相同的元组投影到指定的某些列上时, 可能成为相同的元组。

用关键字 DISTINCT 来取消重复元组。

- (4) 使用别名查询
- (5) 查询经过计算的值

SELECT 命令动词后的<目标列表达式>[,<目标列表达式>] … 可以是一个或几个对某些列进行计算的表达式(算术表达式,字符串常量,函数,列别名等), 查询结果则是对这些列计算而得到的结果数据。

## 2. 选择表中的若干元组

### (1) 使用比较运算符进行条件查询

语法格式:

WHERE <表达式 1><比较运算符>< 表达式 2>;

其中 <表达式 1>和< 表达式 2>表示要进行比较的表达式，其中之一应包含关系的属性列；

比较运算符主要有=、<、>、>=、<=、<>、!=等。

### (2) 使用范围运算符进行条件查询

语法格式:

WHERE <表达式> [NOT] BETWEEN <值 1> AND <值 2>;

### (3) 使用列表运算符进行条件查询（确定集合）

语法格式:

WHERE <表达式> [NOT] IN (<值列表>) ;

其中<值列表>可以看成一个集合，则 IN 相当于集合的属于 ( $\in$ ) 运算。

### (4) 使用字符匹配运算符进行条件查询

语法格式:

WHERE <表达式> [NOT] LIKE <‘字符串’>;

此操作符只有在它的模式匹配整个串的时候才能成功。如果要匹配在串内任何位置的序列，该模式必须以百分号开头和结尾。

<'字符串'>可以使用通配符，一般常见通配符有以下两种:

1) %: 匹配任意多个字符;

2)\_: 匹配单个字符 (openGauss 中为 Unit8 编码汉字为 3 个字符)

有时，查询目标字符串中包含 “%” 或 “\_”，这时可以使用换码字符 ESCAPE '\' 将通配符 “%” 或 “\_” 转义为普通字符。

### (5) 涉及空值的查询

空值 NULL 是关系数据库中的特定的概念，表示属性值不确定、未定义。

判断表达式值是否为空值的语法为:

<表达式> IS NULL 或 <表达式> IS NOT NULL

注意：“IS” 不能用 “=” 代替，不能写“列名=NULL”这样的表达式。

## (6) 使用逻辑运算符进行条件查询 (多条件查询)

逻辑运算符

AND: 逻辑“与”运算

OR: 逻辑“或”运算

NOT: 对查询条件进行取反操作

语法格式如下:

WHERE [NOT] <表达式 1> AND|OR <表达式 2>

AND: 当指定的所有查询条件都成立时返回结果集

OR: 当指定的所有查询条件只要有一个成立就返回结果集

NOT: 否定查询条件

运算优先级: NOT 最高, AND 高于 OR, 括号可改变优先级

## 3. 排序查询

ORDER BY 子句

对于使用 SELECT 语句查询出的结果集, 可以用 ORDER BY 子句对结果集按一个或多个属性列进行排序, ASC 代表升序, DESC 代表降序, 缺省值为升序 ASC.

基本语法格式为:

ORDER BY <列名 1>[ASC|DESC],<列名 2>[ASC|DESC]...

说明:

当排序列中含有空值 NULL 时, 显示结果如下:

ASC 按升序排序时排序列为 null 的元组最后显示;

DESC 按降序排序时排序列为 null 的元组最先显示, 相当于系统默认 NULL 是最大的值。

## 4. 聚集函数

### (1) 计数

COUNT ([DISTINCT|ALL] \*) 对表中元组进行计数

COUNT ([DISTINCT|ALL] <列名>) 对表中元组按<列名>值进行计数

(2) 计算总和

SUM ([DISTINCT|ALL] <列名>) 对表中<列名>属性列的值进行求和运算

(3) 计算平均值

AVG ([DISTINCT|ALL] <列名>) 对表中<列名>属性列的值进行求平均值运算

(4) 最大最小值

MAX ([DISTINCT|ALL] <列名>) 对表中<列名>属性列的值进行求最大值运算

MIN ([DISTINCT|ALL] <列名>) 对表中<列名>属性列的值进行求最小值运算

以上聚集函数中， DISTINCT 选项表示相同属性值只参与计算一次；

ALL 表示相同属性值重复参加计算， 缺省值为 ALL。

## 5. 分组查询

GROUP BY 子句：

在 SELECT 语句查询中，可以用 GROUP BY 子句进行分类汇总。 GROUP BY 后面还可以跟 HAVING 短语，用来找出满足条件的分组。

语法格式如下：

GROUP BY <列名 1> [<列名 2>]...

[HAVING <条件表达式>]

分组是按指定的一列或多列值对元组进行分组，指定的一列或多列值相等的元组为一组。分组的作用通常是细化聚集函数的计算对象，作用对象是查询的中间结果，未对中间结果分组，聚集函数将作用于整个中间结果；对中间结果分组后，聚集函数将分别作用于每个组。

注意：

HAVING 短语与 WHERE 子句的区别——二者作用对象不同：

WHERE 子句作用于基本表或视图，从中选择满足条件的元组；

HAVING 短语作用于组，从中选择满足条件的组， HAVING 短语依附于 GROUP BY 短语，不能独立出现在 SELECT 语句中。

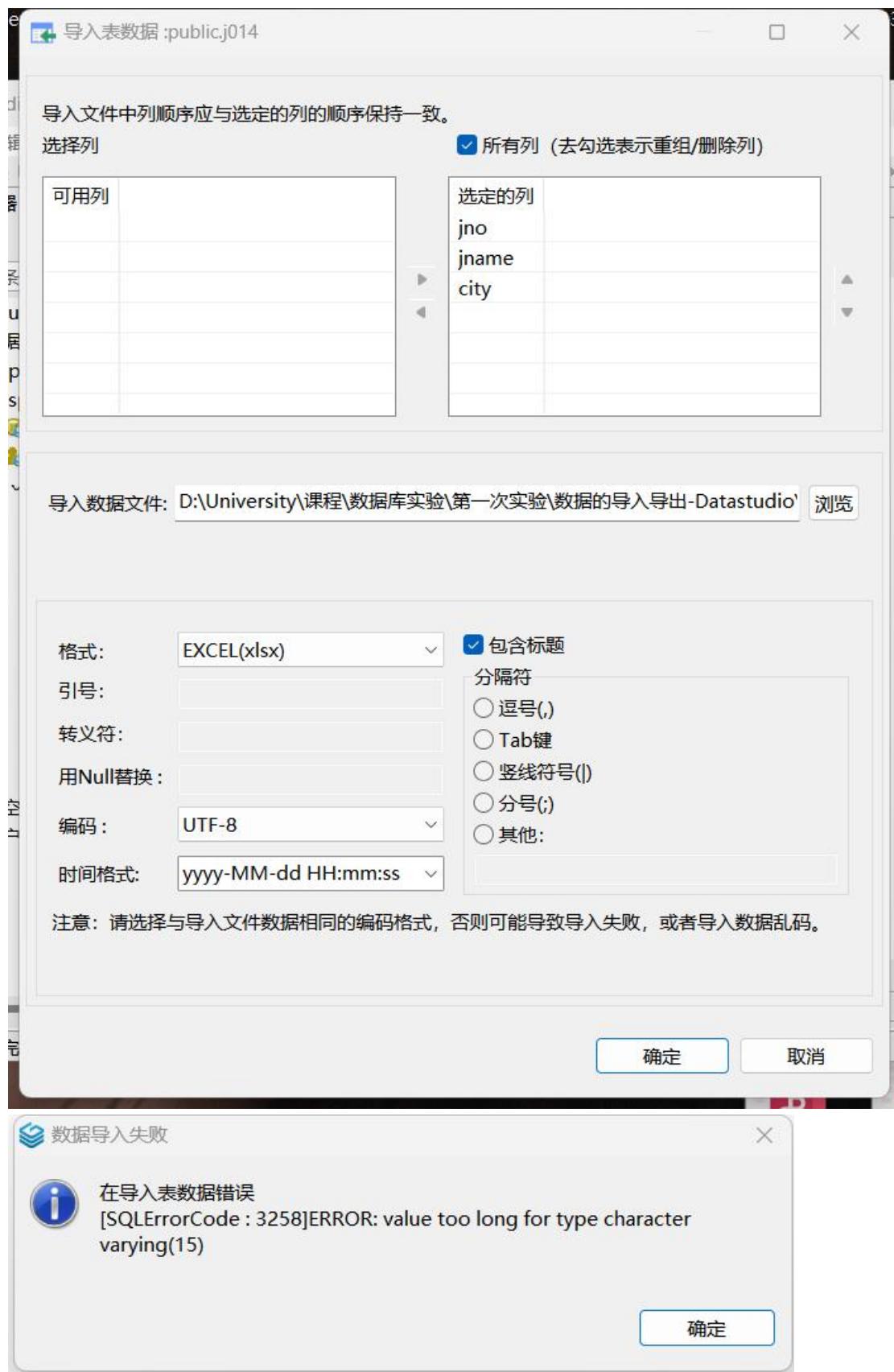
## 【实验内容】

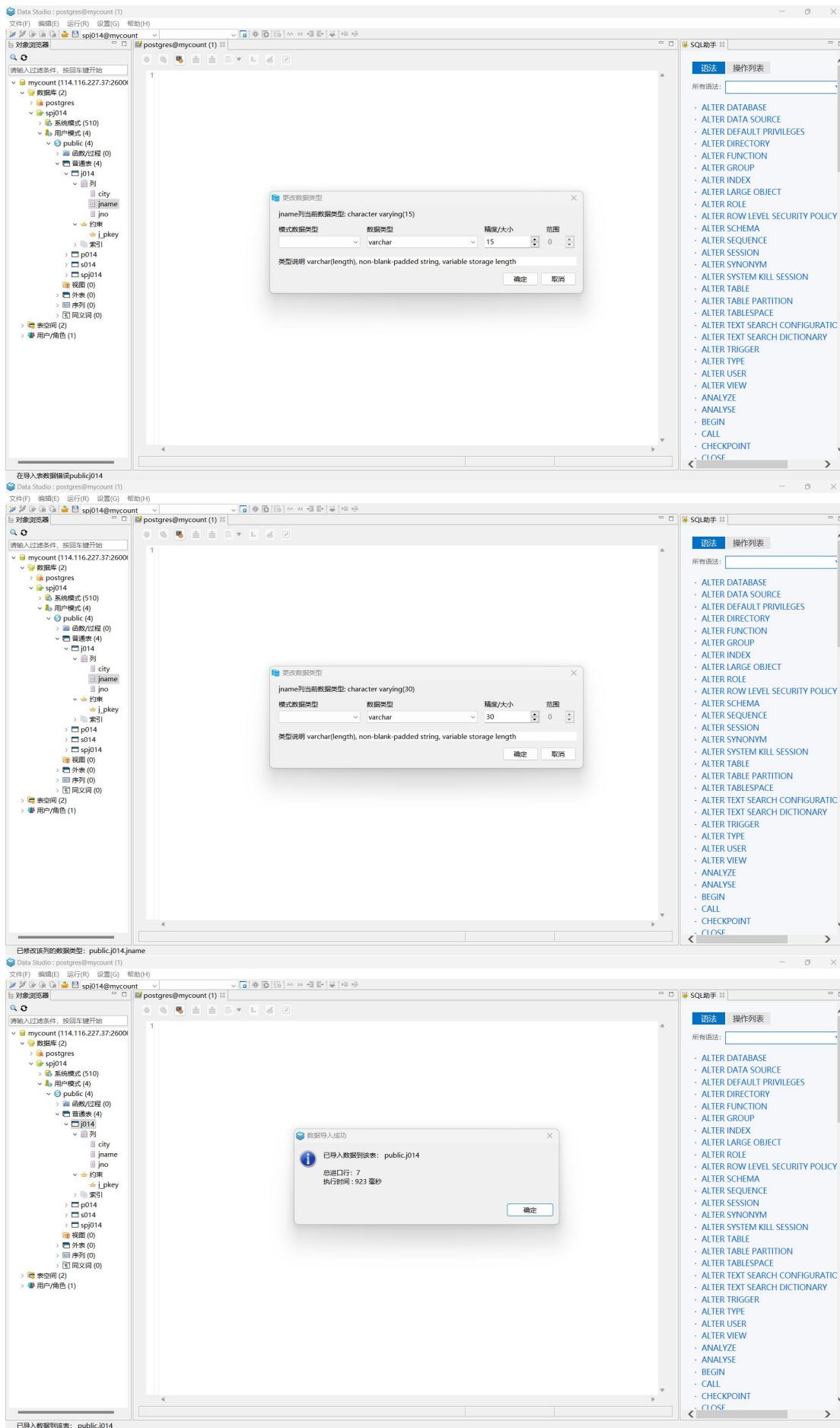
针对供应管理数据库 SPJ，进行以下各种简单查询：

1. 查询所有供应商的信息，用中文表头显示；
2. 查询位于“北京”的名称包含“星”的供应商信息；
3. 查询供应商名中第二个字是“海”的供应商信息；
4. 查询零件名以“螺丝”开头的零件信息；
5. 查询名称含有“车”的工程项目信息；
6. 查询名称为“螺母”、“螺栓”、“螺丝刀”的零件信息；
7. 查询“S001”号供应商的供应情况；
8. 查询“P002”号零件的总供应量；
9. 查询“P002”号零件供应量的最大、最小和平均值；
10. 分组计算每个工程项目使用每种零件的供应量；
11. 查询供应量在 300 以上的供应信息；
12. 查询供应量最低的两个供应信息；
13. 查询供应量前三名的供应商的编号；
14. 分组统计每个供应商供应每种零件的供应量。

要求：每个数据库和表名最后应加上实验者个人学号的末三位（如 SPJ001, S001 等），并在主要截图中体现。

1.





Three screenshots of Data Studio showing the process of importing data from a CSV file into PostgreSQL tables.

**Screenshot 1:** Importing data into public.p014. The SQL Assistant shows the command: `copy p014 from 'D:\我的桌面\p014.csv' with delimiter ',' header;`. A message box indicates "已导入数据到读表: public.p014" with 6 rows imported in 769毫秒.

**Screenshot 2:** Importing data into public.s014. The SQL Assistant shows the command: `copy s014 from 'D:\我的桌面\s014.csv' with delimiter ',' header;`. A message box indicates "已导入数据到读表: public.s014" with 6 rows imported in 826毫秒.

**Screenshot 3:** Importing data into public.cs014. The SQL Assistant shows the command: `copy cs014 from 'D:\我的桌面\cs014.csv' with delimiter ',' header;`. A message box indicates "已导入数据到读表: public.cs014". However, it also displays an error message: "在写入数据时 [SQL ErrorCode: 3649] ERROR: duplicate key value violates unique constraint \"idx\_qty\". Detail: Key (qty)=(100) already exists." The status bar at the bottom right shows "1 运行结束".

Three screenshots of the Data Studio PostgreSQL client interface showing the process of importing data from a CSV file into a PostgreSQL database.

**Screenshot 1:** The interface shows the connection to mycount (114.116.227.37:2600). A modal dialog box displays the message: "已导入数据到表: public.spj014" (Data imported successfully to table: public.spj014), with a total of 19 rows imported and a execution time of 875毫秒 (milliseconds).

**Screenshot 2:** The interface shows the connection to public.p014-spj014@mycount. A table named "spj014" is displayed with the following data:

pno	pname	color	wt
P001	螺母	红	12
P002	螺栓	绿	17
P003	螺丝刀	蓝	14
P004	螺丝刀	红	14
P005	凸轮	蓝	40
P006	齿轮	红	30

**Screenshot 3:** The interface shows the connection to public.p014-spj014@mycount. A table named "spj014" is displayed with the same data as Screenshot 2.

**SQL Assistant:** The right-hand panel shows a list of SQL keywords categorized by type, such as ALTER, CREATE, SELECT, etc.

**SQL语句 (spj014@mycount)**

```

1 SELECT
2   SNO AS "供应商ID",
3   SNAME AS "供应商名称",
4   STAT AS "重要程度",
5   CITY AS "城市"
6 FROM sp014;
7

```

**消息**

(2024-05-30 15:26:58.860 GMT+08:00) [ERROR] 执行失败  
错误代码: 0xSQL语法错误 - 42703  
ERROR: column "status" does not exist  
Position: 105  
Where: referenced column: 重要程度  
Line Number: 6

**查询结果**

供应商ID	供应商名称	重要程度	城市
S001	天津宝贝儿	B	天津
S002	北京启明星	A	北京
S003	北京新天地	C	北京
S004	天津非遗盛	B	天津
S005	上海莘生	C	上海
S006	合肥四达	B	合肥

自动保存完成: Thu May 30 15:25:54 GMT+08:00 2024

**SQL语句 (spj014@mycount)**

```

1 SELECT
2   SNO AS "供应商ID",
3   SNAME AS "供应商名称",
4   STAT AS "重要程度",
5   CITY AS "城市"
6 FROM sp014;
7

```

**消息**

(2024-05-30 15:28:01.816 GMT+08:00) [成功] 执行完成  
无法编辑查询结果。

**查询结果**

供应商ID	供应商名称	重要程度	城市
S001	天津宝贝儿	B	天津
S002	北京启明星	A	北京
S003	北京新天地	C	北京
S004	天津非遗盛	B	天津
S005	上海莘生	C	上海
S006	合肥四达	B	合肥

自动保存完成: Thu May 30 15:25:54 GMT+08:00 2024

导入数据，在导入数据过程中，需要修改 VARCHAR 类型长度，并删除 spj014 表中的部分索引，才能正常导入，并用 SQL 语句实现查询所有供应商的信息，用中文表头显示；在查询过程中发现 STATUS 只有前四位有效，应该写 STAT。

2.

The screenshot shows the Data Studio interface with a query window open. The SQL code is:

```

1 SELECT *
2 FROM s014
3 WHERE CITY = '北京'
4 AND SNAME LIKE '%星%';
    
```

The results table shows one row:

SNO	sname	stat	city
1	北京启明星	A	北京

SQL Assistant pane on the right shows common SELECT statements.

查询位于“北京”的名称包含“星”的供应商信息;

3.

The screenshot shows the Data Studio interface with a query window open. The SQL code is:

```

1 SELECT *
2 FROM s014
3 WHERE SNAME LIKE '_海%';
    
```

The results table shows one row:

SNO	sname	stat	city
1	上海普华	C	上海

SQL Assistant pane on the right shows common ALTER statements.

查询供应商名中第二个字是“海”的供应商信息;

4.

The screenshot shows a Data Studio interface with a SQL query results window. The query is:

```

1 SELECT *
2 FROM p014
3 WHERE PName LIKE '螺丝%';

```

The results table has columns: pno, pname, color, and wt. It contains two rows:

	pno	pname	color	wt
1	P003	螺丝刀	蓝	14
2	P004	螺丝刀	红	14

SQL助手 pane displays the following text:

```

SELECT语句从表或视图中取出数据。
SELECT语句就像添加在数据库表上的过滤器，利用SQL关键字从数据表中过滤出用户需要的数据。
注意事项
• 必须对每个在SELECT命令中使用的字段有SELECT权限。
示例
--先通过子查询得到一张临时表temp_t, 然后查询表temp_t中的所有数据。
postgres=# WITH temp_t ( name
, isdba ) AS ( SELECT username
, use_superuser FROM pg_user ) SELECT * FROM temp_t;
--查询tpcds.reason表的所有r_reason_sk记录, 且去除重复。
postgres=# SELECT DISTINCT ( r_reason_sk ) FROM tpcds.reason;
--LIMIT子句示例, 获取表中一条记录。
。
```

查询零件名以“螺丝”开头的零件信息；

5.

截图展示了Data Studio的界面，正在运行一个名为“spj014@mycount (1) 的会话。左侧是对象浏览器，显示了数据库、表和视图等结构。右侧是SQL编辑器，显示了以下SQL语句：

```
1 SELECT *
2 FROM j014
3 WHERE JNAME LIKE '%车%';
4
```

下方的结果区域显示了运行时间：471 ms，并且有一个结果集窗口，显示了以下数据：

	jno	jname	city
1	1005	唐山机车厂	唐山

右侧有“语法”、“操作”、“功能描述”、“注意事项”、“示例”等辅助信息。

自动保存完成: Thu May 30 15:45:55 GMT+08:00 2024

查询名称含有“车”的工程项目信息；

6.

The screenshot shows the Data Studio interface with the following details:

- Top Bar:** File(F), Edit(E), Run(R), Settings(G), Help(H).
- Left Sidebar:** Object Browser (显示 mycount (1)、数据库 (2)、postg、spj014 (显示 系统、用户))，搜索框。
- Central Area:** SQL Editor (显示 SQL 语句: `SELECT * FROM p014 WHERE PName IN ('螺母', '螺栓', '螺丝刀');`)，Run Time: 384 ms，结果集 (显示表 p014 的数据)。
- Right Sidebar:**
  - 语法 (Syntax):** 所有语法: SELECT
  - 功能描述 (Function Description):** SELECT 用于从表或视图中取出数据。SELECT 语句就像叠加在数据库表上的过滤器，利用 SQL 关键字从数据表中过滤出用户需要的数据。
  - 注意事项 (Attention):**
    - 必须对每个在 SELECT 命令中使用的字段有 SELECT 权限。
  - 示例 (Example):** -- 先通过子查询得到一张临时表 temp\_tt，然后查询

自动保存完成: Thu May 30 15:45:55 GMT+08:00 2024

查询名称为“螺母”、“螺栓”、“螺丝刀”的零件信息;

7.

截图展示了Data Studio的界面，显示了一个名为“spj014@mycount (1) 的连接。在SQL编辑器中输入并运行了以下SQL语句：

```
1 SELECT SNAME, JNAME, PNAME, QTY
2 FROM s014, p014, j014, spj014
3 WHERE s014.SNO=spj014.SNO
4 AND p014.PNO=spj014.PNO
5 AND j014.JNO=spj014.JNO
6 AND s014.SNO='S001';
```

运行时间为 534 ms。结果集显示了4行数据：

	sname	jname	pname	qty
1	天津安贝儿	北京三建	螺母	200
2	天津安贝儿	长春一汽	螺栓	100
3	天津安贝儿	新安弹簧厂	螺母	100
4	天津安贝儿	临江造船厂	螺母	700

右侧有“SQL助手”功能描述和注意事项。

注意事项：

- 必须对每个在SELECT命令中使用的字段有SELECT权限。

示例：

--先通过子查询得到一张临时表temp\_tt, 然后查询

查询“S001”号供应商的供应情况；

8.

The screenshot shows the Data Studio interface with the following details:

- Object Browser:** Shows the database structure under "mycount".
  - Database: mycount (114.116.227.3)
  - PostgreSQL: postgres
  - Schema: spj014
    - System Mode (510)
    - User Mode (4)
      - public (4)
    - Functions/Procedures (0)
    - Tables (4)
      - j014
      - p014
        - Columns
        - Constraints
        - Indexes
    - s014
    - spj014
      - Columns
      - Constraints
      - Indexes
  - SP
  - Views (0)
  - External Tables (0)
  - Sequences (0)
  - Synonyms (0)
  - Tablespaces (2)
  - Users/Roles (1)
- SQL Editor:** Displays the following SQL query:

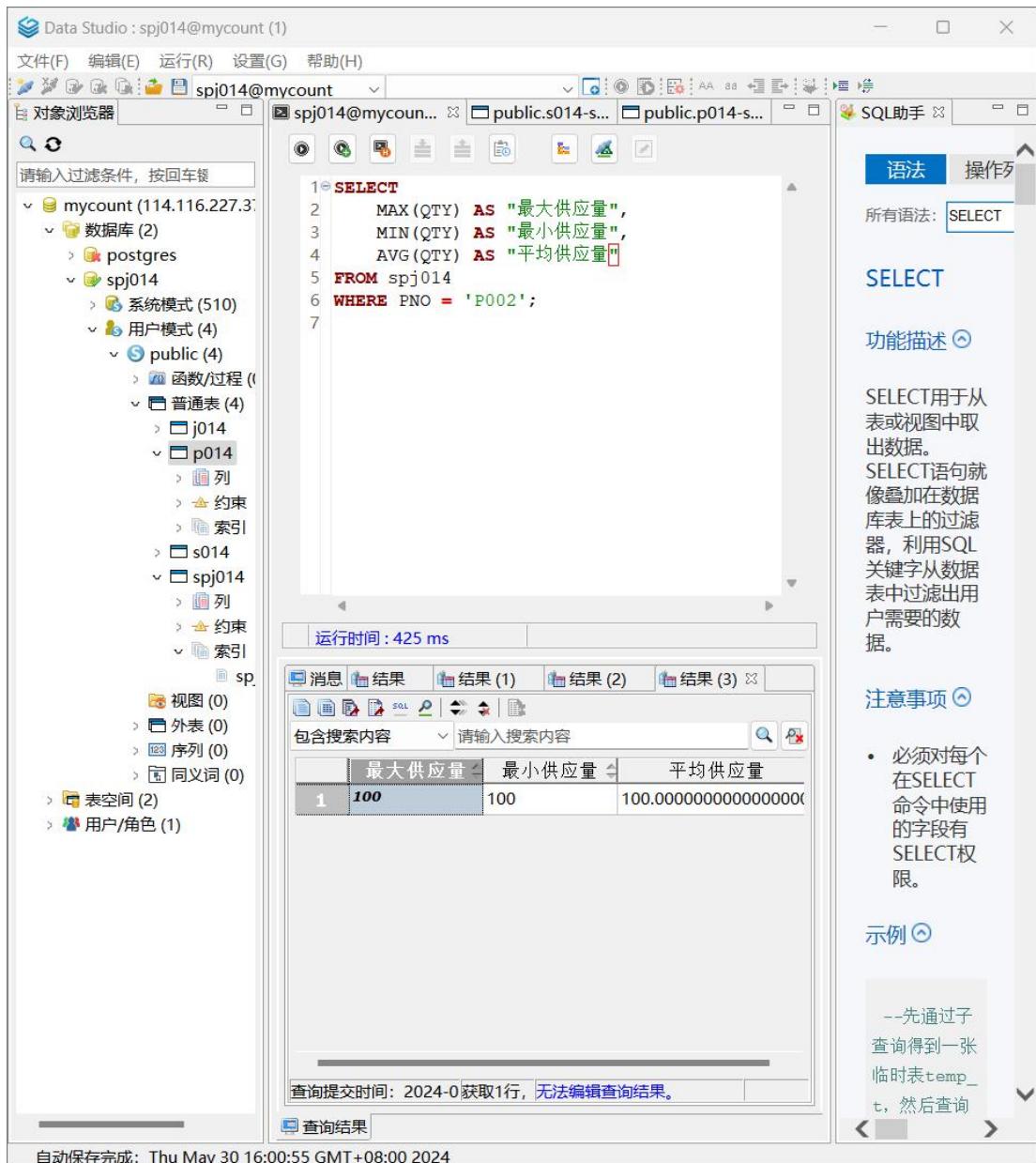
```
1 SELECT
2     SUM(QTY) AS "总供应量"
3 FROM spj014
4 WHERE PNO = 'P002';
```
- Results Panel:** Shows the execution time: 391 ms. The results table contains one row:

	总供应量
1	200
- Message Panel:** States: "查询提交时间: 2024-05-30 15:55:55 GMT+08:00 2024" and "无法编辑查询结果。"
- Right Sidebar:**
  - 语法** tab is selected.
  - 所有语法:** SELECT
  - 功能描述:** SELECT  
SELECT用于从表或视图中取出数据。SELECT语句就像叠加在数据库表上的过滤器，利用SQL关键字从数据表中过滤出用户需要的数据。
  - 注意事项:**
    - 必须对每个在SELECT命令中使用的字段有SELECT权限。
  - 示例:**

--先通过子查询得到一张临时表temp\_t, 然后查询

查询“P002”号零件的总供应量；

9.



查询“P002”号零件供应量的最大、最小和平均值；

10.

The screenshot shows the Data Studio interface with a database connection to spj014@mycount. In the left sidebar, the schema tree is visible, including the public schema which contains functions, tables, and other objects. The main area displays a SQL query:

```
1: SELECT JNO, PNO, SUM(QTY) AS "总供应量"
2: FROM spj014
3: GROUP BY JNO, PNO;
```

The results pane shows the output of the query:

	JNO	PNO	总供应量
1	J004	P002	100
2	J001	P001	400
3	J002	P002	100
4	J005	P003	400
5	J004	P006	700
6	J001	P005	500
7	J004	P003	500
8	J002	P006	200
9	J003	P001	100
10	J003	P006	300
11	J004	P001	700
12	J001	P003	800
13	J002	P005	100
14	J002	P003	200

The SQL helper panel on the right lists various ALTER commands.

分组计算每个工程项目使用每种零件的供应量;

11.

The screenshot shows the Data Studio interface with a database connection to spj014@mycount. The schema tree is visible, including the public schema. The main area displays a SQL query:

```
1: SELECT *
2: FROM spj014
3: WHERE QTY>300;
```

The results pane shows the output of the query:

	SNO	PNO	JNO	QTY
1	S001	P001	J004	700
2	S002	P003	J001	400
3	S002	P003	J004	500
4	S002	P003	J005	400
5	S002	P005	J001	400
6	S005	P006	J004	500

The SQL helper panel on the right lists various SELECT-related commands.

查询供应量在 300 以上的供应信息;

12.

The screenshot shows the Data Studio interface with a database connection to spj014@mycount. The SQL editor contains the following query:

```

1 SELECT *
2 FROM spj014
3 ORDER BY QTY ASC
4 LIMIT 2;
    
```

The results pane displays two rows of data:

SNO	PNO	JNO	QTY	
1	S001	P001	J003	100
2	S001	P002	J002	100

SQL助手 panel shows the 'SELECT' statement under the '语法' tab.

查询供应量最低的两个供应信息;

13.

The screenshot shows the Data Studio interface with a database connection to spj014@mycount. The SQL editor contains the following query:

```

1 SELECT SNO
2 FROM spj014
3 GROUP BY SNO
4 ORDER BY SUM(QTY) DESC
5 LIMIT 3;
    
```

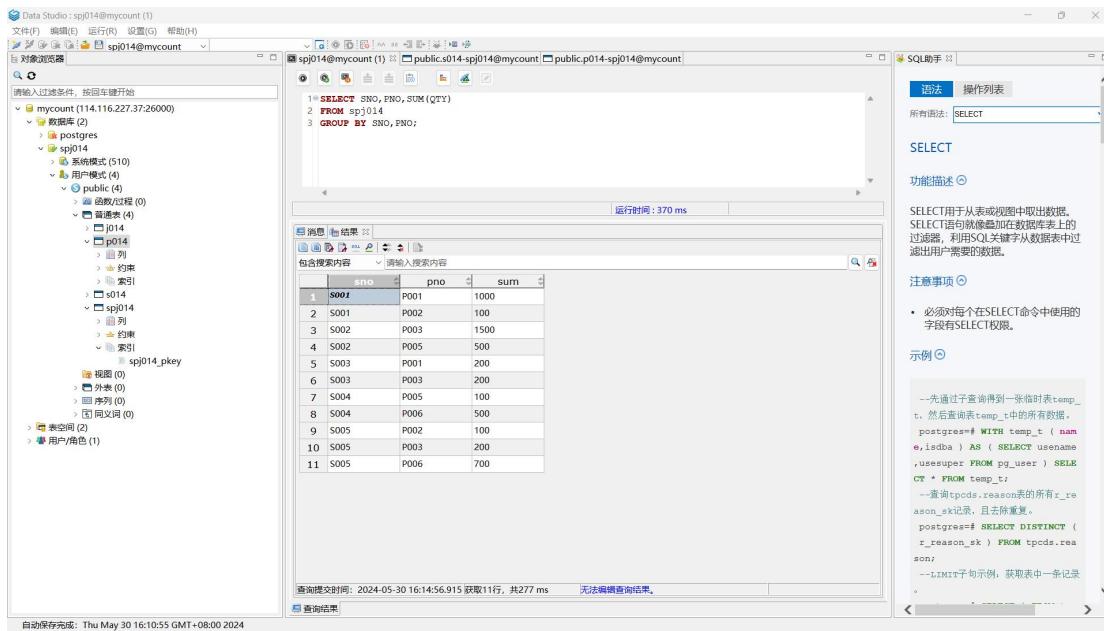
The results pane displays three rows of data:

SNO	
1	S002
2	S001
3	S005

SQL助手 panel shows the 'SELECT' statement under the '语法' tab.

查询供应量前三名的供应商的编号;

14.



分组统计每个供应商供应每种零件的供应量。

## 【小结或讨论】

这次实验主要围绕 SQL 查询语句的应用展开，特别是 SELECT 语句的多种用法。通过本次实验，我对 SQL 语句的结构有了更深入的理解，包括如何选择特定的列和元组、如何使用聚集函数以及如何进行数据排序和分组。实验中的具体任务让我有机会实际操作和探索各种 SQL 功能，比如如何通过 WHERE 子句设置复杂的查询条件，以及如何运用 GROUP BY 和 ORDER BY 子句进行数据的分组和排序。

在完成各种基于供应管理数据库的查询任务中，我不仅复习了 SQL 基本语法，还了解到了在实际数据库操作中可能遇到的一些问题，如数据类型调整和索引问题的处理。例如，导入数据时需要调整 VARCHAR 类型长度，并删除部分索引以确保数据可以正常导入。此外，通过构建具体的查询语句来获取供应商信息、零件供应情况和工程项目数据等，我能更清楚地看到数据查询在实际应用中的重要性和实用性。

通过这些操作，我加深了对 SQL 查询优化的理解，并认识到了在设计查询时如何有效地使用聚集函数和排序、分组子句来提高数据处理的效率和准确性。总的来说，这次实验不仅增强了我对 SQL 查询命令的掌握，也提高了我解决实际问题的能力。