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 SQL 视图 实验（实习）日期 2024-4-27 得分 指导教师
系 计算机学院 专业 计科

一、实验目的

1. 理解 SQL 视图的基本概念
2. 掌握 SQL 视图的创建方法
3. 掌握 SQL 视图查询的设计方法
4. 理解 SQL 视图更新的要求
5. 掌握使用 INSERT、UPDATE、DELETE 通过视图更新基本表数据的方法

二、实验内容

在 GoodsOrder 数据库中，用 SQL 语句实现如下查询：

79、创建视图 GInfo_food：食品类商品信息。

80、创建视图 Order_food：客户订购食品类商品的订单信息，包括客户编号、客户姓名、商品编号、商品名称、数量、订购时间、需要日期、付款方式。

81、创建视图 Goods_Expired：过期食品信息，即保质期在查询当日之前的食品类商品。

82、创建视图 Goods_OrderNum：统计各个编号商品订购的总数量 ，若商品未被订购则统计为 0。

83、创建视图 Goods_Sales：统计各个编号商品的销售金额，包括商品编号、商品名称、销售金额；若商品未被订购则统计为 0。

84、创建视图 Cust_TotalCost：统计每个客户订购总金额，包括客户编号、客户姓名、订购总金额；若客户未订购任何商品，则订购总金额为 0。

85、查询食品类商品信息。

86、查询商品“咖啡”的信息。

87、查询单价小于 20 的食品类商品信息。

88、查询 “食品”类商品的总价值。

89、查询客户订购“食品”类商品的订单信息。

90、查询客户订购“食品”类商品、单个订单中“数量”>=2 的订单信息。

91、查询“张小林”订购过的“食品”类订单信息。

92、查询过期食品信息。

93、查询过期时间最长的食品信息。

94、查询各个编号商品被订购的总数量（若商品未被订购则统计为 0） 。

95、查询订购总数量在 2 个及以上的商品编号及其订购总数量。

96、统计全部商品的订购总数量。

97、查询订购总数量最多的商品编号及其订购数。

98、查询各商品的商品编号、商品名称、销售金额（若商品未被订购则统计为 0）。

100、查询销售金额最多的商品编号、商品名称和销售额。

101、查询所有商品的销售总额。

102、【视图连接】查询每个商品的商品编号、商品名称、销售额、订购数量。

103、查询每个客户客户编号、客户姓名、订购总金额；若客户未订购任何商品，则订购总金额为 0。

104、查询订购总金额最多的客户编号、客户姓名、订购总金额。

105、通过 GInfo_food 视图，向商品表中插入一条商品数据： ('10030001' , '食品' , '白糖' , '蜜之源' ,6, '康源糖业有限公司' , '2021-09- 20' ,200,NULL)

106、通过 GInfo_food 视图，将商品表中“宇一饮料公司”生产的食品类商品的备注信息均改为“宇一生产” 。

107、通过 GInfo_food 视图，删除商品表中编号为“10030001”的商品数据。

108、通过 Goods_Expired 视图将商品表中过期食品的库存量和单价均置为 0。

三、实验过程与结果

79、创建视图 GInfo_food：食品类商品信息。

- 分析：选择 GoodsInfo 的所有列，WHERE 选择商品类别为“食品”的创建视图 GInfo_food。
- 设计的 SQL 语句如下：

```
CREATE VIEW GInfo_fodd
    AS
    SELECT *
        FROM GoodsInfo
        WHERE  商品类别 ＝'食品'
```

- 语句执行结果如下：

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

80、创建视图 Order_food：客户订购食品类商品的订单信息，包括客户编号、客户姓名、商品编号、商品名称、数量、订购时间、需要日期 、付款方式。

- 分析：先将 OrderList 表和 CustomerInfo 表、GoodsInfo 表相连接，再选择自己需要的列创建视图。
- 设计的 SQL 语句如下：

```
CREATE VIEW Order_food
    AS
    SELECT b.客户编号,b.客户姓名,c.商品编号,c.商品名称,a.数量,a.订购时间,a.需要日期,a.付款方式
        FROM OrderList a
        JOIN CustomerInfo b ON a.客户编号 ＝b.客户编号
        JOIN GoodsInfo c ON a.商品编号 ＝c.商品编号
        WHERE c.商品类别 ＝'食品'
```

- 语句执行结果如下：

结果 消息

	客户编号	客户姓名	商品编号	商品名称	数量	订购时间	需要日期	付款方式
1	100001	张小林	10010001	咖啡	2	2020-02-18 12:20:00.000	2020-02-20 00:00:00.000	支付宝
2	100002	李红红	10010001	咖啡	1	2020-02-18 13:00:00.000	2020-02-21 00:00:00.000	微信支付
3	100006	王芳芳	10020001	大米	5	2020-02-23 09:00:00.000	2020-02-26 00:00:00.000	信用卡

81、创建视图 Goods_Expired：过期食品信息，即保质期在查询当日之前的食品类商品。

- 分析：选择 GoodsInfo 的所有列，WHERE 选择商品类别为“食品”且保质期在当日之前的商品的创建视图 Goods_Expired。
- 设计的 SQL 语句如下：

```
CREATE VIEW Goods_Expired
    AS
    SELECT *
        FROM GoodsInfo a
        WHERE a.商品类别 ='食品' AND a.保质期 <GETDATE()
```

- 语句执行结果如下：

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

82、创建视图 Goods_OrderNum：统计各个编号商品订购的总数量 ，若商品未被订购则统计为 0。

分析：GoodsInfo 左连接 OrderList，使用 ISNULL()函数判断是否为空值。

- 设计的 SQL 语句如下：

```
CREATE VIEW Goods_OrderNum
    AS
    SELECT a.商品编号,SUM(ISNULL(b.数量,0)) AS '订购总数量'
        FROM GoodsInfo a
        LEFT JOIN OrderList b ON a.商品编号 =b.商品编号
        GROUP BY a.商品编号
```

- 语句执行结果如下：

结果 消息

	商品编号	订购总数量
1	10010001	3
2	10010002	0
3	10020001	5
4	10020002	0
5	20180001	0
6	20180002	1
7	30010001	10
8	30010002	10
9	40010001	2
10	40010002	3
11	50020001	1
12	50020002	2
13	50020003	0

83、创建视图 Goods_Sales：统计各个编号商品的销售金额，包括商品编号、商品名称、销售金额；若商品未被订购则统计为 0。

- 分析：GoodsInfo 左连接 OrderList，使用 ISNULL()函数判断订购数量是否为空值，在上一个视图的基础上乘以单价即得到销售金额，再额外选择其他的列。
- 设计的 SQL 语句如下：

```
CREATE VIEW Goods_Sales
    AS
    SELECT a.商品编号,a.商品名称,SUM(ISNULL(b.数量,0)) * a.单价 AS 销售金额
        FROM GoodsInfo a
        LEFT JOIN OrderList b ON a.商品编号 = b.商品编号
        GROUP BY a.商品编号,a.商品名称,a.单价
```

- 语句执行结果如下：

结果 消息

	商品编号	商品名称	销售金额
1	10010001	咖啡	150
2	10010002	苹果汁	0
3	10020001	大米	175
4	10020002	面粉	0
5	20180001	运动服	0
6	20180002	T恤	120
7	30010001	签字笔	35
8	30010002	文件夹	56
9	40010001	营养菜谱	76
10	40010002	豆浆的做法	60
11	50020001	羽毛球拍	30
12	50020002	篮球	160
13	50020003	足球	0

84、创建视图 Cust_TotalCost：统计每个客户订购总金额，包括客户编号、客户姓名、订购总金额；若客户未订购任何商品，则订购总金额为 0。

- 分析：CustomerInfo 左连接 OrderList 和 GoodsInfo，用 ISNULL()函数处理空值。
- 设计的 SQL 语句如下：

```
CREATE VIEW Cust_TotalCost
    AS
    SELECT a.客户编号,a.客户姓名,ISNULL(SUM(ISNULL(b.数量,0) * c.单价),0) AS 订购总金额
        FROM CustomerInfo a
```

LEFT JOIN OrderList b ON a.客户编号 ＝b.客户编号
LEFT JOIN GoodsInfo c ON b.商品编号 ＝c.商品编号
GROUP BY a.客户编号,a.客户姓名

- 语句执行结果如下：

	客户编号	客户姓名	订购总金额
1	100001	张小林	135
2	100002	李红红	80
3	100003	王晓美	0
4	100004	赵明	336
5	100005	张帆一	136
6	100006	王芳芳	175

85、查询食品类商品信息。

- 分析：选择 GInfo_food 所有列。
- 设计的 SQL 语句如下：

```
SELECT *
    FROM GInfo_food
```

- 语句执行结果如下：

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	库存里	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

86、查询商品“咖啡”的信息。

- 分析：由于咖啡属于商品类，在 GInfo_food 视图里面查找。
- 设计的 SQL 语句如下：

```
SELECT *
    FROM GInfo_food
    WHERE  商品名称 ＝'咖啡'
```

- 语句执行结果如下：

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	库存里	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

87、查询单价小于 20 的食品类商品信息。

- 分析：在 GInfo_food 视图里面查找单价小于 20 的商品信息。
- 设计的 SQL 语句如下：

```
SELECT *
```

```
FROM GInfo_food
WHERE 单价 ＜20
```

- 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	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

- 进一步的思考与练习：

（1）查找单价在 20 和 50 之间（包含 20 和 50）的食品类商品信息，如下：

```
SELECT *
    FROM GInfo_food
    WHERE 单价 BETWEEN 20 AND 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	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

88、查询 “食品”类商品的总价值。

- 分析：在 GInfo_food 表中查找，SUM()函数算出总价值。
- 设计的 SQL 语句如下：

```
SELECT SUM(单价 * 库存量) AS 总价值
    FROM GInfo_food
```

- 语句执行结果如下：

	总价值
1	11460

89、查询客户订购“食品”类商品的订单信息。

- 分析：选择 Order_food 的所有列。
- 设计的 SQL 语句如下：

```
SELECT *
    FROM Order_food
```

- 语句执行结果如下：

	客户编号	客户姓名	商品编号	商品名称	数量	订购时间	需要日期	付款方式
1	100001	张小林	10010001	咖啡	2	2020-02-18 12:20:00.000	2020-02-20 00:00:00.000	支付宝
2	100002	李红红	10010001	咖啡	1	2020-02-18 13:00:00.000	2020-02-21 00:00:00.000	微信支付
3	100006	王芳芳	10020001	大米	5	2020-02-23 09:00:00.000	2020-02-26 00:00:00.000	信用卡

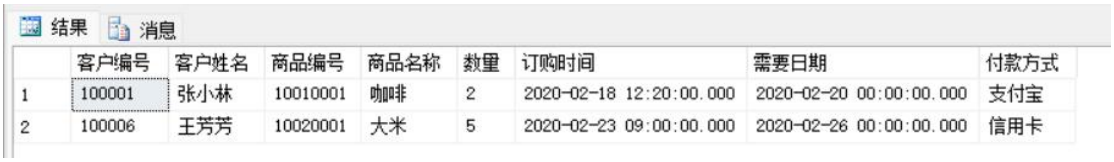
90、查询客户订购“食品”类商品、单个订单中“数量”>=2 的订单信息。。

分析：选择 GoodsInfo 的所有列，WHERE 选择商品订购数量大于等于 2 的。

- 设计的 SQL 语句如下：

```
SELECT *
    FROM Order_food
    WHERE  数量  >= 2
```

- 语句执行结果如下：



	客户编号	客户姓名	商品编号	商品名称	数量	订购时间	需要日期	付款方式
1	100001	张小林	10010001	咖啡	2	2020-02-18 12:20:00.000	2020-02-20 00:00:00.000	支付宝
2	100006	王芳芳	10020001	大米	5	2020-02-23 09:00:00.000	2020-02-26 00:00:00.000	信用卡

91、查询“张小林”订购过的“食品”类订单信息。。

- 分析：选择 GoodsInfo 的所有列，WHERE 选择客户姓名为“张小林”的客户。
- 设计的 SQL 语句如下：

```
SELECT *
    FROM Order_food
    WHERE  客户姓名  = '张小林'
```

- 语句执行结果如下：

	客户编号	客户姓名	商品编号	商品名称	数量	订购时间	需要日期	付款方式
1	100001	张小林	10010001	咖啡	2	2020-02-18 12:20:00.000	2020-02-20 00:00:00.000	支付宝

92、查询过期食品信息。

- 分析：选择 Goods_Expired 的所有列。
- 设计的 SQL 语句如下：

```
SELECT *
    FROM Goods_Expired
```

- 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

93、查询过期时间最长的食品信息。

- 分析：选择 GoodsInfo 的所有列，WHERE 选择保质期最小的商品，嵌套查询实现。
- 设计的 SQL 语句如下：

```
SELECT *
    FROM Goods_Expired
    WHERE  保质期  =
    (
        SELECT MIN(保质期)
            FROM Goods_Expired
    )
```

- 语句执行结果如下：

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	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

94、查询各个编号商品被订购的总数量（若商品未被订购则统计为 0）。

- 分析：选择 Goods_OrderNum 所有列。
- 设计的 SQL 语句如下：

```
SELECT *
    FROM Goods_OrderNum
```

- 语句执行结果如下：

结果 消息

	商品编号	订购总数量
1	10010001	3
2	10010002	0
3	10020001	5
4	10020002	0
5	20180001	0
6	20180002	1
7	30010001	10
8	30010002	10
9	40010001	2
10	40010002	3
11	50020001	1
12	50020002	2
13	50020003	0

95、查询订购总数量在 2 个及以上的商品编号及其订购总数量。

- 分析：选择 GoodsInfo 的所有列，WHERE 选择订购总数量大于等于 2 的商品。
- 设计的 SQL 语句如下：

```
SELECT *
    FROM Goods_OrderNum
    WHERE  订购总数量  >= 2
```

- 语句执行结果如下：

结果 消息

	商品编号	订购总数量
1	10010001	3
2	10020001	5
3	30010001	10
4	30010002	10
5	40010001	2
6	40010002	3
7	50020002	2

96、统计全部商品的订购总数量。

- 分析：使用 SUM()函数。
- 设计的 SQL 语句如下：

```
SELECT SUM(订购总数量) AS  订购全部数量
    FROM Goods_OrderNum
```

- 语句执行结果如下：

	订购全部数量
1	37

97、查询订购总数量最多的商品编号及其订购数。

- 分析：选择 GoodsInfo 的所有列，WHERE 选择商品订购总数量等于最多数量的商品，嵌套查询实现。
- 设计的 SQL 语句如下：

```
SELECT *
    FROM Goods_OrderNum
    WHERE  订购总数量 =
    (
        SELECT MAX(订购总数量)
            FROM Goods_OrderNum
    )
```

- 语句执行结果如下：

结果 消息

	商品编号	订购总数量
1	30010001	10
2	30010002	10

98、查询各商品的商品编号、商品名称、销售金额（若商品未被订购则统计为 0）。

- 分析：选择 Goods_Sales 所有列。
- 设计的 SQL 语句如下：

```
SELECT *
    FROM Goods_Sales
```

- 语句执行结果如下：

	商品编号	商品名称	销售金额
1	10010001	咖啡	150
2	10010002	苹果汁	0
3	10020001	大米	175
4	10020002	面粉	0
5	20180001	运动服	0
6	20180002	T恤	120
7	30010001	签字笔	35
8	30010002	文件夹	56
9	40010001	营养菜谱	76
10	40010002	豆浆的做法	60
11	50020001	羽毛球拍	30
12	50020002	篮球	160
13	50020003	足球	0

100、查询销售金额最多的商品编号、商品名称和销售额。

- 分析：选择 Goods_Sales 的所有列，WHERE 选择销售金额等于最多销售金额的，嵌套查询。
- 设计的 SQL 语句如下：

```
SELECT *
    FROM Goods_Sales
    WHERE  销售金额 =
    (
        SELECT MAX(销售金额)
            FROM Goods_Sales
    )
```

- 语句执行结果如下：

结果 消息

	商品编号	商品名称	销售金额
1	10020001	大米	175

101、查询所有商品的销售总额。

- 分析：使用 SUM()函数。
- 设计的 SQL 语句如下：

```
SELECT SUM(销售金额) AS  销售总额
    FROM Goods_Sales
```

- 语句执行结果如下：

	销售总额
1	862

102、【视图连接】查询每个商品的商品编号、商品名称、销售额、订购数量。

分析：连接 Goods_OrderNum 和 Goods_Sales，选择目标列。

- 设计的 SQL 语句如下：

```
SELECT a.商品编号,a.商品名称,a.销售金额,b.订购总数量
    FROM Goods_Sales a JOIN Goods_OrderNum b ON a.商品编号 = b.商品编号
```

- 语句执行结果如下：

	商品编号	商品名称	销售金额	订购总数量
1	10010001	咖啡	150	3
2	10010002	苹果汁	0	0
3	10020001	大米	175	5
4	10020002	面粉	0	0
5	20180001	运动服	0	0
6	20180002	T恤	120	1
7	30010001	签字笔	35	10
8	30010002	文件夹	56	10
9	40010001	营养菜谱	76	2
10	40010002	豆浆的做法	60	3
11	50020001	羽毛球拍	30	1
12	50020002	篮球	160	2
13	50020003	足球	0	0

103、查询每个客户客户编号、客户姓名、订购总金额；若客户未订购任何商品，则订购总金额为 0。

- 分析：选择 Cust_TotalCost 的所有列。
- 设计的 SQL 语句如下：

```
SELECT *
    FROM Cust_TotalCost
```

- 语句执行结果如下：

	客户编号	客户姓名	订购总金额
1	100001	张小林	135
2	100002	李红红	80
3	100003	王晓美	0
4	100004	赵明	336
5	100005	张帆一	136
6	100006	王芳芳	175

104、查询订购总金额最多的客户编号、客户姓名、订购总金额。

- 分析：选择 Cust_TotalCost 的所有列，WHERE 选择订购总金额最大的行，嵌套查询实现。
- 设计的 SQL 语句如下：

```
SELECT *
    FROM Cust_TotalCost
    WHERE  订购总金额 =
    (
        SELECT MAX(订购总金额)
            FROM Cust_TotalCost
    )
```

- 语句执行结果如下：

结果 消息

	客户编号	客户姓名	订购总金额
1	100004	赵明	336

105、通过 GInfo_food 视图，向商品表中插入一条商品数据： ('10030001' , '食品' , '白糖' , '蜜之源' ,6, '康源糖业有限公司' , '2021-09- 20' ,200,NULL)。

- 分析：使用 INSERT INTO 语句。
- 设计的 SQL 语句如下：

```
INSERT INTO GInfo_food
    (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)
VALUES
    ('10030001','食品','白糖','蜜之源',6,'康源糖业有限公司','2021-09- 20',200,NULL)
```

- 语句执行结果如下：

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	库存里	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
5	10030001	食品	白糖	蜜之源	6	康源糖业有限公司	2021-09-20 00:00:00.000	200	NULL

106、通过 GInfo_food 视图，将商品表中“宇一饮料公司”生产的食品类商品的备注信息均改为“宇一生产”。

- 分析：使用 UPDATE，选择生产商为“宇一饮料公司”，将其备注改成宇一生产。
- 设计的 SQL 语句如下：

```
UPDATE GInfo_food
    SET 备注 ='宇一生产'
    WHERE 生产商 ='宇一饮料公司'
```

- 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	库存里	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	宇一生产
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	宇一生产
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
5	10030001	食品	白糖	蜜之源	6	康源糖业有限公司	2021-09-20 00:00:00.000	200	NULL

107、通过 GInfo_food 视图，删除商品表中编号为“10030001”的商品数据。

- 分析：使用 DELETE 语句。
- 设计的 SQL 语句如下：

```
DELETE FROM GInfo_food
    WHERE 商品编号 ='10030001'
```

- 语句执行结果如下：

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	库存里	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	宇一生产
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	宇一生产
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

108、通过 Goods_Expired 视图将商品表中过期食品的库存量和单价均置为 0。

- 分析：使用 UPDATE 语句。
- 设计的 SQL 语句如下：

```
UPDATE Goods_Expired
    SET 库存量 =0,单价 =0
```

- 语句执行结果如下：

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	0	宇一饮料公司	2021-08-31 00:00:00.000	0	宇一生产
2	10010002	食品	苹果汁	宇一	0	宇一饮料公司	2020-12-31 00:00:00.000	0	宇一生产
3	10020001	食品	大米	健康	0	健康粮食生产基地	2020-12-20 00:00:00.000	0	NULL
4	10020002	食品	面粉	健康	0	健康粮食生产基地	2021-01-20 00:00:00.000	0	NULL

四、实验总结

在这次实验中，我了解到视图创建的基本方法，也了解了如何在创建的视图上进行查询，以及视图的添加、修改和删除等操作。通过此次实践，我对视图的使用与管理有了更加深入的理解。