西南财经大学

Southwestern University of Finance and Economics

数据库综合实验

姓　　名：	李远航
学　　号：	41933076
学　　院：	统计学院
年级专业：	2019 级统计学
指导教师：	陈久强

目录

第 2 题

2.1 问题

创建视图。查询位于 3 楼、共需接待 11 人的当日空闲、且相邻的标准间。

2.2 解答

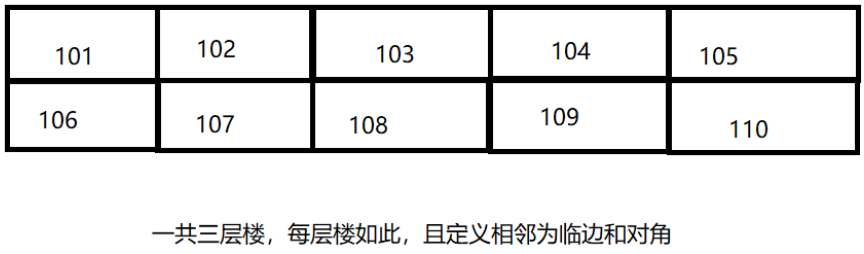

图 1:定义相邻房间

相邻房间的定义如上，由此可通过存储过程生成相邻房间，算法为除 101、105、106、110 外其他房间分别减 5、减 4、减 6、减 1、加 1、加 4、加 5、加 6 后是否真实存在，101 等特殊情况单独写。

标准间默认容纳最大人数相同。

2.3 代码

```
1.  CREATE VIEW V1
2.  AS
3.  SELECT M.ROOM_ID,SUM(A.ROOM_MAX) * 2 - COUNT(A.ROOM_MAX) * SUM(A.ROOM_MAX) /
     COUNT(A.ROOM_MAX) '最大人数' FROM ROOM_INFO M
4.      LEFT JOIN ROOM_TYPE A ON A.ROOM_TYPE_ID = M.ROOM_TYPE_ID
5.      LEFT JOIN ROOM_CONNECT ON ROOM_CONNECT.ROOM_CONNECT_ID_LEFT = M.ROOM_ID
6.      LEFT JOIN ROOM_INFO N ON N.ROOM_ID = ROOM_CONNECT.ROOM_CONNECT_ID_RIGHT
7.      LEFT JOIN ROOM_TYPE B ON B.ROOM_TYPE_ID = N.ROOM_TYPE_ID
8.  WHERE
9.      M.ROOM_STATUS_ID = 1 AND N.ROOM_STATUS_ID = 1 AND
10.     LEFT(M.ROOM_LOCATION,1) = 3 AND
11.     A.ROOM_TYPE_ID = 3 AND B.ROOM_TYPE_ID = 3
12. GROUP BY M.ROOM_ID HAVING SUM(A.ROOM_MAX) * 2 - COUNT(A.ROOM_MAX) * SUM(A.RO
    OM_MAX) / COUNT(A.ROOM_MAX) >= 11;
```

2.4 运行结果

	ROOM_ID	最大人数
1	301	6
2	302	10
3	303	8
4	304	6
5	305	2
6	306	6
7	307	10
8	308	8

图 2.1: 11 人太多没有结果，将 11 人替换为 0 后的结果

第 3 题

3.1 问题

创建存贮过程，实现团队入住登记功能，主要记录团队成员姓名、证件号码、房间号、计划入住天数。

3.2 代码

```
1.  CREATE PROCEDURE P_GROUP_CHECKIN
2.      @NAME VARCHAR(20),
3.      @IDCARD VARCHAR(18),
4.      @ROOM_ID INT,
5.      @DAY INT
6.  AS
7.  BEGIN
8.      DECLARE @GROUP_CHECKIN_ID INT;
9.      IF OBJECT_ID('GROUP_CHECKIN') IS NULL
10.     BEGIN
11.         CREATE TABLE GROUP_CHECKIN(
12.         GROUP_CHECKIN_ID INT PRIMARY KEY,
13.         GROUP_CHECKIN_NAME VARCHAR(20) NOT NULL,
14.         GROUP_CHECKIN_IDCARD VARCHAR(18) NOT NULL,
15.         GROUP_CHECKIN_ROOM_ID INT FOREIGN KEY REFERENCES ROOM_INFO,
16.         GROUP_CHECKIN_DAY INT CHECK(GROUP_CHECKIN_DAY>0) )
17.         SET @GROUP_CHECKIN_ID = 1;
18.     END
19.     ELSE
20.     BEGIN
21.         SELECT @GROUP_CHECKIN_ID = MAX(GROUP_CHECKIN_ID) + 1 FROM GROUP_CHEC
    KIN;
22.     END
23.     IF  @NAME IS NOT NULL AND LEN(@IDCARD) = 18 AND @ROOM_ID IN (SELECT @ROO
    M_ID FROM ROOM_INFO WHERE ROOM_STATUS_ID = 1)
24.     AND @DAY > 0
25.     BEGIN
26.         INSERT INTO GROUP_CHECKIN VALUES(@GROUP_CHECKIN_ID,@NAME,@IDCARD,@RO
    OM_ID,@DAY);
27.     END
28.     ELSE
29.     BEGIN
30.         SELECT '输入参数有问题！';
31.     END
32. END
```

3.3 运行结果

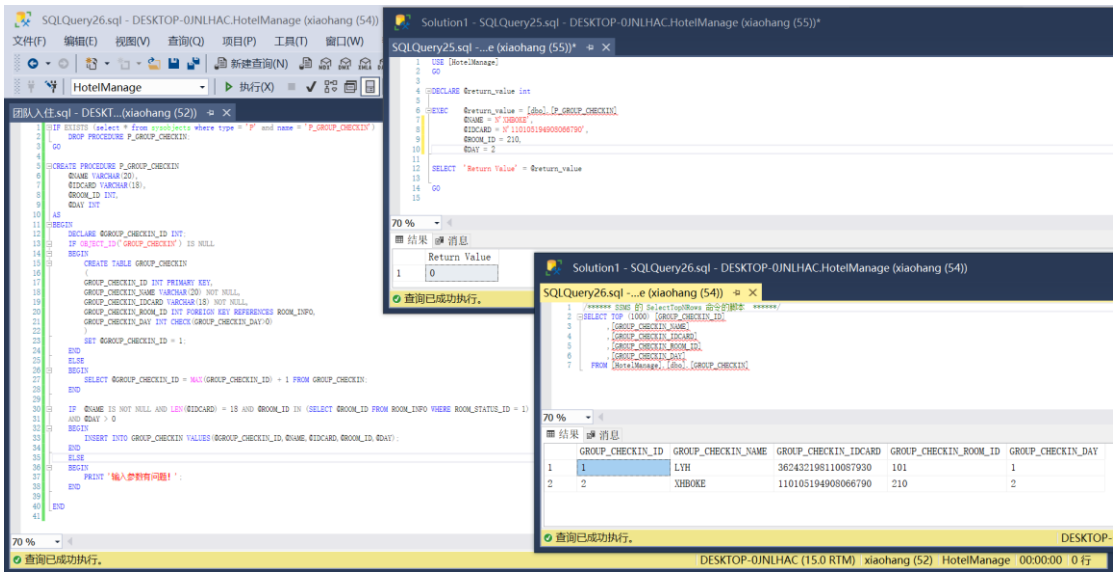

图 3.1:团队入住功能截图

第 4 题

4.1 问题

创建存贮过程，实现换房功能，记录换房前入住房间的退房日期和时间。

4.2 代码

```
1.  CREATE PROCEDURE P_CHANGE_ROOM
2.      @CHECKIN_ID INT, /** 入住号 */
3.      @ROOM_ID INT,  /** 准备更换的房间号 */
4.      @CHANGE_DATE DATE /** 更换的日期 */
5.  AS
6.  BEGIN
7.      DECLARE @CHANGE_ID INT; /** 换房标识号 */
8.      DECLARE @CHANGE_BEFORE_ROOM_ID INT; /** 换房前的房间号 */
9.      DECLARE @CHANGE_BEFORE_DATE DATE; /** 换房前的入住日期 */
10.     SET @CHANGE_BEFORE_ROOM_ID = (SELECT ROOM_ID FROM CHECKIN WHERE CHECKIN_
    ID = @CHECKIN_ID);
11.     SET @CHANGE_BEFORE_DATE = (SELECT CHECKIN_START_TIME FROM CHECKIN WHERE
    CHECKIN_ID = @CHECKIN_ID);
12.
13.     IF ((SELECT ROOM_STATUS_ID FROM ROOM_INFO WHERE ROOM_ID = @ROOM_ID) = 1)

14.     BEGIN
15.         UPDATE ROOM_INFO SET ROOM_STATUS_ID = 2 WHERE ROOM_ID = (SELECT ROOM
    _ID FROM CHECKIN WHERE CHECKIN_ID = @CHECKIN_ID);
16.         UPDATE CHECKIN SET ROOM_ID = @ROOM_ID WHERE CHECKIN_ID = @CHECKIN_ID
    ;
17.         UPDATE ROOM_INFO SET ROOM_STATUS_ID = 4 WHERE ROOM_ID = @ROOM_ID;
18.         SET @CHANGE_ID = (SELECT MAX(CHANGE_ID) FROM CHANGE_ROOM);
19.         IF @CHANGE_ID IS NULL
20.         BEGIN SET @CHANGE_ID = 1; END
21.         ELSE
22.         BEGIN SET @CHANGE_ID = @CHANGE_ID + 1; END
23.         INSERT INTO CHANGE_ROOM VALUES(@CHANGE_ID,@CHECKIN_ID,@CHANGE_BEFORE
    _ROOM_ID,@ROOM_ID,@CHANGE_DATE);
24.     END
25. END
```

4.3 运行结果

结果 消息

	CHANGE_ID	CHECKIN_ID	BEFORE_ROOM_ID	AFTER_ROOM_ID	CHANGE_TIME
1	1	1	307	101	2021-01-01

图 4.1:换房后换房记录表记录

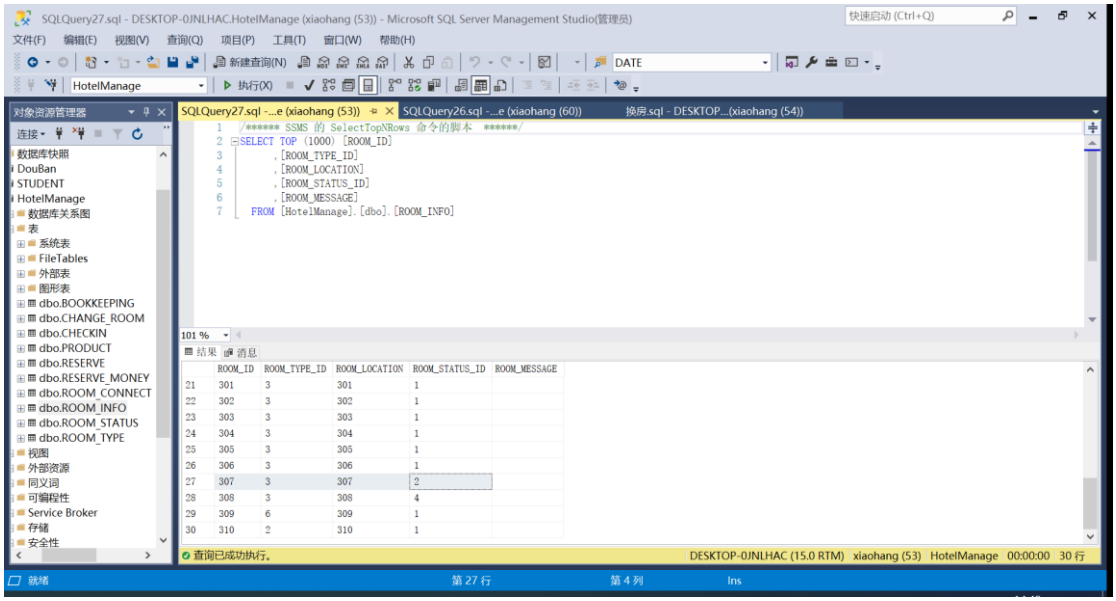

图 4.2:换房后原房间预留新房间占用

第 5 题

5.1 问题

创建存贮过程，实现退房功能，计算实际入住天数和总费用（房费和记账消费），显示详细费用清单（包括换房涉及到的不同房间费用、记账消费），记录退房日期和时间。

5.2 代码

```
IF EXISTS (select * from sysobjects where type = 'P' and name = 'P_CHECKOUT')
    DROP PROCEDURE P_CHECKOUT;
GO
CREATE PROCEDURE P_CHECKOUT @CHECKIN_ID INT
AS
BEGIN
    DECLARE @True_Day INT;
    DECLARE @Start DATE;
    DECLARE @Last DATE;
SET @Start = (SELECT CHECKIN_START_TIME FROM CHECKIN WHERE CHECKIN_ID = @CHECKIN_ID);
SET @Last = GETDATE();
SET @True_Day = DATEDIFF(day,@Start,@Last);
CREATE TABLE #Temp
(
    PRICE_NAME VARCHAR(50),
    NUMBER INT,
    PRICE MONEY
)
IF @CHECKIN_ID IN (SELECT CHECKIN_ID FROM CHANGE_ROOM)
BEGIN

    /** 计算第一次换房前的价格 */
    DECLARE @FIRST_BEFORE_ROOM_ID INT;
    DECLARE @FIRST_AFTER_ROOM_ID INT;
    DECLARE @FIRST_START_TIME DATE;
    DECLARE @FIRST_AFTER_TIME DATE;
    DECLARE @FIRST_PRICE MONEY;
    DECLARE @FIRST_DAY INT;
    SET @FIRST_BEFORE_ROOM_ID = (SELECT BEFORE_ROOM_ID FROM CHANGE_ROOM WHERE CHANGE_ID = (SELECT MIN(CHANGE_ID) FROM CHANGE_ROOM));
    SET @FIRST_AFTER_ROOM_ID = (SELECT AFTER_ROOM_ID FROM CHANGE_ROOM WHERE CHANGE_ID = (SELECT MIN(CHANGE_ID) FROM CHANGE_ROOM));
    SET @FIRST_START_TIME = (SELECT CHECKIN_START_TIME FROM CHECKIN WHERE CHECKIN_ID = @CHECKIN_ID);
```

```
32.     SET @FIRST_AFTER_TIME = (SELECT CHANGE_TIME FROM CHANGE_ROOM WHERE CHANG
    E_ID = (SELECT MIN(CHANGE_ID) FROM CHANGE_ROOM));
33.
34.     SET @FIRST_PRICE = (SELECT ROOM_PRICE FROM ROOM_INFO
35.         LEFT JOIN ROOM_TYPE ON ROOM_INFO.ROOM_TYPE_ID = ROOM_TYPE.ROOM_TYPE_
    ID
36.     WHERE ROOM_ID = @FIRST_BEFORE_ROOM_ID);
37.
38.     SET @FIRST_DAY = DATEDIFF(DD,@FIRST_START_TIME,@FIRST_AFTER_TIME);
39.
40.     INSERT INTO #Temp VALUES (CONCAT('[',@FIRST_BEFORE_ROOM_ID,']',@FIRST_ST
    ART_TIME,'->',@FIRST_AFTER_TIME,'[',@FIRST_AFTER_ROOM_ID,']'),@FIRST_DAY,@FI
    RST_PRICE);
41.
42.     /** 计算换房期间消费 */
43.     SELECT * INTO #TT
44.     FROM
45.     (
46.         SELECT A.CHANGE_ID,A.AFTER_ROOM_ID,B.ROOM_ID,A.CHANGE_TIME,A.ROOM_PR
    ICE,B.CHANGE_TIME AS 'AFTER_TIME' FROM
47.         (
48.         SELECT CHANGE_ID,AFTER_ROOM_ID,CHANGE_TIME,ROOM_PRICE FROM CHANGE_RO
    OM
49.             LEFT JOIN ROOM_INFO ON CHANGE_ROOM.AFTER_ROOM_ID = ROOM_INFO.ROO
    M_ID
50.             LEFT JOIN ROOM_TYPE ON ROOM_INFO.ROOM_TYPE_ID = ROOM_TYPE.ROOM_T
    YPE_ID
51.         )A
52.         LEFT JOIN
53.         (
54.         SELECT CHANGE_ID,ROOM_ID,CHANGE_TIME FROM CHANGE_ROOM
55.             LEFT JOIN ROOM_INFO ON CHANGE_ROOM.AFTER_ROOM_ID = ROOM_INFO.ROO
    M_ID
56.             LEFT JOIN ROOM_TYPE ON ROOM_INFO.ROOM_TYPE_ID = ROOM_TYPE.ROOM_T
    YPE_ID
57.         )B
58.         ON A.CHANGE_ID = B.CHANGE_ID - 1
59.         WHERE B.CHANGE_ID IS NOT NULL
60.     ) AS aaa;
61.
62.
63.     SELECT * INTO #TTT
64.     FROM
65.     (
```

```
        SELECT ROW_NUMBER() OVER (ORDER BY #TT.CHANGE_ID ASC) AS XUHAO,#TT.* FROM #TT
    ) AS BBB;


    /** 遍历表中每一行 */
    DECLARE @i INT;
    DECLARE @max INT;
    SET @i = 1;
    SET @max = (SELECT COUNT(*) FROM #TTT);


    WHILE @i <= @max
    BEGIN

        DECLARE @BEFORE_ROOM_ID INT;
        DECLARE @AFTER_ROOM_ID INT;
        DECLARE @BEFORE_TIME DATE;
        DECLARE @AFTER_TIME DATE;
        DECLARE @PRICE MONEY;
        DECLARE @DAY INT;


        SET @BEFORE_ROOM_ID = (SELECT AFTER_ROOM_ID FROM #TTT WHERE CHANGE_ID = @i);
        SET @AFTER_ROOM_ID = (SELECT ROOM_ID FROM #TTT WHERE CHANGE_ID = @i);
        SET @BEFORE_TIME = (SELECT CHANGE_TIME FROM #TTT WHERE CHANGE_ID = @i);
        SET @AFTER_TIME = (SELECT AFTER_TIME FROM #TTT WHERE CHANGE_ID = @i);
        SET @PRICE = (SELECT ROOM_PRICE FROM #TTT WHERE CHANGE_ID = @i);
        SET @DAY = DATEDIFF(DD,@BEFORE_TIME,@AFTER_TIME);

        INSERT INTO #Temp VALUES (CONCAT('[',@BEFORE_ROOM_ID,']',@BEFORE_TIME,'->',@AFTER_TIME,'[',@AFTER_ROOM_ID,']'),@DAY,@PRICE);

        SET @i = @i + 1;
    END


     /** 最后一次换完房到结房 */
     DECLARE @LAST_ROOM_ID INT;
```

```
104.     DECLARE @LAST_START_TIME DATE;
105.     DECLARE @LAST_ROOM_PRICE MONEY;
106.     DECLARE @LAST_END_TIME DATE;
107.     DECLARE @LAST_DAY INT;
108.
109.     SET @LAST_ROOM_ID = (SELECT ROOM_ID FROM #TT WHERE CHANGE_ID = (SELECT
    MAX(CHANGE_ID) FROM #TT));
110.     SET @LAST_START_TIME = (SELECT AFTER_TIME FROM #TT WHERE CHANGE_ID = (S
    ELECT MAX(CHANGE_ID) FROM #TT));
111.     SET @LAST_ROOM_PRICE = (
112.         SELECT ROOM_PRICE FROM ROOM_INFO
113.             LEFT JOIN ROOM_TYPE ON ROOM_INFO.ROOM_TYPE_ID = ROOM_TYPE.ROOM_
    TYPE_ID
114.         WHERE ROOM_ID = @LAST_ROOM_ID
115.     );
116.     SET @LAST_END_TIME = GETDATE();
117.     SET @LAST_DAY = DATEDIFF(DD,@LAST_START_TIME,@LAST_END_TIME);
118.
119.     INSERT INTO #Temp VALUES (CONCAT('[',@LAST_ROOM_ID,']',@LAST_START_TIME
    ,'->',@LAST_END_TIME),@LAST_DAY,@LAST_ROOM_PRICE);
120.
121. END
122. ELSE
123. BEGIN
124. /** 没有换房 */
125.
126. DECLARE @B_ROOM_ID INT;
127.
128. DECLARE @B_TIME DATE;
129. DECLARE @N_TIME DATE;
130.
131. DECLARE @N_PRICE MONEY;
132. DECLARE @N_DAY INT;
133.
134. SET @B_ROOM_ID = (SELECT ROOM_ID FROM CHECKIN WHERE CHECKIN_ID = @CHECKIN_I
    D);
135.
136. SET @B_TIME = (SELECT CHECKIN_START_TIME FROM CHECKIN WHERE CHECKIN_ID = @C
    HECKIN_ID);
137. SET @N_TIME = GETDATE();
138.
139. SET @N_PRICE = (
140. SELECT ROOM_PRICE FROM CHECKIN
141.     LEFT JOIN ROOM_INFO ON CHECKIN.ROOM_ID = ROOM_INFO.ROOM_ID
```

```
142.     LEFT JOIN ROOM_TYPE ON ROOM_INFO.ROOM_TYPE_ID = ROOM_TYPE.ROOM_TYPE_ID
143. WHERE CHECKIN_ID = @CHECKIN_ID
144. )
145.
146. SET @DAY = DATEDIFF(DD,@B_TIME,@N_TIME);
147.
148. INSERT INTO #Temp VALUES (CONCAT('[',@B_ROOM_ID,']',@B_TIME,'->',@N_TIME),@DAY,@N_PRICE);
149. END
150.
151. /** 附加产品 */
152. DECLARE @M INT;
153. DECLARE @P_MAX INT;
154. SET @M = 1;
155. SET @P_MAX = (SELECT COUNT(*) FROM BOOKKEEPING WHERE CHECKIN_ID = @CHECKIN_ID);
156.
157. SELECT * INTO #TA
158. FROM
159. (
160.     SELECT ROW_NUMBER() OVER (ORDER BY BOOKKEEPING.BOOKKEEPING_ID ASC) AS XUHAO,BOOKKEEPING.* FROM BOOKKEEPING
161. ) AS BBB;
162.
163. WHILE @M <= @P_MAX
164. BEGIN
165.     DECLARE @PRODUCE_ID INT;
166.     DECLARE @PRODUCE_NAME VARCHAR(20);
167.     DECLARE @PRODUCE_NUMBER INT;
168.     DECLARE @PRODUCE_PRICE MONEY;
169.
170.     SET @PRODUCE_ID = ( SELECT #TA.PRODUCT_ID FROM #TA LEFT JOIN PRODUCT ON #TA.PRODUCT_ID = PRODUCT.PRODUCT_ID WHERE XUHAO = @M);
171.     SET @PRODUCE_NAME = (   SELECT PRODUCT_NAME FROM #TA LEFT JOIN PRODUCT ON #TA.PRODUCT_ID = PRODUCT.PRODUCT_ID WHERE XUHAO = @M);
172.     SET @PRODUCE_NUMBER = ( SELECT NUMBER FROM #TA LEFT JOIN PRODUCT ON #TA.PRODUCT_ID = PRODUCT.PRODUCT_ID WHERE XUHAO = @M);
173.     SET @PRODUCE_PRICE = (  SELECT PRODUCT.PRODUCT_PRICE FROM #TA LEFT JOIN PRODUCT ON #TA.PRODUCT_ID = PRODUCT.PRODUCT_ID WHERE XUHAO = @M);
174.
175.     INSERT INTO #Temp VALUES (CONCAT('[',@PRODUCE_ID,']',@PRODUCE_NAME),@PRODUCE_NUMBER,@PRODUCE_PRICE);
176.     SET @M = @M + 1;
```

```
177. END
178.
179.
180. /** 计算总消费 */
181. DECLARE @_ALL MONEY;
182.
183. SET @_ALL = (SELECT SUM(#Temp.NUMBER * #Temp.PRICE) FROM #Temp);
184. INSERT INTO #Temp VALUES ('[入住天数]',@True_Day,NULL);
185. INSERT INTO #Temp VALUES ('[总消费]',1,@_ALL);
186. SELECT * FROM #Temp;
187.
188. /** 记录到退房表 */
189. DECLARE @CHECKOUT_MAX_ID INT;
190. SET @CHECKOUT_MAX_ID = (SELECT MAX(CHECKOUT_ID) FROM CHECKOUT);
191. IF @CHECKOUT_MAX_ID IS NULL
192. BEGIN
193.     INSERT INTO CHECKOUT VALUES (1,@CHECKIN_ID,GETDATE());
194. END
195. ELSE
196. BEGIN
197.     INSERT INTO CHECKOUT VALUES (@CHECKOUT_MAX_ID + 1,@CHECKIN_ID,GETDATE()
    );
198. END
199.
200. /** 更新到房间信息表状态*/
201. UPDATE ROOM_INFO SET ROOM_STATUS_ID = 2 WHERE ROOM_ID = (SELECT ROOM_ID FRO
    M CHECKIN WHERE CHECKIN_ID = @CHECKIN_ID);
202.
203. END
```

5.3 运行结果

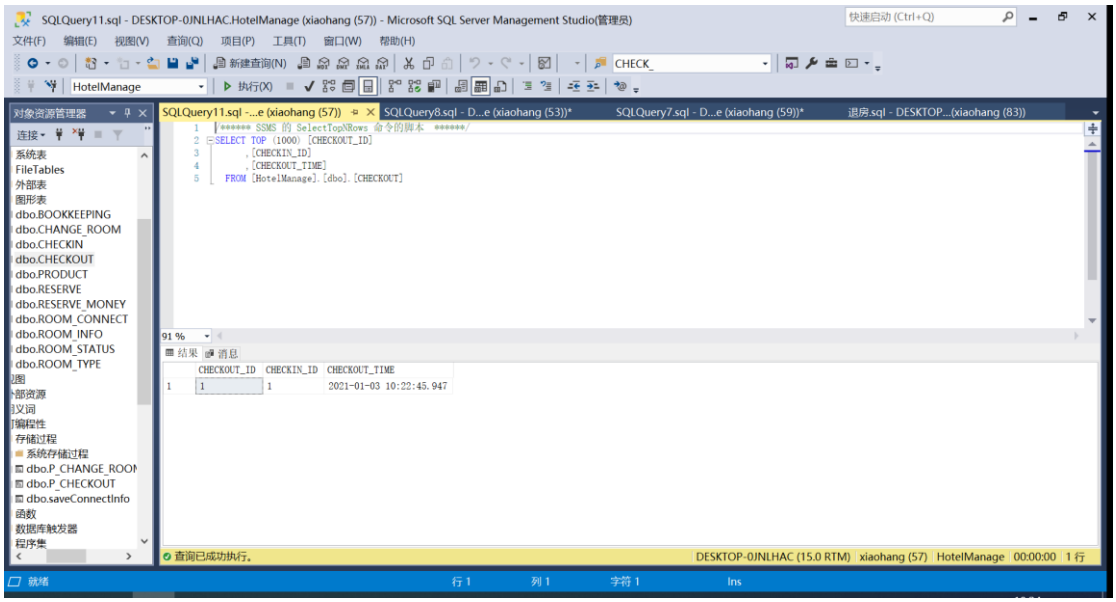

图 5.1:退房记录到退房记录表中

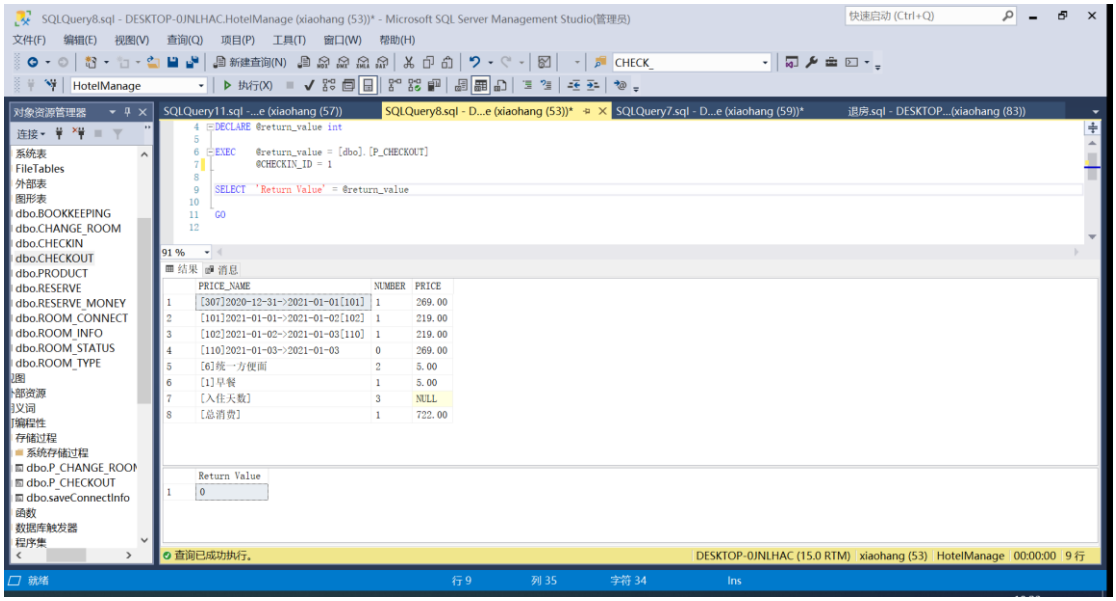

图 5.2:有换房的退房总清单

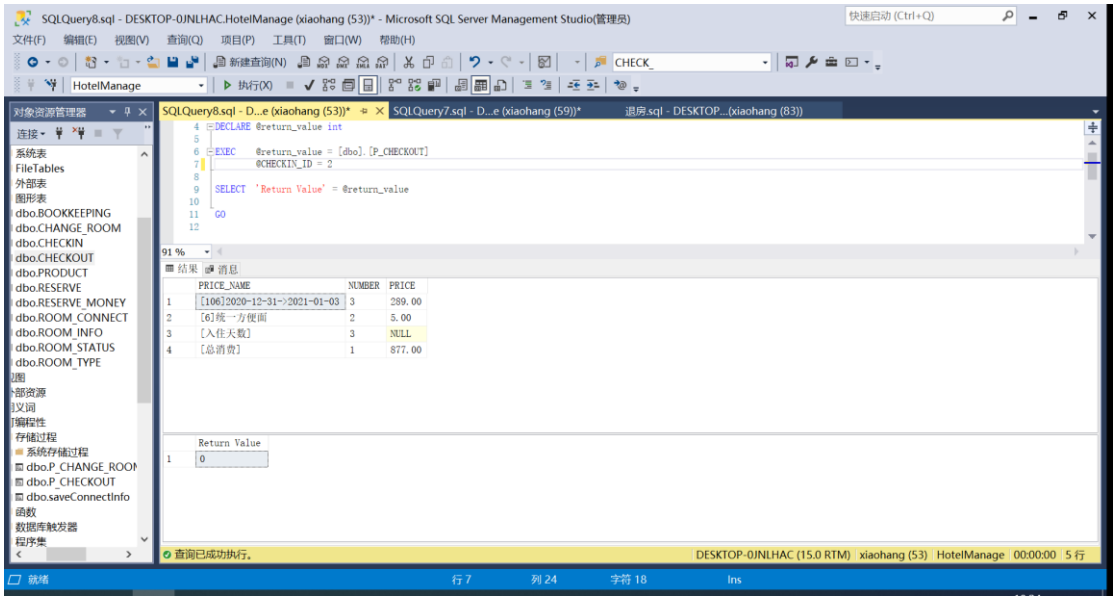

图 5.3:无换房的退房总清单

第 6 题

6.1 问题

创建存贮过程，根据基础数据，自动生成当日、未来 15 日房态信息。 房态表参考列名：顺序号、房间号、F0(当日房态)、F1、F2、F3、...、F15(未来 1——15 日房态)。

6.2 代码

```
IF EXISTS (select * from sysobjects where type = 'P' and name = 'FangTai')
    DROP PROCEDURE FangTai;
GO
CREATE PROCEDURE FangTai
AS
BEGIN
    CREATE TABLE #Temp
    (
        ShunXu INT,
        ROOM_ID INT,
        F0 VARCHAR(4),
        F1 VARCHAR(4),
        F2 VARCHAR(4),
        F3 VARCHAR(4),
        F4 VARCHAR(4),
        F5 VARCHAR(4),
        F6 VARCHAR(4),
        F7 VARCHAR(4),
        F8 VARCHAR(4),
        F9 VARCHAR(4),
        F10 VARCHAR(4),
        F11 VARCHAR(4),
        F12 VARCHAR(4),
        F13 VARCHAR(4),
        F14 VARCHAR(4),
        F15 VARCHAR(4),
    );
    DECLARE @A INT;
    DECLARE @B INT;
    DECLARE @F INT;
    DECLARE @NOW DATE;
    DECLARE @ShunXu INT;

    SET @A = 100;
    SET @B = 1;
    SET @F = 0;
```

```
37.     SET @ShunXu = 1;
38.     WHILE ( @A <= (SELECT MAX(ROOM_ID) FROM ROOM_INFO) )
39.     BEGIN
40.         WHILE( @B <= (SELECT RIGHT(MAX(ROOM_ID),2) FROM ROOM_INFO ))
41.         BEGIN
42.             INSERT INTO #Temp VALUES(@ShunXu,@A+@B,NULL,NULL,NULL,NULL,NULL,
    NULL,NULL,NULL,NULL,NULL,NULL,NULL,NULL,NULL,NULL,NULL,NULL);
43.             SET @ShunXu = @ShunXu + 1;
44.
45.             WHILE( @F <= 15)
46.             BEGIN
47.                 SET @NOW = GETDATE();
48.                 DECLARE @SQL VARCHAR(2000);
49.                 /** 入住表信息 */
50.                 IF (SELECT COUNT(*) FROM CHECKIN WHERE ROOM_ID = @A + @B AND
     DATEADD(DAY,PLAN_TIME,CHECKIN_START_TIME) >= DATEADD(DD,@F,GETDATE())) = 1
51.                 BEGIN
52.                     SET @SQL = CONCAT('UPDATE #Temp SET F',CAST(@F AS varcha
    r),'=''占用''','WHERE ROOM_ID = ',CAST((@A+@B) AS varchar) );
53.                     EXEC(@SQL);
54.                 END
55.                 /** 换房 预留信息 */ /** 预定 预留信息 */
56.                 ELSE IF( SELECT COUNT(*) FROM CHANGE_ROOM WHERE BEFORE_ROOM_
    ID = @A+@B AND CHANGE_TIME >= DATEADD(DD,@F,GETDATE()) ) = 1
57.                 OR (SELECT COUNT(*) FROM RESERVE WHERE ROOM_ID = @A+@B AND D
    ATEADD(DD,RESERVE_LAST_TIME,RESERVE_START_TIME) >= DATEADD(DD,@F,GETDATE())
    ) = 1
58.                 BEGIN
59.                     SET @SQL = CONCAT('UPDATE #Temp SET F',CAST(@F AS varcha
    r),'=''预留''','WHERE ROOM_ID = ',CAST((@A+@B) AS varchar) );
60.                     EXEC(@SQL);
61.                 END
62.
63.
64.                 /** 其余 空闲信息 */
65.                 ELSE
66.                 BEGIN
67.                     SET @SQL = CONCAT('UPDATE #Temp SET F',CAST(@F AS varcha
    r),'=''空闲''','WHERE ROOM_ID = ',CAST((@A+@B) AS varchar) );
68.                     EXEC(@SQL);
69.                 END
70.
71.
```

```
72.                 SET @F = @F + 1;
73.             END
74.             SET @F = 0;
75.             SET @B = @B + 1;
76.         END
77.         SET @B = 1;
78.         SET @A = @A + 100;
79.     END
80.     SELECT * FROM #Temp;
81. END
```

6.3 运行结果

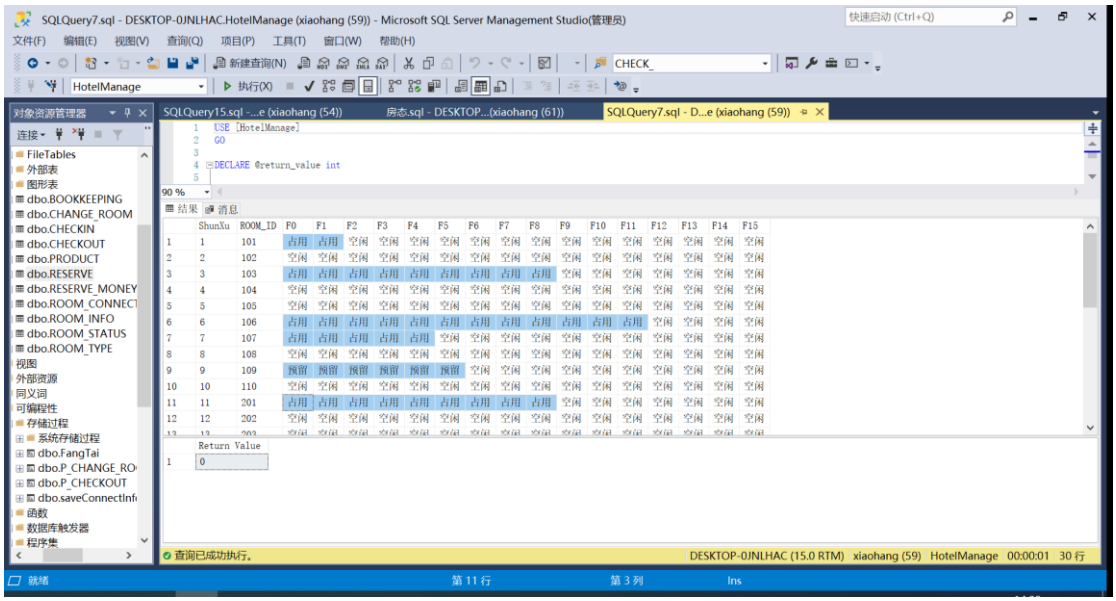

图 6.1:房态信息

第 7 题

7.1 问题

基于预订、入住、换房、退房涉及到的基本表，建立触发器，自动更新当前房间状态。

7.2 代码

入住登记表触发器

```
IF (OBJECT_ID('T_CHECKIN', 'tr') IS NOT  NULL)
    DROP TRIGGER T_CHECKIN
GO
/** 入住表的触发器 */
CREATE TRIGGER T_CHECKIN ON CHECKIN
    FOR INSERT,UPDATE,DELETE
AS
    DECLARE @_A INT;
    DECLARE @_B INT;
    SET @_A = (SELECT COUNT(ROOM_STATUS_ID) FROM ROOM_INFO WHERE ROOM_ID IN (SELECT INSERTED.ROOM_ID FROM INSERTED));
    SET @_B = (SELECT COUNT(ROOM_STATUS_ID) FROM ROOM_INFO WHERE ROOM_ID IN (SELECT INSERTED.ROOM_ID FROM INSERTED) AND ROOM_STATUS_ID = 1);
    /** 判断入住的房间均为空闲 */
    IF @_A = @_B
    BEGIN
        /** 对于更新新的设置为 4(占用)，旧的设置为 1(空闲) */
        /** 对于插入设置为 4(占用) */
        UPDATE ROOM_INFO SET ROOM_STATUS_ID = 4 WHERE ROOM_ID IN (SELECT INSERTED.ROOM_ID  FROM INSERTED);
        /** 对于更新和删除设置为 2(预留) */
        UPDATE ROOM_INFO SET ROOM_STATUS_ID = 2 WHERE ROOM_ID IN (SELECT DELETED.ROOM_ID  FROM DELETED);
    END
    ELSE
    BEGIN
        SELECT '插入失败! ';
        ROLLBACK;
    END
```

结账表触发器

```
IF (OBJECT_ID('T_CHECKOUT', 'tr') IS NOT  NULL)
    DROP TRIGGER T_CHECKOUT
```

```
GO

CREATE TRIGGER T_CHECKOUT ON CHECKOUT
    FOR INSERT
AS
    DECLARE @_A INT;
    DECLARE @_B INT;
    SET @_A =
    (
        SELECT COUNT(ROOM_STATUS_ID) FROM ROOM_INFO WHERE ROOM_ID IN
        (SELECT ROOM_ID FROM CHECKIN WHERE CHECKIN_ID IN (SELECT INSERTED.CHECKIN_ID FROM INSERTED))
    )
    SET @_B =
    (
        SELECT COUNT(ROOM_STATUS_ID) FROM ROOM_INFO WHERE ROOM_ID IN
        (SELECT ROOM_ID FROM CHECKIN WHERE CHECKIN_ID IN (SELECT INSERTED.CHECKIN_ID FROM INSERTED))
        AND ROOM_STATUS_ID = 4

    )
    /** 退房均为占用*/
    IF(@_A = @_B)
    BEGIN
        /** 更新房间状态*/
        UPDATE ROOM_INFO SET ROOM_STATUS_ID = 2 WHERE ROOM_ID IN (
            SELECT CHECKIN.ROOM_ID  FROM INSERTED
            LEFT JOIN CHECKIN ON INSERTED.CHECKIN_ID = CHECKIN.CHECKIN_ID
        );
    END
    ELSE
    BEGIN
        SELECT '退房失败! ';
        ROLLBACK;
    END
```

预定表触发器

```
IF (OBJECT_ID('T_RESERVE', 'tr') IS NOT  NULL)
    DROP TRIGGER T_RESERVE
GO

CREATE TRIGGER T_RESERVE ON RESERVE
    FOR INSERT,UPDATE,DELETE
```

```
AS
    DECLARE @_A INT;
    DECLARE @_B INT;
    SET @_A = (SELECT COUNT(ROOM_STATUS_ID) FROM ROOM_INFO WHERE ROOM_ID IN (SELECT INSERTED.ROOM_ID FROM INSERTED));
    SET @_B = (SELECT COUNT(ROOM_STATUS_ID) FROM ROOM_INFO WHERE ROOM_ID IN (SELECT INSERTED.ROOM_ID FROM INSERTED) AND ROOM_STATUS_ID = 1);
    IF @_A = @_B
    BEGIN
        UPDATE ROOM_INFO SET ROOM_STATUS_ID = 3 WHERE ROOM_ID IN (SELECT INSERTED.ROOM_ID  FROM INSERTED);
        UPDATE ROOM_INFO SET ROOM_STATUS_ID = 1 WHERE ROOM_ID IN (SELECT DELETED.ROOM_ID  FROM DELETED);
    END
    ELSE
    BEGIN
        SELECT '插入失败';
        ROLLBACK;
    END
```

7.3 运行结果

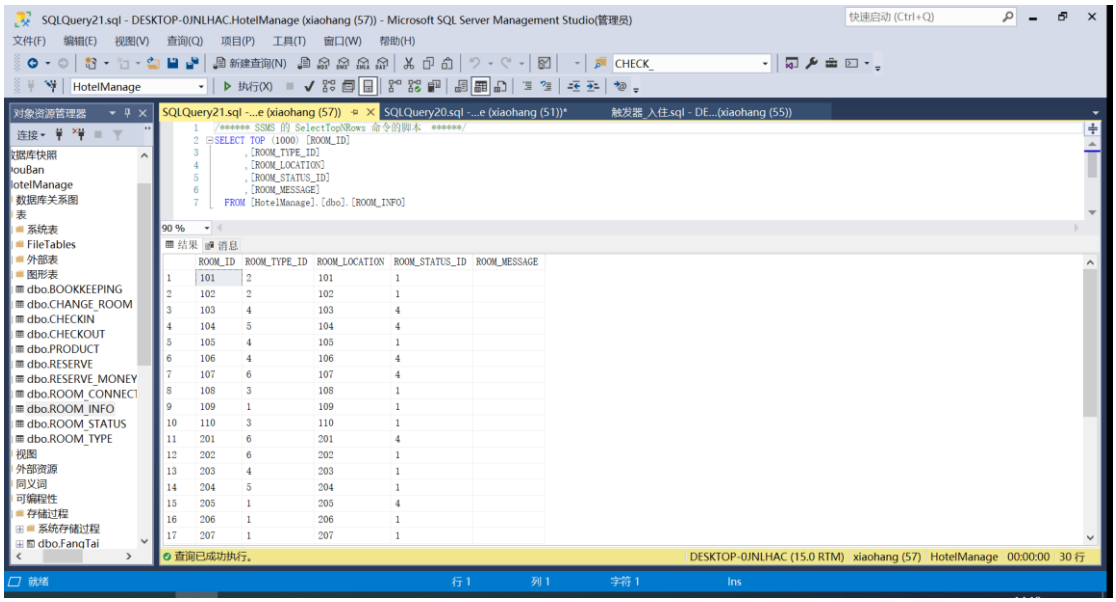

图 7.1:入住表更新房间状态

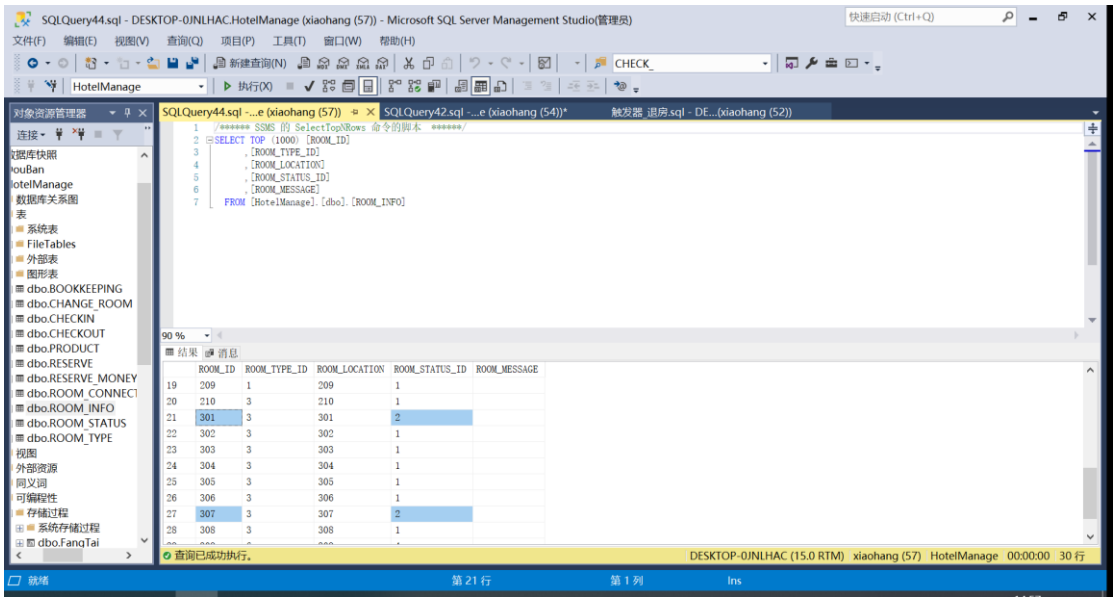

图 7.2:退房更新房间状态

第 8 题

8.1 问题

基于退房涉及到的基本表，建立触发器，自动记录入住天数大于 3 天或消费金额大于 1000 的 VIP 常 旅客消费信息（日期、姓名、证件号码、支付金额）。

8.2 代码

```
1.  IF (OBJECT_ID('T_VIP', 'tr') IS NOT  NULL)
2.      DROP TRIGGER T_VIP
3.  GO
4.  /** VIP 记录 */
5.  CREATE TRIGGER T_VIP ON CHECKOUT
6.      FOR INSERT
7.  AS
8.      DECLARE @CHECKIN_ID INT;
9.      DECLARE @TABLE table(_name varchar(100),_number INT,_price MONEY);
10.     DECLARE @XIAOFEI MONEY;
11.     DECLARE @DAY INT;
12.
13.     DECLARE @NAME VARCHAR(20);
14.     DECLARE @ID INT;
15.     DECLARE @IDCARD VARCHAR(18);
16.
17.     SET @CHECKIN_ID = (SELECT CHECKIN_ID FROM INSERTED);
18.     INSERT INTO @TABLE EXEC P_CHECKOUT @CHECKIN_ID;
19.     SET @XIAOFEI = (SELECT MAX(_price) FROM @TABLE);
20.     SET @DAY = (SELECT _number FROM @TABLE WHERE _name = '[入住天数]');
21.     SET @NAME = (SELECT NAME_CONSUMER FROM CHECKIN WHERE CHECKIN_ID = @CHECK
    IN_ID);
22.     SET @IDCARD = (SELECT CONTACT_CONSUMER FROM CHECKIN WHERE CHECKIN_ID = @
    CHECKIN_ID);
23.     SET @ID = (SELECT MAX(_ID) FROM VIP);
24.
25.     IF @ID IS NOT NULL
26.     BEGIN
27.         SET @ID = @ID + 1;
28.     END
29.     ELSE
30.     BEGIN
31.         SET @ID = 1;
32.     END
33.
34.     IF(@DAY > 3 OR @XIAOFEI > 1000 )
```

```
35.     BEGIN
36.         INSERT INTO VIP VALUES(@ID,@NAME,GETDATE(),@IDCARD,@XIAOFEI);
37.         SELECT * FROM VIP;
38.     END
```

8.3 运行截图

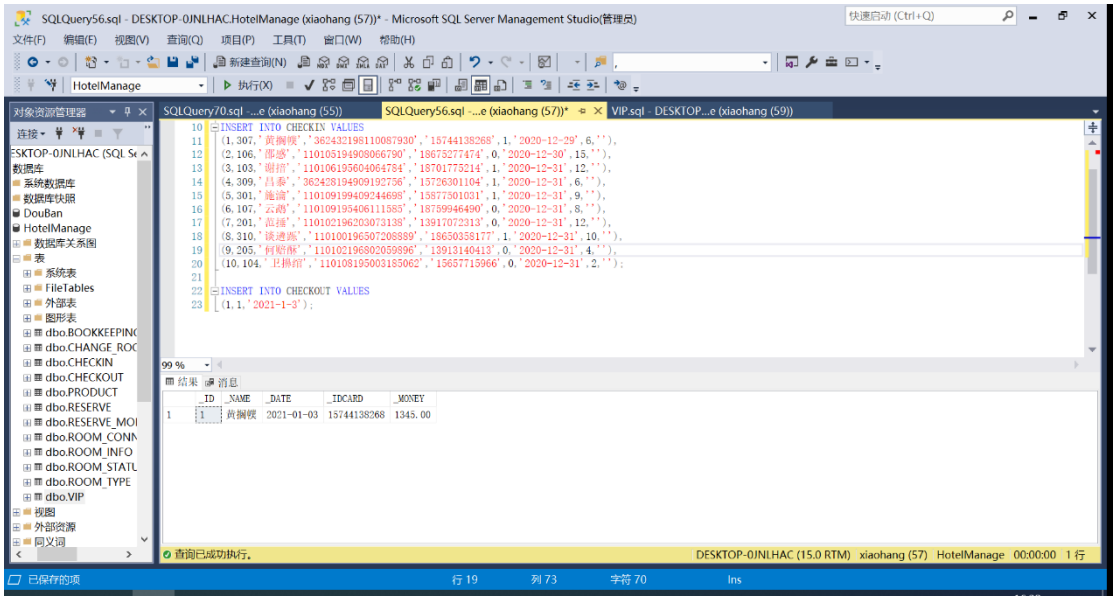

图 8.1:记录 VIP 客户截图

第 9 题

9.1 问题

创建存贮过程，当调价窗口期（如：2 个月）到来时，某些房间类型的入住率大于 90%，上浮该房 间类型的价格 20%；入住率小于 60%，价格下降 20%，且不低于 100。

9.2 代码

```
1.  IF EXISTS (select * from sysobjects where type = 'P' and name = 'TiaoJia')
2.      DROP PROCEDURE TiaoJia;
3.  GO
4.  CREATE PROCEDURE TiaoJia
5.      @start_time date
6.  AS
7.  BEGIN
8.      DECLARE @end_time date;
9.      SET @end_time = DATEADD(DAY,60,@start_time);
10.
11.     CREATE TABLE #Temp
12.     (
13.         ROOM_TYPE_ID INT,
14.         _RATE FLOAT
15.     );
16.
17. INSERT INTO #Temp
18.
19.     SELECT S.ROOM_TYPE_ID,ROUND( CAST(SUM(入住天
    数) AS FLOAT)/60,3) 'RATE'  FROM
20.     (
21.         /** 换房的入住 */
22.         SELECT ROOM_TYPE.ROOM_TYPE_ID,SUM(DATEDIFF(DD,CHECKIN_START_TIME,CHA
    NGE_TIME)) '入住天数' FROM CHECKIN
23.             LEFT JOIN ROOM_INFO ON CHECKIN.ROOM_ID = ROOM_INFO.ROOM_ID
24.             LEFT JOIN ROOM_TYPE ON ROOM_INFO.ROOM_TYPE_ID = ROOM_TYPE.ROOM_T
    YPE_ID
25.             LEFT JOIN CHANGE_ROOM ON CHANGE_ROOM.CHECKIN_ID = CHECKIN.CHECKI
    N_ID
26.             WHERE
27.         DATEADD(DAY,PLAN_TIME,CHECKIN_START_TIME) BETWEEN @start_time AND @e
    nd_time
28.         AND CHANGE_ID IS NOT NULL
29.         GROUP BY ROOM_TYPE.ROOM_TYPE_ID
30.     union all
31.         /** 没有换房的入住 */
```

```
32.         SELECT ROOM_TYPE.ROOM_TYPE_ID,SUM(PLAN_TIME) '入住天数
    ' FROM CHECKIN
33.             LEFT JOIN ROOM_INFO ON CHECKIN.ROOM_ID = ROOM_INFO.ROOM_ID
34.             LEFT JOIN ROOM_TYPE ON ROOM_INFO.ROOM_TYPE_ID = ROOM_TYPE.ROOM_T
    YPE_ID
35.             LEFT JOIN CHANGE_ROOM ON CHANGE_ROOM.CHECKIN_ID = CHECKIN.CHECKI
    N_ID
36.             WHERE
37.         DATEADD(DAY,PLAN_TIME,CHECKIN_START_TIME) BETWEEN @start_time AND @e
    nd_time
38.         AND CHANGE_ID IS NULL
39.         GROUP BY ROOM_TYPE.ROOM_TYPE_ID
40.     )S GROUP BY S.ROOM_TYPE_ID;
41.
42.
43.     DECLARE @START INT;
44.     DECLARE @END INT;
45.
46.         SELECT * FROM(
47.             SELECT row_number() OVER(ORDER BY #Temp.ROOM_TYPE_ID) as Xuhao,#
    TEMP.ROOM_TYPE_ID,_RATE from #Temp LEFT JOIN ROOM_TYPE ON #Temp.ROOM_TYPE_ID
     = ROOM_TYPE.ROOM_TYPE_ID
48.         )A
49.
50.     SET @START = (
51.         SELECT MIN(Xuhao) FROM(
52.         SELECT row_number() OVER(ORDER BY #Temp.ROOM_TYPE_ID) as Xuhao,#TEMP
    .ROOM_TYPE_ID,_RATE from #Temp LEFT JOIN ROOM_TYPE ON #Temp.ROOM_TYPE_ID = R
    OOM_TYPE.ROOM_TYPE_ID
53.         )A
54.     )
55.     SET @END = (
56.         SELECT MAX(Xuhao) FROM(
57.         SELECT row_number() OVER(ORDER BY #Temp.ROOM_TYPE_ID) as Xuhao,#TEMP
    .ROOM_TYPE_ID,_RATE from #Temp LEFT JOIN ROOM_TYPE ON #Temp.ROOM_TYPE_ID = R
    OOM_TYPE.ROOM_TYPE_ID
58.         )A
59.     )
60.
61.
62.     DECLARE @ROOM_TYPE_ID INT;
63.     DECLARE @ROOM_RATE FLOAT;
64.     DECLARE @ROOM_PRICE MONEY;
65.
```

```
    WHILE @START <= @END
    BEGIN
        SET @ROOM_TYPE_ID = (SELECT ROOM_TYPE_ID FROM(
            SELECT row_number() OVER(ORDER BY #Temp.ROOM_TYPE_ID) as Xuhao,#TEMP.ROOM_TYPE_ID,_RATE from #Temp LEFT JOIN ROOM_TYPE ON #Temp.ROOM_TYPE_ID = ROOM_TYPE.ROOM_TYPE_ID
        )A WHERE Xuhao = @START);
        SET @ROOM_RATE = (SELECT _RATE FROM(
            SELECT row_number() OVER(ORDER BY #Temp.ROOM_TYPE_ID) as Xuhao,#TEMP.ROOM_TYPE_ID,_RATE from #Temp LEFT JOIN ROOM_TYPE ON #Temp.ROOM_TYPE_ID = ROOM_TYPE.ROOM_TYPE_ID
        )A WHERE Xuhao = @START);
        SET @ROOM_PRICE = (SELECT ROOM_PRICE FROM(
            SELECT row_number() OVER(ORDER BY #Temp.ROOM_TYPE_ID) as Xuhao,#TEMP.ROOM_TYPE_ID,_RATE,ROOM_PRICE from #Temp LEFT JOIN ROOM_TYPE ON #Temp.ROOM_TYPE_ID = ROOM_TYPE.ROOM_TYPE_ID
        )A WHERE Xuhao = @START);
        IF( @ROOM_RATE > 0.9)
        BEGIN
            UPDATE ROOM_TYPE
            SET ROOM_PRICE = ROOM_PRICE * 1.2
            WHERE ROOM_TYPE_ID = @ROOM_TYPE_ID;
        END
        ELSE IF ( @ROOM_RATE <= 0.6)
        BEGIN
            IF @ROOM_PRICE * 0.8 < 100
            BEGIN
            SET @ROOM_PRICE = 125
            END
            UPDATE ROOM_TYPE
            SET ROOM_PRICE = ROOM_PRICE * 0.8
            WHERE ROOM_TYPE_ID = @ROOM_TYPE_ID;
        END

        SET @START = @START + 1;
    END

END
```

9.3 运行截图

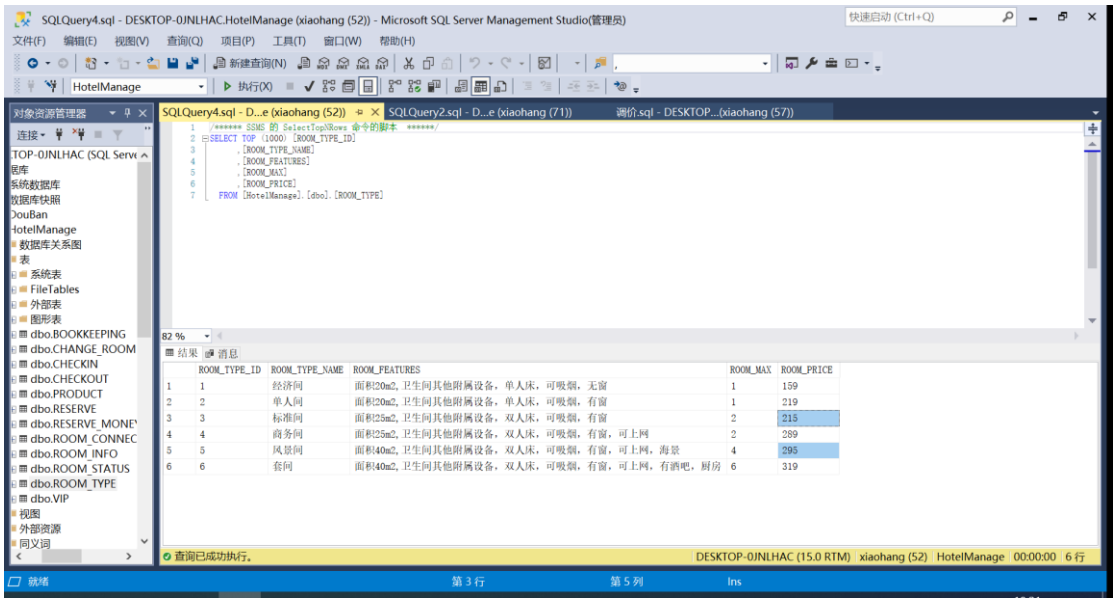

图 9.1：更新后房间信息表的房间价格

附：测试数据

入住登记表：

入住号	房间号	客户名	身份证	手机号	预定	入住时间	计 划时间	其他
1	307	黄搁幞	362432198110087930	15744138268	1	2020-12-31	5	
2	106	邵感	110105194908066790	18675277474	0	2020-12-31	15	
3	103	谢掊	110106195604064784	18701775214	1	2020-12-31	12	
4	309	昌黍	362428194909192756	15726301104	1	2020-12-31	6	
5	301	施澹	110109199409244698	15877501031	1	2020-12-31	9	
6	107	云鹇	110109195406111585	18759946490	0	2020-12-31	8	
7	201	范捶	110102196203073138	13917072313	0	2020-12-31	12	
8	310	谈遴跺	110100196507208889	18650358177	1	2020-12-31	10	
9	205	何赔酥	110102196802059896	13913140413	0	2020-12-31	4	
10	104	卫擒绾	110108195003185062	15657715966	0	2020-12-31	2	

房间信息表：

房间号	房间类型号	房间位置	房间状态号	房间额外信息
101	2	101	1	
102	2	102	1	
103	4	103	1	
104	5	104	1	
105	4	105	1	
106	4	106	1	
107	6	107	1	
108	3	108	1	
109	1	109	1	
110	3	110	1	
201	6	201	1	
202	6	202	1	
203	4	203	1	
204	5	204	1	
205	1	205	1	
206	1	206	1	
207	1	207	1	
208	5	208	1	
209	1	209	1	
210	3	210	1	
301	3	301	1	
302	3	302	1	
303	3	303	1	
304	3	304	1	
305	3	305	1	

306	3	306	1	
307	3	307	1	
308	3	308	1	
309	6	309	1	
310	2	310	1	

消费产品表：

产品号	产品名称	单价
1	早餐	5
2	午餐	10
3	晚餐	8
4	雪碧	4
5	可口可乐	4
6	统一方便面	5