南京信息工程大学数据库系统原理实验（实习）报告

实验（实习）名称 单表查询 实验（实习）日期 2025-4-18 得分 指导教师
系 计算机学院 专业 计科

一、实验目的

1. 掌握 SELECT 语句的基本语法
2. 掌握 SELECT 子句的目标列构建方法
3. 掌握 SELECT 查询条件表示方法
4. 掌握聚合函数的作用和使用方法
5. 掌握 ORDER BY 子句的作用和使用方法
6. 掌握 GROUP BY 子句的作用和使用方法
7. 掌握 HAVING 子句的作用和使用方法

二、实验内容

在 GoodsOrder 数据库中，用 SQL 语句实现如下查询：

1、查询在 1990 年 1 月 1 日~1995 年 12 月 31 日之间出生的客户信息。
2、查询“上海、江苏南京、河南郑州”所有的客户信息。
3、查询商品类别为“食品”的所有商品信息。
4、查询“健康粮食生产基地”所生产的商品信息。
5、查询商品名称中包含“球”的所有商品信息。
6、查询 微信号 形如“wxid_”的客户情况。
7、查询备注不为空的客户信息。
8、查询保质期在“2020-02-19”及以后的食品类商品信息。
9、查询食品类或文具类、库存量不少于 50 的商品信息。
10、将食品类商品信息按价格降序排列。
11、查询客户最早和最晚出生日期。
12、查询每个客户的年龄。
13、查询客户最大和最小年龄。
14、查询客户的平均年龄。
15、查询订单信息中订购数量的最大值和最小值。
16、统计食品类商品的总库存量。
17、查询食品类的商品数。
18、查询订购了商品的客户总数。
19、统计各地的客户数量。
20、按性别统计江苏的客户数量。
21、按付款方式统计订单数量不少于 2 笔的付款方式及订单数。
22、查询商品类别为“食品”或“服装”，且单价在 20 元（含）以上的商品的编号和名称。
23、查询订购时间在“2020-2-19”和“2020-2-20”之间，且送货方式为“送货上门”的订单信息。
24、查询每个客户订购商品的平均数量。
25、查找 1980 年以后出生的客户数不少于 2 个的省市。
26、查找总库存量超过 100 件的商品类别及其总库存量。

三、实验过程与结果

1、查询在 1990 年 1 月 1 日~1995 年 12 月 31 日之间出生的客户信息。

- 分析：查询结果信息来自 CustomerInfo，查询条件只涉及 CustomerInfo，故为单表查询；目标列为 CustomerInfo 表的全部列，WHERE 子句的条件使用指定范围谓词 BETWEEN...AND....。
- 设计的 SQL 语句如下：

```
SELECT *
    FROM CustomerInfo
    WHERE 出生日期 BETWEEN '1990-01-01' AND '1999-12-31'
```

- 语句执行结果如下：

	客户编号	客户姓名	出生日期	性别	所在省市	联系电话	微信号	VIP	备注
1	100002	李红红	1991-03-22 00:00:00.000	女	江苏苏州	13908899120	13908899120	1	金牌客户
2	100004	赵明	1992-03-28 00:00:00.000	男	河南郑州	13809900118	NULL	0	新客户
3	100005	张帆一	1990-08-10 00:00:00.000	男	山东烟台	13880933201	NULL	0	NULL

- 进一步的思考与练习：

（1）目标列的构成：在语句中写出结果需要的列，如下：

```
SELECT 客户编号,客户姓名,出生日期,性别,所在省市,联系电话,微信号,VIP,备注
    FROM CustomerInfo
    WHERE 出生日期 BETWEEN '1990-01-01' AND '1999-12-31'
```

	客户编号	客户姓名	出生日期	性别	所在省市	联系电话	微信号	VIP	备注
1	100002	李红红	1991-03-22 00:00:00.000	女	江苏苏州	13908899120	13908899120	1	金牌客户
2	100004	赵明	1992-03-28 00:00:00.000	男	河南郑州	13809900118	NULL	0	新客户
3	100005	张帆一	1990-08-10 00:00:00.000	男	山东烟台	13880933201	NULL	0	NULL

（2）WHERE 子句的条件也可使用比较运算，如下：

```
SELECT *
    FROM CustomerInfo
    WHERE 出生日期>= '1990-01-01'  AND 出生日期<= '1999-12-31'
```

	客户编号	客户姓名	出生日期	性别	所在省市	联系电话	微信号	VIP	备注
1	100002	李红红	1991-03-22 00:00:00.000	女	江苏苏州	13908899120	13908899120	1	金牌客户
2	100004	赵明	1992-03-28 00:00:00.000	男	河南郑州	13809900118	NULL	0	新客户
3	100005	张帆一	1990-08-10 00:00:00.000	男	山东烟台	13880933201	NULL	0	NULL

2、查询“上海市、江苏南京、河南郑州”所有的客户信息。

- 分析：查询结果信息来自 CustomerInfo，查询条件只涉及 CustomerInfo，故为单表查询；

目标列为 CustomerInfo 表的全部列，WHERE 子句使用 IN 关键字查找列值属于指定集合的所有行。

- 设计的 SQL 语句：

```
SELECT *
    FROM CustomerInfo
    WHERE  所在省市  IN ('上海市','江苏南京','河南郑州')
```

- 语句执行结果如下：

```
SELECT *
    FROM CustomerInfo
    WHERE 所在省市 IN ('上海市','江苏南京','河南郑州')
```

结果 消息

	客户编号	客户姓名	出生日期	性别	所在省市	联系电话	微信号	VIP	备注
1	100001	张小林	1982-02-01 00:00:00.000	男	江苏南京	02581334567	13980030075	1	银牌客户
2	100003	王晓美	1986-08-20 00:00:00.000	女	上海市	02166552101	wxid_0021001	0	新客户
3	100004	赵明	1992-03-28 00:00:00.000	男	河南郑州	13809900118	NULL	0	新客户
4	100006	王芳芳	1996-05-01 00:00:00.000	女	江苏南京	13709092011	wxid_7890921	0	NULL

- 进一步的思考与练习：

（1）WHERE 条件子句的条件也可以使用比较运算结合逻辑运算，如下：

```
SELECT *
    FROM CustomerInfo
    WHERE 所在省市 = '上海市' OR 所在省市= '江苏南京' OR 所在省市 = '河南郑州'
```

结果 消息

	客户编号	客户姓名	出生日期	性别	所在省市	联系电话	微信号	VIP	备注
1	100001	张小林	1982-02-01 00:00:00.000	男	江苏南京	02581334567	13980030075	1	银牌客户
2	100003	王晓美	1986-08-20 00:00:00.000	女	上海市	02166552101	wxid_0021001	0	新客户
3	100004	赵明	1992-03-28 00:00:00.000	男	河南郑州	13809900118	NULL	0	新客户
4	100006	王芳芳	1996-05-01 00:00:00.000	女	江苏南京	13709092011	wxid_7890921	0	NULL

3、查询商品类别为“食品”的所有商品信息。

- 分析：查询结果信息来自 GoodsInfo，查询条件只涉及 GoodsInfo，故为单表查询；目标列为 GoodsInfo 表的全部列，WHERE 子句使用比较运算符“=”。
- 设计的 SQL 语句：

```
SELECT *
    FROM GoodsInfo
    WHERE  商品类别 ='食品'
```

- 语句执行结果如下：

```
SELECT *
    FROM GoodsInfo
    WHERE 商品类别 = '食品'
```

结果 消息

	商品编号	商品类别	商品名称	品牌	单价	生产商	保质期	库存量	备注
1	10010001	食品	咖啡	宇一	50	宇一饮料公司	2021-08-31 00:00:00.000	100	NULL
2	10010002	食品	苹果汁	宇一	5.2	宇一饮料公司	2020-12-31 00:00:00.000	500	NULL
3	10020001	食品	大米	健康	35	健康粮食生产基地	2020-12-20 00:00:00.000	100	NULL
4	10020002	食品	面粉	健康	18	健康粮食生产基地	2021-01-20 00:00:00.000	20	NULL

- 进一步的思考与练习：

（1）查询商品类别不是“食品”的所有商品信息，WHERE 子句使用比较运算符“!=”，如下：

```
SELECT *
    FROM GoodsInfo
    WHERE  商品类别  !=   '食品'
```

	商品编号	商品类别	商品名称	品牌	单价	生产商	保质期	库存量	备注
1	20180001	服装	运动服	天天	200	天天服饰公司	2000-01-01 00:00:00.000	5	有断码
2	20180002	服装	T恤	天天	120	天天服饰公司	2000-01-01 00:00:00.000	10	NULL
3	30010001	文具	签字笔	新新	3.5	新新文化用品制造厂	2000-01-01 00:00:00.000	100	NULL
4	30010002	文具	文件夹	新新	5.6	新新文化用品制造厂	2000-01-01 00:00:00.000	50	NULL
5	40010001	图书	营养菜谱	新华	38	新华图书出版公司	2000-01-01 00:00:00.000	12	NULL
6	40010002	图书	豆浆的做法	新华	20	新华图书出版公司	2000-01-01 00:00:00.000	20	NULL
7	50020001	体育用品	羽毛球拍	美好	30	美好体育用品公司	2000-01-01 00:00:00.000	30	NULL
8	50020002	体育用品	篮球	美好	80	美好体育用品公司	2000-01-01 00:00:00.000	20	NULL
9	50020003	体育用品	足球	美好	65	美好体育用品公司	2000-01-01 00:00:00.000	20	NULL

4、查询“健康粮食生产基地”所生产的商品信息。

- 分析：查询结果信息来自 GoodsInfo，查询条件只涉及 GoodsInfo，故为单表查询；目标列为 GoodsInfo 表的全部列，WHERE 子句的条件使用比较运算符“=”。
- 设计的 SQL 语句：

```
SELECT *
    FROM GoodsInfo
    WHERE  生产商  ='健康粮食生产基地'
```

- 语句执行结果如下：

	商品编号	商品类别	商品名称	品牌	单价	生产商	保质期	库存量	备注
1	10020001	食品	大米	健康	35	健康粮食生产基地	2020-12-20 00:00:00.000	100	NULL
2	10020002	食品	面粉	健康	18	健康粮食生产基地	2021-01-20 00:00:00.000	20	NULL

- 进一步的思考与练习：

（1）查询“健康粮食生产基地”所生产的商品中库存量大于等于 50 的商品信息，WHERE 子句使用比较运算符，如下：

```
SELECT *
    FROM GoodsInfo
    WHERE  生产商  ='健康粮食生产基地' AND  库存量  >= '50'
```

	商品编号	商品类别	商品名称	品牌	单价	生产商	保质期	库存量	备注
1	10020001	食品	大米	健康	35	健康粮食生产基地	2020-12-20 00:00:00.000	100	NULL

5、查询商品名称中包含“球”的所有商品信息。

- 分析：查询结果信息来自 GoodsInfo，查询条件只涉及 GoodsInfo，故为单表查询；目标列为 GoodsInfo 表的全部列，使用模糊查询，WHERE 子句的条件使用通配符“%”。
- 设计的 SQL 语句：

```
SELECT *
    FROM GoodsInfo
    WHERE  商品名称  LIKE '%球%'
```

- 语句执行结果如下：

	商品编号	商品类别	商品名称	品牌	单价	生产商	保质期	库存量	备注
1	50020001	体育用品	羽毛球拍	美好	30	美好体育用品公司	2000-01-01 00:00:00.000	30	NULL
2	50020002	体育用品	篮球	美好	80	美好体育用品公司	2000-01-01 00:00:00.000	20	NULL
3	50020003	体育用品	足球	美好	65	美好体育用品公司	2000-01-01 00:00:00.000	20	NULL

- 进一步的思考与练习：

（1）查询商品名称形如“xx 球”的所有商品信息，WHERE 子句的条件使用通配符“%”，如下：

```
SELECT *
    FROM GoodsInfo
    WHERE  商品名称  LIKE '%球'
```

	商品编号	商品类别	商品名称	品牌	单价	生产商	保质期	库存量	备注
1	50020002	体育用品	篮球	美好	80	美好体育用品公司	2000-01-01 00:00:00.000	20	NULL
2	50020003	体育用品	足球	美好	65	美好体育用品公司	2000-01-01 00:00:00.000	20	NULL

6、查询微信号形如“wxid_”的客户情况

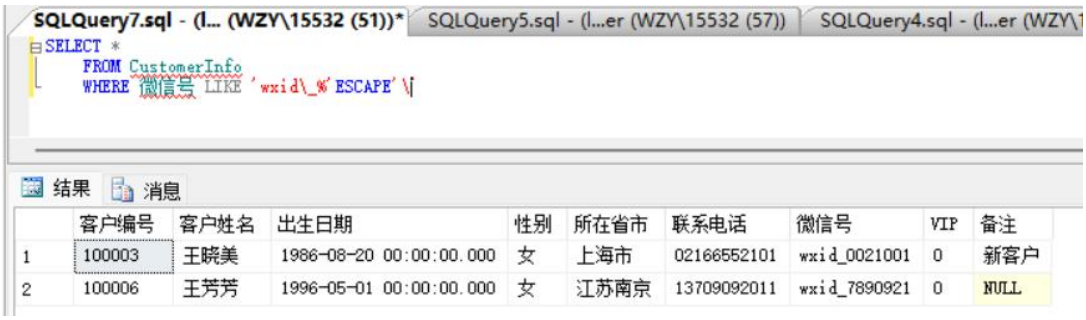
- 分析：查询结果信息来自 CustomerInfo，查询条件只涉及 CustomerInfo，故为单表查询；目标列为 CustomerInfo 表的全部列，使用模糊查询，WHERE 子句的条件使用通配符“%”，其中查询条件含有通配符“_”，故需要使用转义序列“ESCAPE\”对通配符进行转义。
- 设计的 SQL 语句：

```
SELECT *
```

FROM CustomerInfo

WHERE 微信号 LIKE 'wxid_%' ESCAPE '\'

- 语句执行结果如下：



	客户编号	客户姓名	出生日期	性别	所在省市	联系电话	微信号	VIP	备注
1	100003	王晓美	1986-08-20 00:00:00.000	女	上海市	02166552101	wxid_0021001	0	新客户
2	100006	王芳芳	1996-05-01 00:00:00.000	女	江苏南京	13709092011	wxid_7890921	0	NULL

7、查询备注不为空的客户信息。

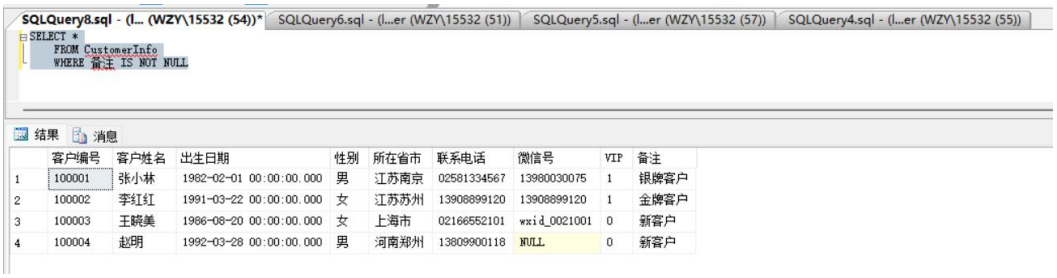
- 分析：查询结果信息来自 CustomerInfo，查询条件只涉及 CustomerInfo，故为单表查询；目标列为 CustomerInfo 表的全部列，判断表达式的值是否不为空值，WHERE 子句的条件使用 IS NOT NULL。
- 设计的 SQL 语句：

SELECT *

FROM CustomerInfo

WHERE 备注 IS NOT NULL

- 语句执行结果如下：



	客户编号	客户姓名	出生日期	性别	所在省市	联系电话	微信号	VIP	备注
1	100001	张小林	1982-02-01 00:00:00.000	男	江苏南京	02581334567	13980030075	1	银牌客户
2	100002	李红红	1991-03-22 00:00:00.000	女	江苏苏州	13908899120	13908899120	1	金牌客户
3	100003	王晓美	1986-08-20 00:00:00.000	女	上海市	02166552101	wxid_0021001	0	新客户
4	100004	赵明	1992-03-28 00:00:00.000	男	河南郑州	13809900118	NULL	0	新客户

- 进一步的思考与练习：

（1）查询备注不为空的客户信息，WHERE 子句使用 IS NULL，如下：

SELECT *

FROM CustomerInfo

WHERE 备注 IS NULL

	客户编号	客户姓名	出生日期	性别	所在省市	联系电话	微信号	VIP	备注
1	100005	张帆一	1990-08-10 00:00:00.000	男	山东烟台	13880933201	NULL	0	NULL
2	100006	王芳芳	1996-05-01 00:00:00.000	女	江苏南京	13709092011	wxid_7890921	0	NULL

8、查询保质期在“2020-02-19”及以后的食品类商品信息。

- 分析：查询结果信息来自 GoodsInfo，查询条件只涉及 GoodsInfo，故为单表查询；

目标列为 GoodsInfo 表的全部列，查询保质期以后的属于食品类的商品信息，WHERE 子句的条件使用比较运算“>=”和逻辑运算“AND”。

- 设计的 SQL 语句：

```
SELECT *
    FROM GoodsInfo
    WHERE 商品类别 ='食品' AND 保质期 >='2020-02-19'
```

- 语句执行结果如下：

	商品编号	商品类别	商品名称	品牌	单价	生产商	保质期	库存量	备注
1	10010001	食品	咖啡	宇一	50	宇一饮料公司	2021-08-31 00:00:00.000	100	NULL
2	10010002	食品	苹果汁	宇一	5.2	宇一饮料公司	2020-12-31 00:00:00.000	500	NULL
3	10020001	食品	大米	健康	35	健康粮食生产基地	2020-12-20 00:00:00.000	100	NULL
4	10020002	食品	面粉	健康	18	健康粮食生产基地	2021-01-20 00:00:00.000	20	NULL

9、查询食品类或文具类、库存量不少于 50 的商品信息。

- 分析：查询结果信息来自 GoodsInfo，查询条件只涉及 GoodsInfo，故为单表查询；

目标列为 GoodsInfo 表的全部列，查询商品类别为食品类或者文具类，且库存量大于等于 50 的商品信息，WHERE 子句的条件使用比较运算“>=”、逻辑运算“AND”和集合运算符“IN”。

- 设计的 SQL 语句：

```
SELECT *
    FROM GoodsInfo
    WHERE 商品类别 IN ('食品','文具') AND 库存量 >= 50
```

- 语句执行结果如下：

	商品编号	商品类别	商品名称	品牌	单价	生产商	保质期	库存量	备注
1	10010001	食品	咖啡	宇一	50	宇一饮料公司	2021-08-31 00:00:00.000	100	NULL
2	10010002	食品	苹果汁	宇一	5.2	宇一饮料公司	2020-12-31 00:00:00.000	500	NULL
3	10020001	食品	大米	健康	35	健康粮食生产基地	2020-12-20 00:00:00.000	100	NULL
4	30010001	文具	签字笔	新新	3.5	新新文化用品制造厂	2000-01-01 00:00:00.000	100	NULL
5	30010002	文具	文件夹	新新	5.6	新新文化用品制造厂	2000-01-01 00:00:00.000	50	NULL

- 进一步的思考与练习：

（1）查询食品类或文具类、库存量少于 50 的商品信息，如下：

```
SELECT *
    FROM GoodsInfo
    WHERE 商品类别 IN ('食品','文具') AND 库存量 < 50
```

	商品编号	商品类别	商品名称	品牌	单价	生产商	保质期	库存量	备注
1	10020002	食品	面粉	健康	18	健康粮食生产基地	2021-01-20 00:00:00.000	20	NULL

10、将食品类商品信息按价格降序排列。

- 分析：查询结果信息来自 GoodsInfo，查询条件只涉及 GoodsInfo，故为单表查询；目标列为 GoodsInfo 表的全部列，先查询商品类别为食品类的商品，WHERE 子句的条件使用比较运算“=”；再将商品价格降序排列，ORDER BY 子句使用“DESC”。
- 设计的 SQL 语句：

```
SELECT *
    FROM GoodsInfo
    WHERE  商品类别 ='食品'
    ORDER BY  单价  DESC
```

- 语句执行结果如下：

	商品编号	商品类别	商品名称	品牌	单价	生产商	保质期	库存量	备注
1	10010001	食品	咖啡	宇一	50	宇一饮料公司	2021-08-31 00:00:00.000	100	NULL
2	10020001	食品	大米	健康	35	健康粮食生产基地	2020-12-20 00:00:00.000	100	NULL
3	10020002	食品	面粉	健康	18	健康粮食生产基地	2021-01-20 00:00:00.000	20	NULL
4	10010002	食品	苹果汁	宇一	5.2	宇一饮料公司	2020-12-31 00:00:00.000	500	NULL

- 进一步的思考与练习：

（1）将食品类商品信息按价格降序排列，ORDER BY 单价，默认升序排列

```
SELECT *
    FROM GoodsInfo
    WHERE  商品类别 ='食品'
    ORDER BY  单价
```

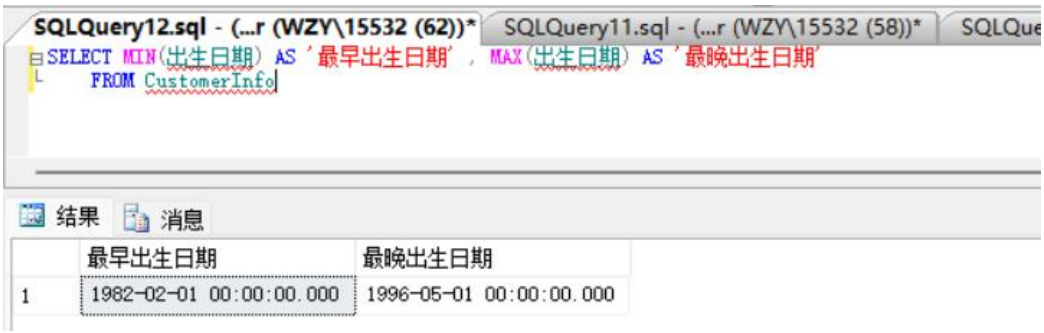
	商品编号	商品类别	商品名称	品牌	单价	生产商	保质期	库存量	备注
1	10010002	食品	苹果汁	宇一	5.2	宇一饮料公司	2020-12-31 00:00:00.000	500	NULL
2	10020002	食品	面粉	健康	18	健康粮食生产基地	2021-01-20 00:00:00.000	20	NULL
3	10020001	食品	大米	健康	35	健康粮食生产基地	2020-12-20 00:00:00.000	100	NULL
4	10010001	食品	咖啡	宇一	50	宇一饮料公司	2021-08-31 00:00:00.000	100	NULL

11、查询客户最早和最晚出生日期。

- 分析：查询结果信息来自 CustomerInfo，查询条件只涉及 CustomerInfo，故为单表查询；目标为客户的最早出生日期和最晚出生日期，根据出生日期特性，最小值为最早出生日期，最大值为最晚出生日期，在 SELECT 语句中书写。
- 设计的 SQL 语句：

```
SELECT MIN(出生日期) AS '最早出生日期' , MAX(出生日期) AS '最晚出生日期'
    FROM CustomerInfo
```

- 语句执行结果如下：

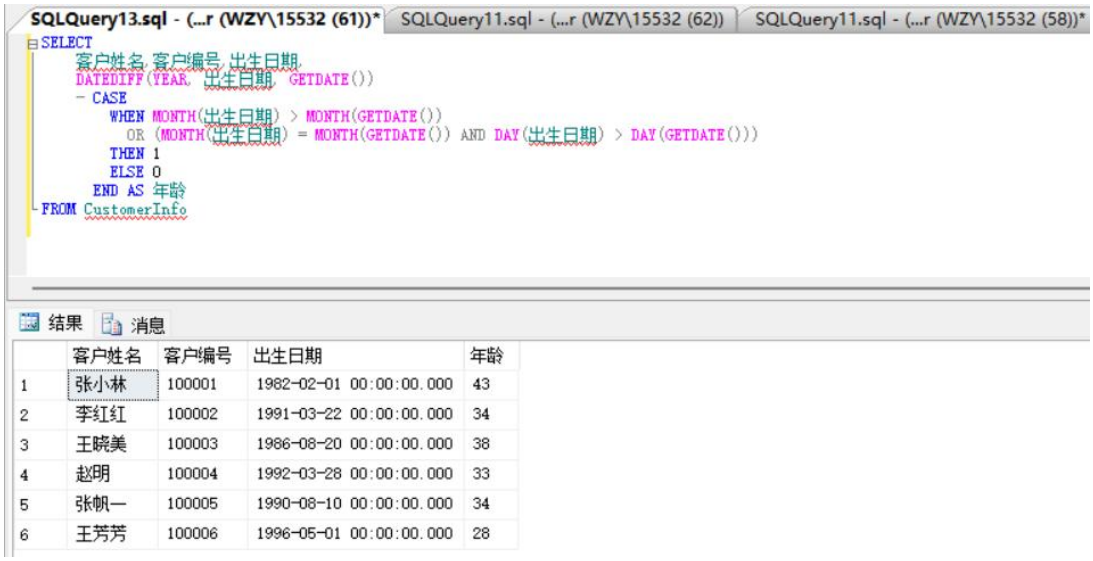


12、查询每个客户的年龄。

- 分析：查询结果信息来自 CustomerInfo，查询条件只涉及 CustomerInfo，故为单表查询；目标列为 GoodsInfo 表的客户姓名、客户编号、出生日期，年龄，根据 DATEDIFF()函数计算年龄，GETDATE()函数获取当前年份，算出的年龄作为新的列值在 SELECT 语句中书写。
- 设计的 SQL 语句：

```
SELECT
    客户姓名,客户编号,出生日期,
    DATEDIFF(YEAR,  出生日期, GETDATE())
    - CASE
        WHEN MONTH(出生日期) > MONTH(GETDATE())
          OR (MONTH(出生日期) = MONTH(GETDATE()) AND DAY(出生日期) > DAY(GETDATE()))
        THEN 1
        ELSE 0
      END AS  年龄
FROM CustomerInfo
```

- 语句执行结果如下：



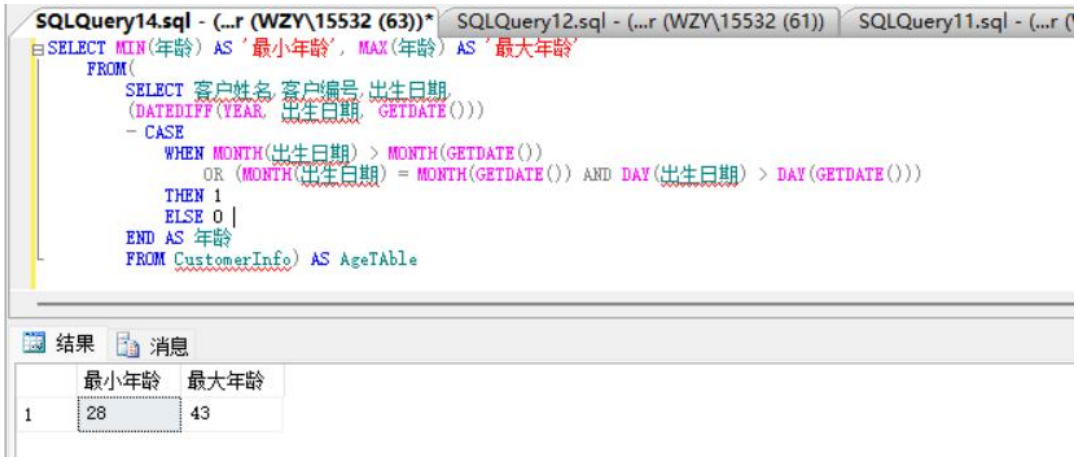
	客户姓名	客户编号	出生日期	年龄
1	张小林	100001	1982-02-01 00:00:00.000	43
2	李红红	100002	1991-03-22 00:00:00.000	34
3	王晓美	100003	1986-08-20 00:00:00.000	38
4	赵明	100004	1992-03-28 00:00:00.000	33
5	张帆一	100005	1990-08-10 00:00:00.000	34
6	王芳芳	100006	1996-05-01 00:00:00.000	28

13、查询客户最大和最小年龄。

- 分析：再上一个问题的基础上在将上一个 SELECT...FROM...命名为 AgeTable，在外嵌套一个 SELECT 语句，使用 MIN()函数和 MAX()函数。
- 设计的 SQL 语句：

```
SELECT MIN(年龄) AS '最小年龄', MAX(年龄) AS '最大年龄'
    FROM(
        SELECT 客户姓名,客户编号,出生日期,(DATEDIFF(YEAR, 出生日期, GETDATE()))
    - CASE
        WHEN MONTH(出生日期) > MONTH(GETDATE())
        OR (MONTH(出生日期) = MONTH(GETDATE()) AND DAY(出生日期) > DAY(GETDATE()))
            THEN 1
            ELSE 0
END AS 年龄
    FROM CustomerInfo) AS AgeTAble
```

- 语句执行结果如下：



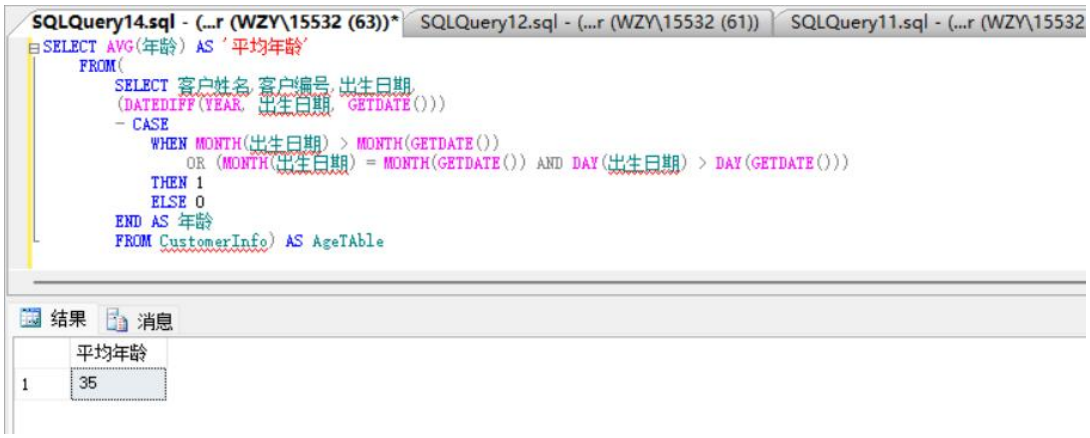
	最小年龄	最大年龄
1	28	43

14、查询客户的平均年龄。

- 分析：与上一个问题类似，换用 AVG 函数即可。
- 设计的 SQL 语句：

```
SELECT AVG(年龄) AS '平均年龄'
    FROM(
        SELECT  客户姓名,客户编号,出生日期,
        (DATEDIFF(YEAR,  出生日期, GETDATE()))
- CASE
WHEN MONTH(出生日期) > MONTH(GETDATE())
OR (MONTH(出生日期) = MONTH(GETDATE()) AND DAY(出生日期) > DAY(GETDATE()))
    THEN 1
    ELSE 0
    END AS  年龄
FROM CustomerInfo) AS AgeTAble
```

- 语句执行结果如下：



15、查询订单信息中订购数量的最大值和最小值。

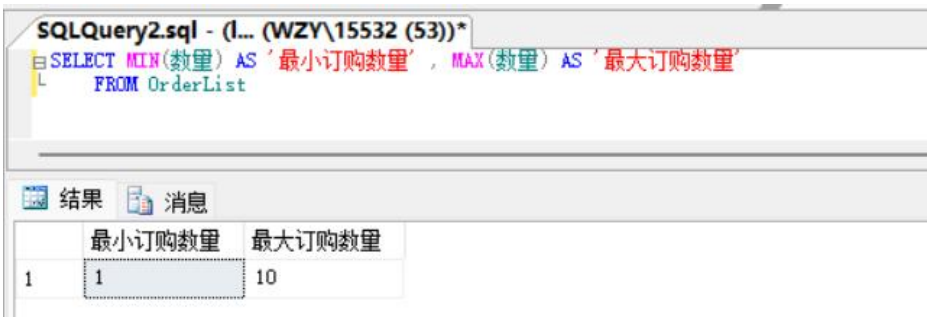
- 分析：查询结果信息来自 OrderList，查询条件只涉及 OrderList，故为单表查询；

目标列为 OrderList 表中订购数量的最大值和最小值，在 SELECT 语句中用 MIN()和 MAX()。

- 设计的 SQL 语句：

```
SELECT MIN(数量) AS '最小订购数量' , MAX(数量) AS '最大订购数量'
    FROM OrderList
```

- 语句执行结果如下：



- 进一步的思考与练习：

（1）查询订单信息中订购数量的平均值，使用 AVG()函数，如下：

```
SELECT AVG(数量) AS '平均订购数量'
        FROM OrderList
```

16、统计食品类商品的总库存量.

- 分析：查询结果信息来自 GoodsInfo，查询条件只涉及 GoodsInfo，故为单表查询；

查询为商品类别为商品类的库存总量，在 SELECT 语句中用 SUM()，WHERE 用比较运算符“=”，注意区分 SUM()和 COUNT()区别。

- 设计的 SQL 语句：

```
SELECT SUM(库存量) AS '食品类库存量'
        FROM GoodsInfo
        WHERE  商品类别='食品'
```

- 语句执行结果如下：

17、查询食品类的商品数。

- 分析：查询结果信息来自 GoodsInfo，查询条件只涉及 GoodsInfo，故为单表查询；

查询为商品类别为商品类的库存总量，在 SELECT 语句中用 COUNT()，WHERE 用比较运算符“=”。注意与 16 区分。

- 设计的 SQL 语句：

```
SELECT COUNT(库存量) AS '食品类库存量'
    FROM GoodsInfo
    WHERE  商品类别='食品'
```

- 语句执行结果如下：

18、查询订购了商品的客户总数。

- 分析：查询结果信息来自 OrderList，查询条件只涉及 OrderList，故为单表查询；查询为订购了商品的客户总数，在 SELECT 语句中用 COUNT()和 DISTINCT()进行去重。
- 设计的 SQL 语句：

```
SELECT COUNT(DISTINCT(客户编号)) AS '订购商品客户数'
    FROM OrderList
```

- 语句执行结果如下：

19、统计各地的客户数量。

- 分析：查询结果信息来自 CustomerInfo，查询条件只涉及 CustomerInfo，故为单表查询；查询为各地客户数量，在 SELECT 语句查询所在省市，GROUP BY 所在省市。
- 设计的 SQL 语句：

```
SELECT  所在省市, COUNT(*) AS '人数'
    FROM CustomerInfo
    GROUP BY  所在省市
```

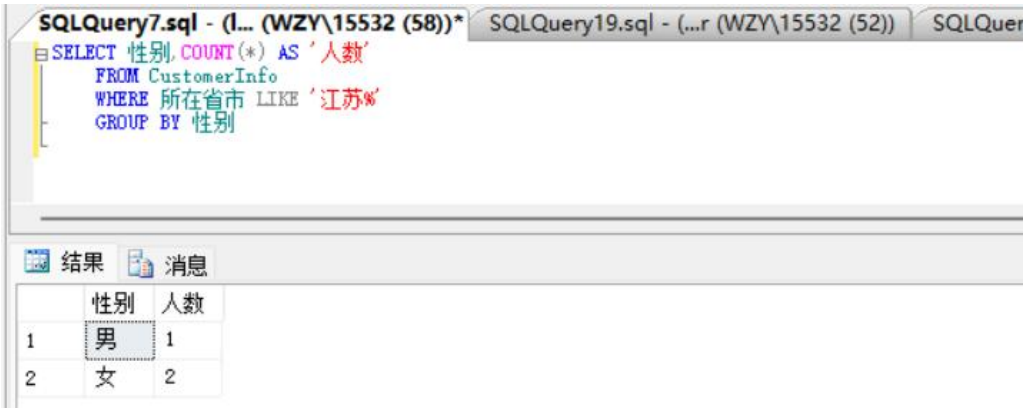
- 语句执行结果如下：

20、按性别统计江苏的客户数量。

- 分析：查询结果信息来自 CustomerInfo，查询条件只涉及 CustomerInfo，故为单表查询；按性别统计江苏的客户数量目标列为性别、人数，在 WHERE 语句中使用模糊查询，再按性别分组。
- 设计的 SQL 语句：

```
SELECT 性别,COUNT(*) AS '人数'
    FROM CustomerInfo
    WHERE 所在省市 LIKE '江苏%'
    GROUP BY 性别
```

- 语句执行结果如下：

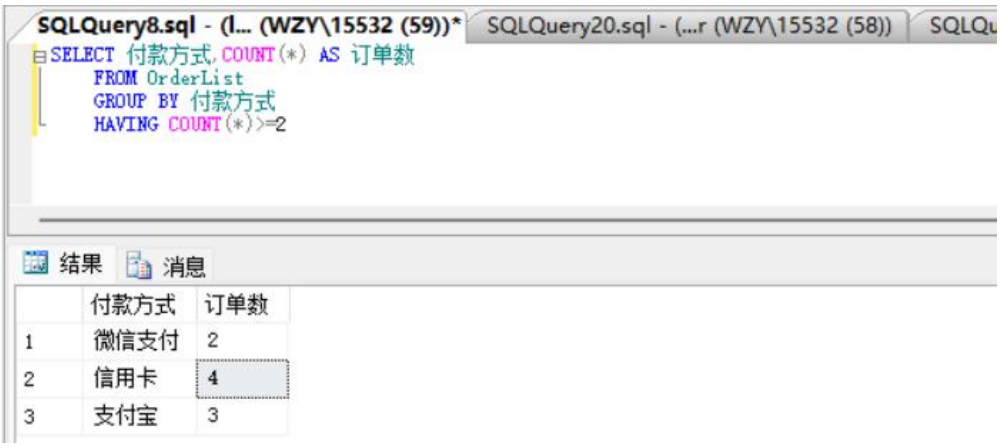


21、按付款方式统计订单数量不少于 2 笔的付款方式及订单数。

- 分析：查询结果信息来自 OrderList，查询条件只涉及 OrderList，故为单表查询；按付款方式分组，再找出其中大于等于 2 的订单数目。
- 设计的 SQL 语句：

```
SELECT 付款方式,COUNT(*) AS 订单数
    FROM OrderList
    GROUP BY 付款方式
    HAVING COUNT(*)>=2
```

- 语句执行结果如下：



22、查询商品类别为“食品”或“服装”，且单价在 20 元（含）以上的商品的编号和名称。

- 分析：查询结果信息来自 GoodsInfo，查询条件只涉及 GoodsInfo，故为单表查询；目标列为商品编号和名称，WHERE 中逻辑判断其他条件。
- 设计的 SQL 语句：

```
SELECT 商品编号,商品名称
    FROM GoodsInfo
    WHERE (商品类别='食品' OR 商品类别='服装') AND (单价 >= 20)
```

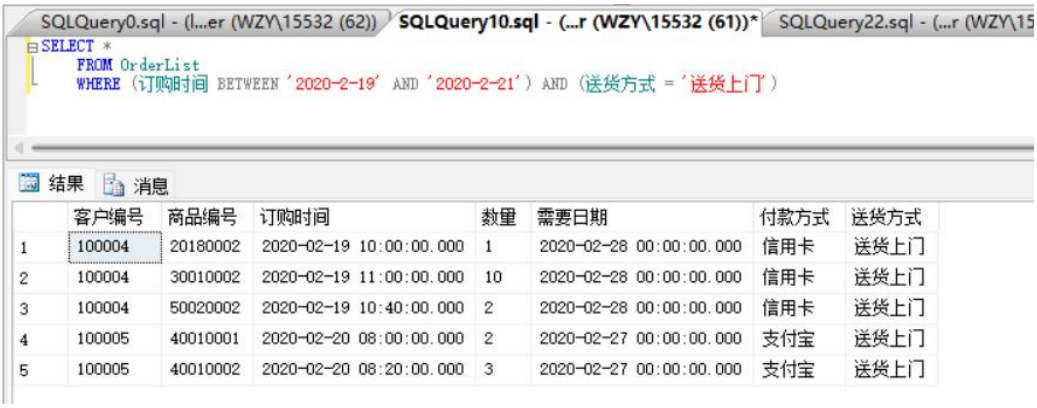
- 语句执行结果如下：

23、查询订购时间在“2020-2-19”和“2020-2-20”之间，且送货方式为“送货上门”的订单信息。

- 分析：查询结果信息来自 OrderList，查询条件只涉及 OrderList，故为单表查询；WHERE 中逻辑判断其他条件，其中若写 2020-2-20，则只会到此天零点，所以应该写 2020-2-21。
- 设计的 SQL 语句：

```
SELECT *
    FROM OrderList
    WHERE (订购时间 BETWEEN '2020-2-19' AND '2020-2-21') AND (送货方式 ='送货上门')
```

- 语句执行结果如下：



	客户编号	商品编号	订购时间	数量	需要日期	付款方式	送货方式
1	100004	20180002	2020-02-19 10:00:00.000	1	2020-02-28 00:00:00.000	信用卡	送货上门
2	100004	30010002	2020-02-19 11:00:00.000	10	2020-02-28 00:00:00.000	信用卡	送货上门
3	100004	50020002	2020-02-19 10:40:00.000	2	2020-02-28 00:00:00.000	信用卡	送货上门
4	100005	40010001	2020-02-20 08:00:00.000	2	2020-02-27 00:00:00.000	支付宝	送货上门
5	100005	40010002	2020-02-20 08:20:00.000	3	2020-02-27 00:00:00.000	支付宝	送货上门

24、查询每个客户订购商品的平均数量。

- 分析：查询结果信息来自 OrderList，查询条件只涉及 OrderList，故为单表查询；目标列为客户编号、平均订购数量，由客户编号分组，在使用 AVG()函数得到平均订购数量。
- 设计的 SQL 语句：

```
SELECT 客户编号 , AVG(数量) AS '平均订购数量'
    FROM OrderList
    GROUP BY 客户编号
```

- 语句执行结果如下：

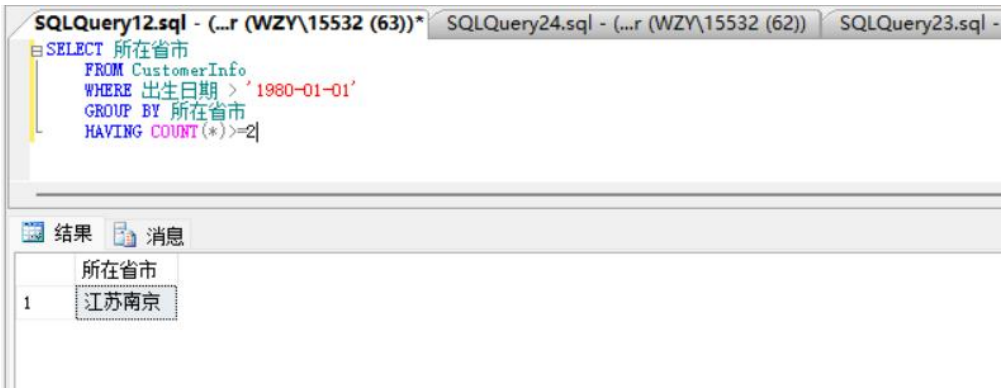
	客户编号	平均订购数量
1	100001	6
2	100002	1
3	100004	4
4	100005	2
5	100006	5

25、查找 1980 年以后出生的客户数不少于 2 个的省市。

- 分析：查询结果信息来自 CustomerInfo，查询条件只涉及 CustomerInfo，故为单表查询；目标列为所在省市，先 WHERE 子句筛选出满足条件的客户，再由所在省市分组，筛选其中满足条件的所在省市。
- 设计的 SQL 语句：

```
SELECT 所在省市
    FROM CustomerInfo
    WHERE 出生日期 >'1980-01-01'
    GROUP BY 所在省市
    HAVING COUNT(*)>=2
```

- 语句执行结果如下：



26、查找总库存量超过 100 件的商品类别及其总库存量。

- 分析：查询结果信息来自 GoodsInfo，查询条件只涉及 GoodsInfo，故为单表查询；目标列为商品类别、总库存量，先由商品类别分组，再用 SUM()函数计算总库存量，并筛选其中满足条件的商品。
- 设计的 SQL 语句：

```
SELECT  商品类别,SUM(库存量) AS '总库存量'
        FROM GoodsInfo
        GROUP BY  商品类别
        HAVING SUM(库存量)>=100
```

- 语句执行结果如下：

四、实验总结

在这次实验中，我通过编写 SQL 语句解决了一系列实际的查询任务。实验内容包括客户信息查询、商品信息筛选以及订单统计等多个方面。通过这些任务，我更加熟练地掌握了 SQL 的基础语法和常用功能，如 WHERE、AND、OR、BETWEEN、IN、LIKE 等关键字，并学会了如何灵活组合它们来实现各种复杂的查询需求。这次实践使我更加清晰地理解了如何在数据库中处理大量数据并高效地提取所需的信息。

在实验中，我也遇到了一些问题。例如，处理日期查询时，我发现 BETWEEN...AND... 中后面的日期比如 2020-2-19 只到当天 0 点，导致当天数据遗漏，所以需要改写成 2020-2-20 以查找所有满足条件数据。此外，在进行模糊匹配时，我初次没有正确使用 LIKE 操作符的通配符，导致查询没有返回正确的结果。经过学习后，我成功掌握各种模糊查询以及转义字符使用方法。