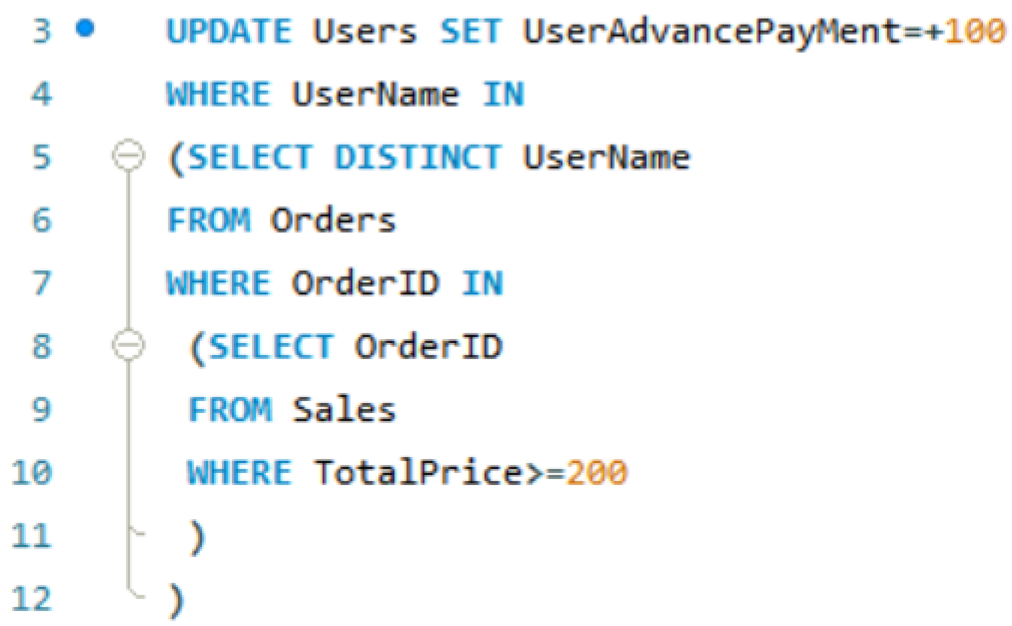
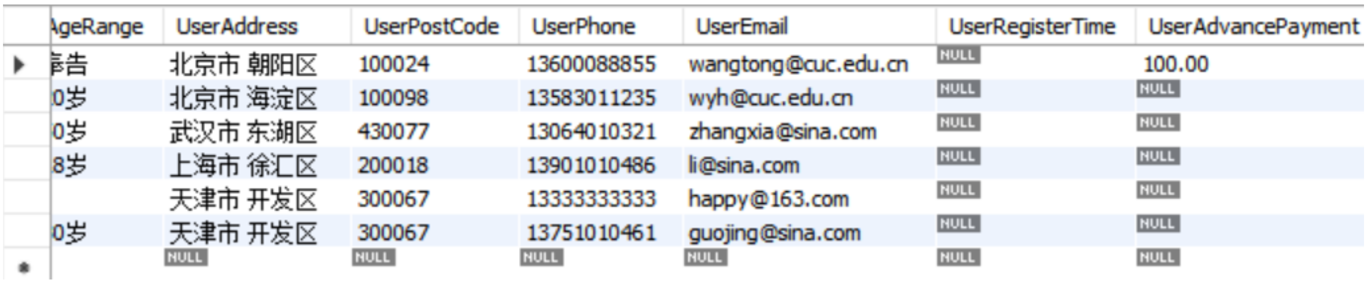
# 批量操作及视图练习

实验内容

（1）商家开展“买200赠100”的促销活动，为一次购买某专辑总价格在200元（含）以上的用户的会员卡（字段：User表的UserAdvancePayment）充值100元。（只充值一次）

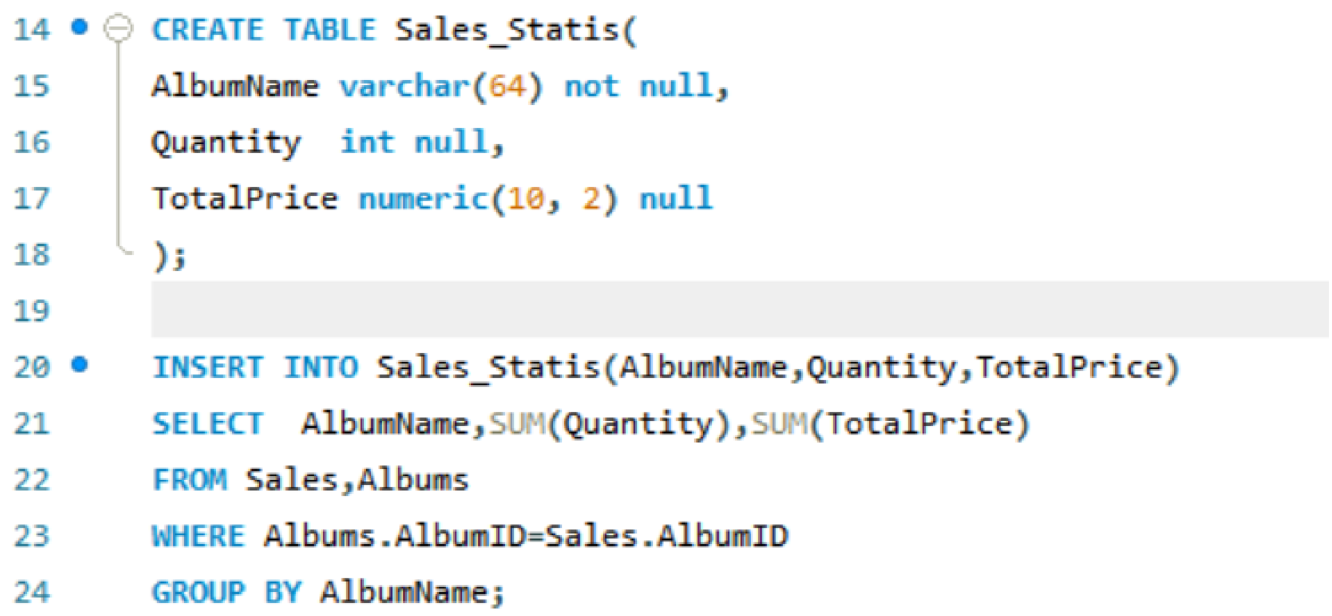


```sql
UPDATE Users SET UserAdvancePayMent=+100
WHERE UserName IN
(SELECT DISTINCT UserName
FROM Orders
WHERE OrderID IN
 (SELECT OrderID
 FROM Sales
 WHERE TotalPrice>=200
 )
)
```



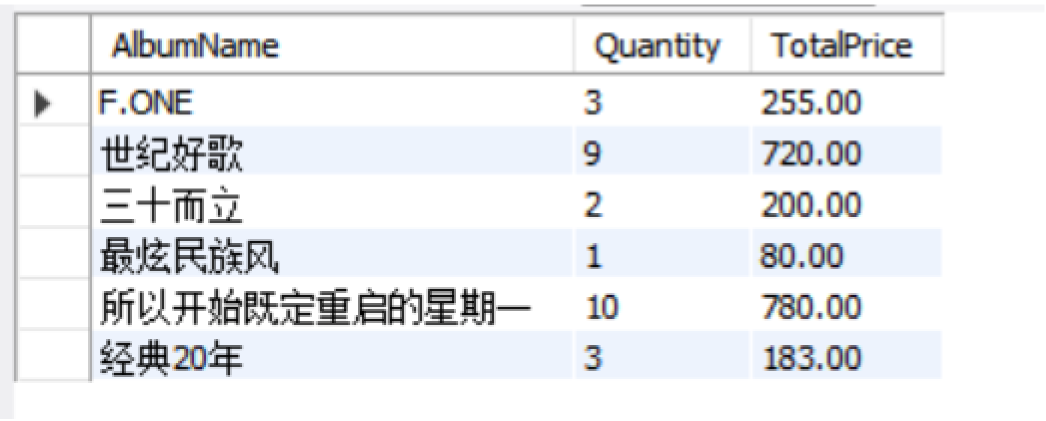
| AgeRange | UserAddress | UserPostCode | UserPhone | UserEmail | UserRegisterTime | UserAdvancePayment |
|---|---|---|---|---|---|---|
| 告 | 北京市 朝阳区 | 100024 | 13600088855 | wangtong@cuc.edu.cn | NULL | 100.00 |
| 0岁 | 北京市 海淀区 | 100098 | 13583011235 | wyh@cuc.edu.cn | NULL | NULL |
| 0岁 | 武汉市 东湖区 | 430077 | 13064010321 | zhangxia@sina.com | NULL | NULL |
| 8岁 | 上海市 徐汇区 | 200018 | 13901010486 | li@sina.com | NULL | NULL |
|  | 天津市 开发区 | 300067 | 13333333333 | happy@163.com | NULL | NULL |
| 0岁 | 天津市 开发区 | 300067 | 13751010461 | guojing@sina.com | NULL | NULL |
|  | NULL | NULL | NULL | NULL | NULL | NULL |

（2）新建“专辑销售统计表”Sales_Statis，其结构能存放专辑名、销售量和销售额。将各个专辑销售量和销售额的统计结果存入Sales_Statis表中



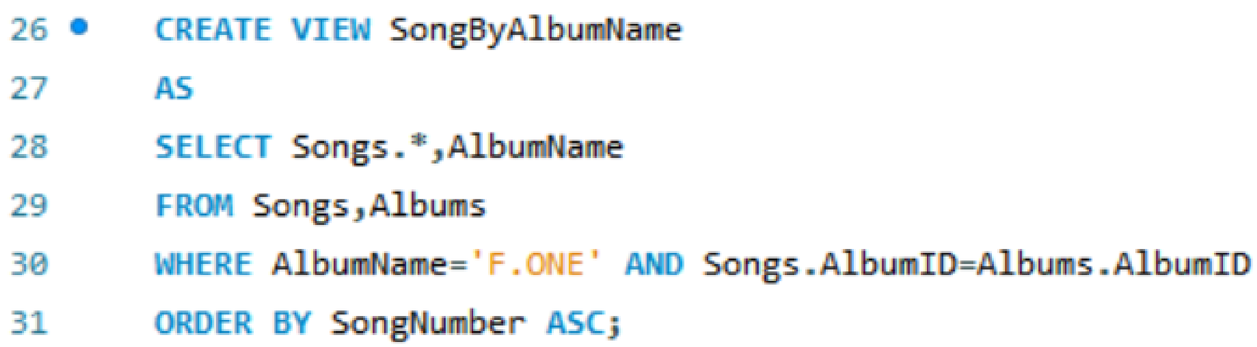
```
CREATE TABLE Sales_Statis(
AlbumName varchar(64) not null,
Quantity  int null,
TotalPrice numeric(10, 2) null
);

INSERT INTO Sales_Statis(AlbumName,Quantity,TotalPrice)
SELECT  AlbumName,SUM(Quantity),SUM(TotalPrice)
FROM Sales,Albums
WHERE Albums.AlbumID=Sales.AlbumID
GROUP BY AlbumName;
```



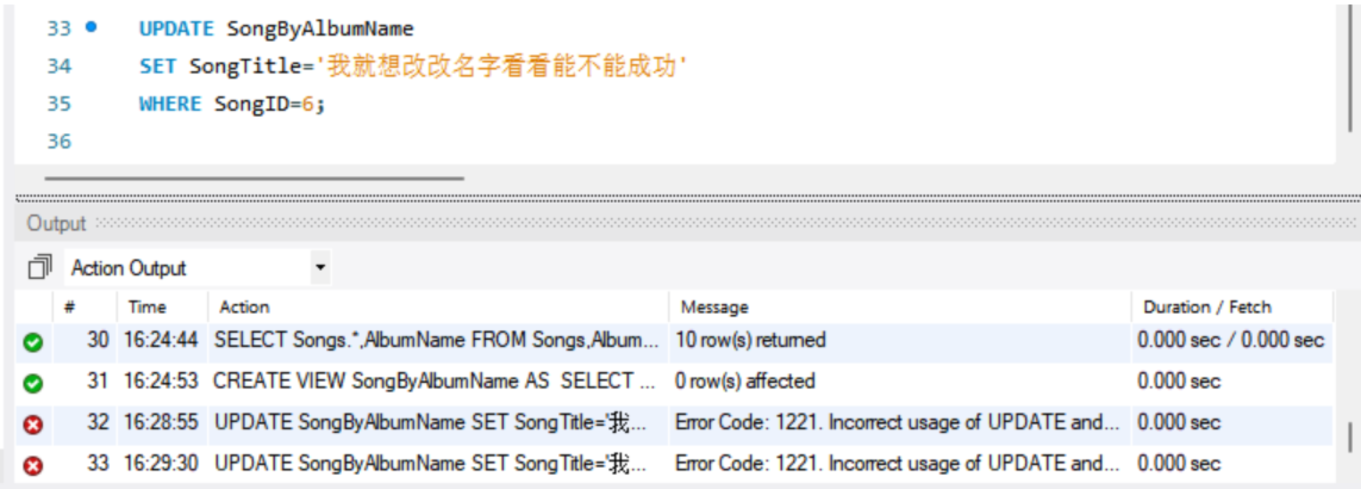
| | AlbumName | Quantity | TotalPrice |
|---|---|---|---|
| ▶ | F.ONE | 3 | 255.00 |
| | 世纪好歌 | 9 | 720.00 |
| | 三十而立 | 2 | 200.00 |
| | 最炫民族风 | 1 | 80.00 |
| | 所以开始既定重启的星期一 | 10 | 780.00 |
| | 经典20年 | 3 | 183.00 |

（3）创建专辑名为“F.ONE”的歌曲视图SongsByAlbumName，该视图包括“歌曲表”中的所有列以及专辑名称，并按歌曲顺序号由小到大排列



```
CREATE VIEW SongByAlbumName
AS
SELECT Songs.*,AlbumName
FROM Songs,Albums
WHERE AlbumName='F.ONE' AND Songs.AlbumID=Albums.AlbumID
ORDER BY SongNumber ASC;
```

（4）将此视图中选择任意SongId，修改歌曲名称为“我就想改改名字看看能不能成功”，看看是否成功



```sql
UPDATE SongByAlbumName
SET SongTitle='我就想改改名字看看能不能成功'
WHERE SongID=6;

```

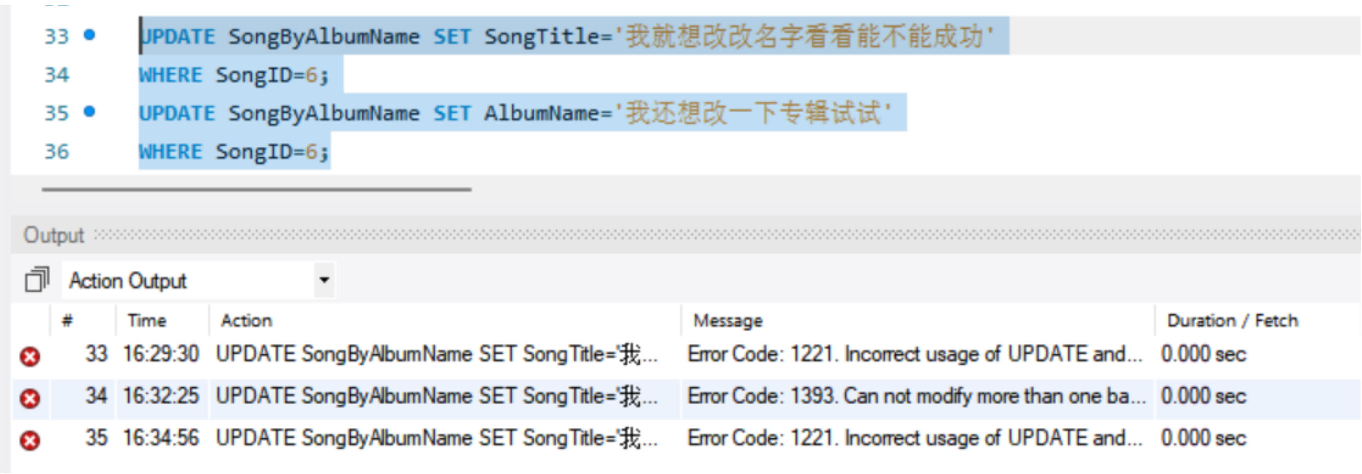
Output

Action Output

|  | # | Time | Action | Message | Duration / Fetch |
|---|---|---|---|---|---|
| ✓ | 30 | 16:24:44 | SELECT Songs.*,AlbumName FROM Songs,Album... | 10 row(s) returned | 0.000 sec / 0.000 sec |
| ✓ | 31 | 16:24:53 | CREATE VIEW SongByAlbumName AS  SELECT ... | 0 row(s) affected | 0.000 sec |
| ✗ | 32 | 16:28:55 | UPDATE SongByAlbumName SET SongTitle='我... | Error Code: 1221. Incorrect usage of UPDATE and... | 0.000 sec |
| ✗ | 33 | 16:29:30 | UPDATE SongByAlbumName SET SongTitle='我... | Error Code: 1221. Incorrect usage of UPDATE and... | 0.000 sec |

无法修改成功

（5）将此视图中选择任意SongId，修改歌曲名称为“我就想改改名字看看能不能成功”，专辑名称修改为“我还想改一下专辑试试”，看看是否成功？



```sql
UPDATE SongByAlbumName SET SongTitle='我就想改改名字看看能不能成功'
WHERE SongID=6;
UPDATE SongByAlbumName SET AlbumName='我还想改一下专辑试试'
WHERE SongID=6;
```

Output

Action Output

|  | # | Time | Action | Message | Duration / Fetch |
|---|---|---|---|---|---|
| ✗ | 33 | 16:29:30 | UPDATE SongByAlbumName SET SongTitle='我... | Error Code: 1221. Incorrect usage of UPDATE and... | 0.000 sec |
| ✗ | 34 | 16:32:25 | UPDATE SongByAlbumName SET SongTitle='我... | Error Code: 1393. Can not modify more than one ba... | 0.000 sec |
| ✗ | 35 | 16:34:56 | UPDATE SongByAlbumName SET SongTitle='我... | Error Code: 1221. Incorrect usage of UPDATE and... | 0.000 sec |

无法修改成功

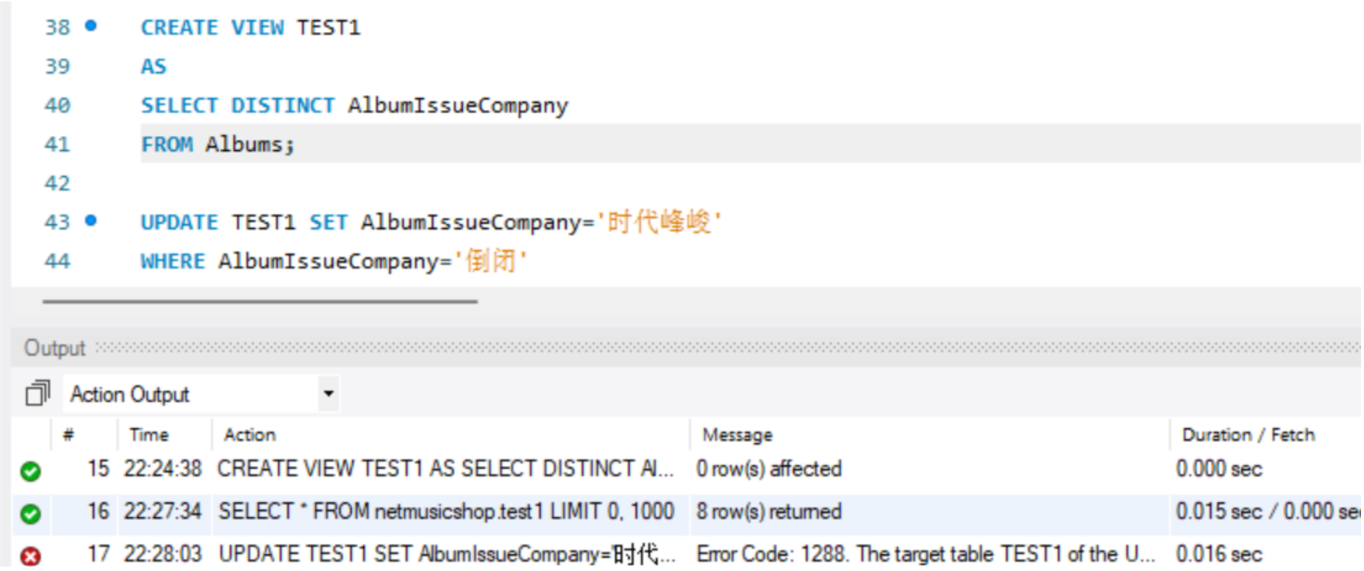
Error Code: 1221. Incorrect usage of UPDATE and ORDER BY

错误原因：因为UPDATE语句需要按照表的逻辑顺序来执行更新操作，而不是按照查询结果的顺序。

解决办法：可以使用子查询和UPDATE语句的多表语法来实现按照特定顺序更新表中的数据。

（6）对于视图是否能修改成功，相关的因素有哪些？请一一列举并举例说明，附上测试结果

1.若视图中含有DISTINCT则无法更新

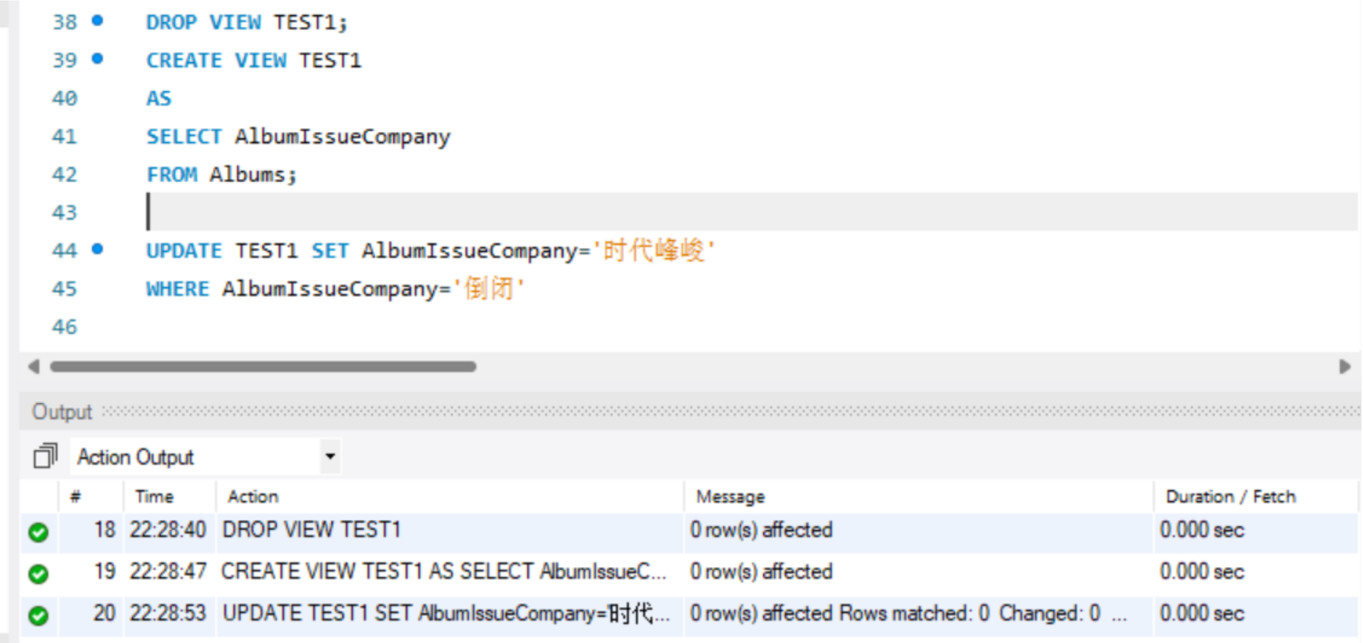


```sql
CREATE VIEW TEST1
AS
SELECT DISTINCT AlbumIssueCompany
FROM Albums;

UPDATE TEST1 SET AlbumIssueCompany='时代峰峻'
WHERE AlbumIssueCompany='倒闭'
```

| | # | Time | Action | Message | Duration / Fetch |
|---|---|---|---|---|---|
| ✓ | 15 | 22:24:38 | CREATE VIEW TEST1 AS SELECT DISTINCT Al... | 0 row(s) affected | 0.000 sec |
| ✓ | 16 | 22:27:34 | SELECT * FROM netmusicshop.test1 LIMIT 0, 1000 | 8 row(s) returned | 0.015 sec / 0.000 se |
| ✗ | 17 | 22:28:03 | UPDATE TEST1 SET AlbumIssueCompany=时代... | Error Code: 1288. The target table TEST1 of the U... | 0.016 sec |

去掉DISTINCT则修改成功

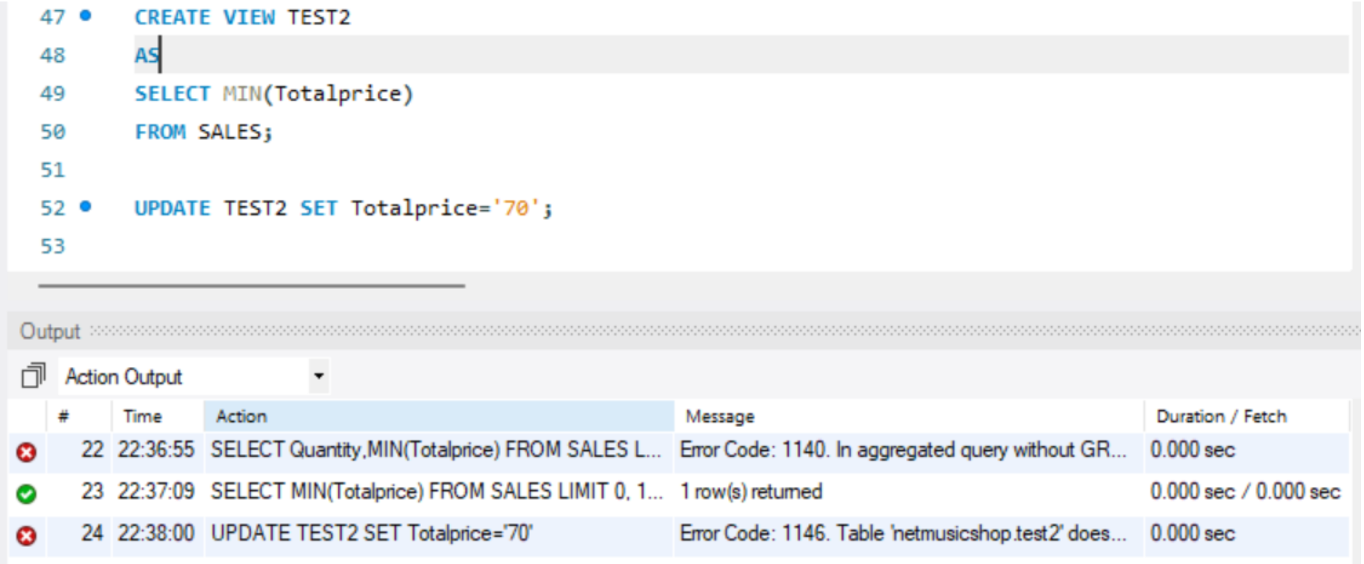


```sql
DROP VIEW TEST1;
CREATE VIEW TEST1
AS
SELECT AlbumIssueCompany
FROM Albums;

UPDATE TEST1 SET AlbumIssueCompany='时代峰峻'
WHERE AlbumIssueCompany='倒闭'
```

| | # | Time | Action | Message | Duration / Fetch |
|---|---|---|---|---|---|
| ✓ | 18 | 22:28:40 | DROP VIEW TEST1 | 0 row(s) affected | 0.000 sec |
| ✓ | 19 | 22:28:47 | CREATE VIEW TEST1 AS SELECT AlbumIssueC... | 0 row(s) affected | 0.000 sec |
| ✓ | 20 | 22:28:53 | UPDATE TEST1 SET AlbumIssueCompany=时代... | 0 row(s) affected Rows matched: 0 Changed: 0 ... | 0.000 sec |

2.若视图中含有聚集函数（SUM/MIN/MAX...）则无法更新

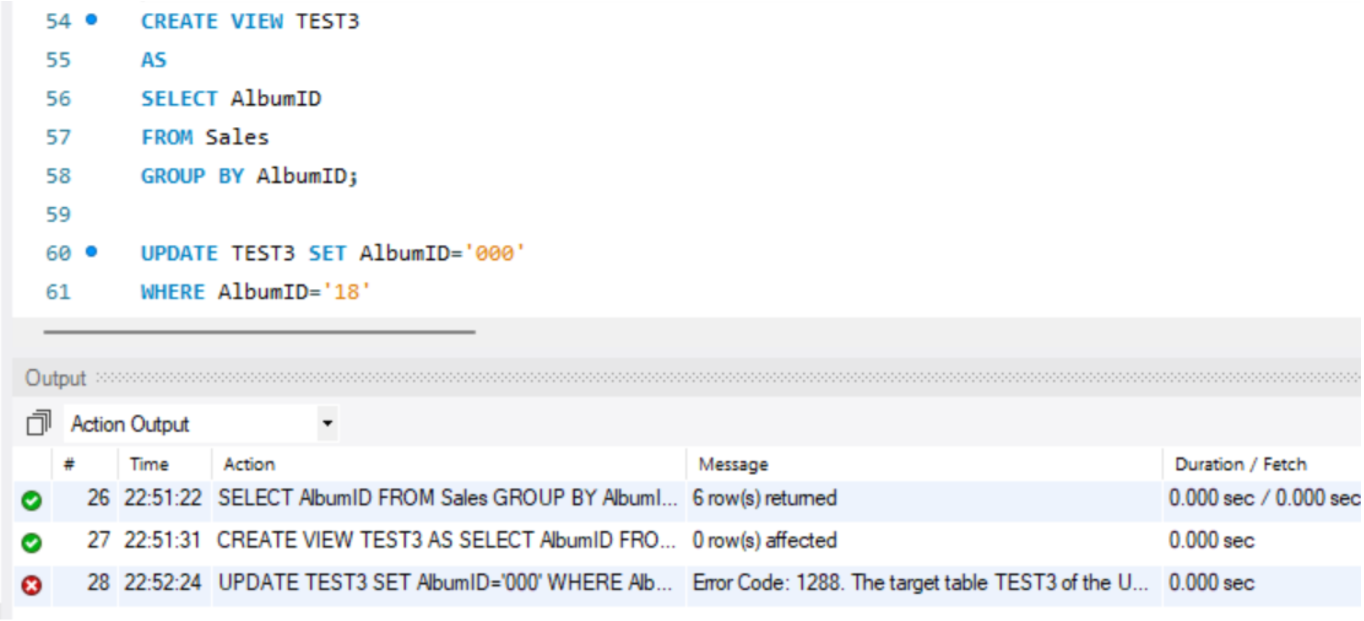


```sql
CREATE VIEW TEST2
AS
SELECT MIN(Totalprice)
FROM SALES;

UPDATE TEST2 SET Totalprice='70';
```

| | # | Time | Action | Message | Duration / Fetch |
|---|---|---|---|---|---|
| ✗ | 22 | 22:36:55 | SELECT Quantity,MIN(Totalprice) FROM SALES L... | Error Code: 1140. In aggregated query without GR... | 0.000 sec |
| ✓ | 23 | 22:37:09 | SELECT MIN(Totalprice) FROM SALES LIMIT 0, 1... | 1 row(s) returned | 0.000 sec / 0.000 sec |
| ✗ | 24 | 22:38:00 | UPDATE TEST2 SET Totalprice='70' | Error Code: 1146. Table 'netmusicshop.test2' does... | 0.000 sec |

3.使用了GROUP BY则无法更新



```
CREATE VIEW TEST3
AS
SELECT AlbumID
FROM Sales
GROUP BY AlbumID;

UPDATE TEST3 SET AlbumID='000'
WHERE AlbumID='18'
```

| | # | Time | Action | Message | Duration / Fetch |
|---|---|---|---|---|---|
| ✓ | 26 | 22:51:22 | SELECT AlbumID FROM Sales GROUP BY AlbumI... | 6 row(s) returned | 0.000 sec / 0.000 sec |
| ✓ | 27 | 22:51:31 | CREATE VIEW TEST3 AS SELECT AlbumID FRO... | 0 row(s) affected | 0.000 sec |
| ✗ | 28 | 22:52:24 | UPDATE TEST3 SET AlbumID='000' WHERE Alb... | Error Code: 1288. The target table TEST3 of the U... | 0.000 sec |

## *附sql代码

```
USE NetMusicShop;

UPDATE Users SET UserAdvancePayMent=UserAdvancePayMent-200
WHERE UserName IN
(SELECT DISTINCT UserName
FROM Orders
WHERE OrderID IN
 (SELECT OrderID
 FROM Sales
 WHERE TotalPrice>=200
 )
);

CREATE TABLE Sales_Statis(
AlbumName varchar(64) not null,
Quantity  int null,
TotalPrice numeric(10, 2) null
);

INSERT INTO Sales_Statis(AlbumName,Quantity,TotalPrice)
SELECT  AlbumName,SUM(Quantity),SUM(TotalPrice)
FROM Sales,Albums
WHERE Albums.AlbumID=Sales.AlbumID
GROUP BY AlbumName;

CREATE VIEW SongByAlbumName
AS
SELECT Songs.*,AlbumName
FROM Songs,Albums
WHERE AlbumName='F.ONE' AND Songs.AlbumID=Albums.AlbumID
ORDER BY SongNumber ASC;
```

```
UPDATE SongByAlbumName SET SongTitle='我就想改改名字看看能不能成功'
WHERE SongID=6;
UPDATE SongByAlbumName SET AlbumName='我还想改一下专辑试试'
WHERE SongID=6;


CREATE VIEW TEST1
AS
SELECT AlbumIssueCompany
FROM Albums;

UPDATE TEST1 SET AlbumIssueCompany='时代峰峻'
WHERE AlbumIssueCompany='倒闭';

CREATE VIEW TEST2
AS
SELECT MIN(Totalprice)
FROM SALES;

UPDATE TEST2 SET Totalprice='70';

CREATE VIEW TEST3
AS
SELECT AlbumID
FROM Sales
GROUP BY AlbumID;

UPDATE TEST3 SET AlbumID='000'
WHERE AlbumID='18';

DROP VIEW TEST1;
DROP VIEW TEST2;
DROP VIEW TEST3;
```