

数据库原理

姓名：李婷婷

学号：071940101

班级：1619201

数据库原理

功能简介

主要操作模块

主界面

员工管理

客房管理

客户管理

订单管理

维护与报表

实现框架

dao

service

ui

后端主要代码（sql语句）

办理入住\办理预定

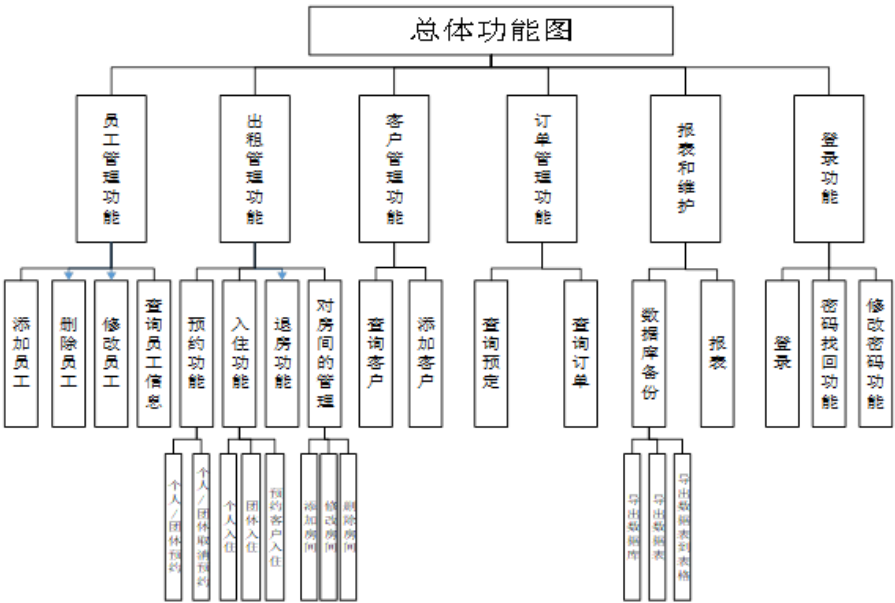
预定登记

退房

客房管理

展望

功能简介



主要操作模块


主界面



- 修改密码

MainWindow

— □ ×



LOGIN HERE

员工编号

编号

旧密码

old password

新密码

new password


确认修改





返回主界面

- 客房管理
- 员工管理
- 客户管理
- 订单管理
- 维护与报表

员工管理

- 首先是显示基本信息


1
权限: 2
*表示需要最高权

 个人信息
 查询员工*
 增添员工*
 删除员工*



1

姓名: 1
性别: 男
入职时间: 2020-01-0
权限: 2
手机号: 112233
身份证号: 112233
员工号: 10


- 表结构





sid	sname	ssex	stime	susername	spassword	srole	sidcard	sphone
主键	姓名	性别	入职时间	账号名	密码	角色	身份证号	电话号码

- sid作为识别的唯一主键
- 账号名和密码作为系统的登录识别接口, 会根据这个信息来确定staff
- role区分staff的角色, 主要是做权限控制, 简单来说, 有两种权限,
 - 一个是普通员工, 包括简单的客房查询, 订单查询, 客房预订等等, 识别号为1;
 - 另一个是管理员, 包含所有权限, 包括增加新的客房种类等等, 识别号为2;
- 对于表的操作:
 - 查询: 根据姓名查询
 - 在查询时提供删除和修改功能

- 增添：需要输入员工的全部信息

- 删除：输入sid,sname,sidcard进行删除


1
权限: 2
*表示需要最高权
搜索

 个人信息
 查询员工*
 增添员工*
 删除员工*

删除员工


选择要删除的员工:






员工编号: 员工姓名: 身份证:

确认删除

员工编号	姓名	性别	登记入职时间
------	----	----	--------

客房管理


1
权限: 2
*表示需要最高权
搜索

 客房信息
 办理入住
 预定登记...
 办理退房
 更新房源*

客...息

房型: 房间状态: ☐ 只看空闲楼层: 预计入住时间: — 价格范围: —

搜索

重置

- room字段信息

rid	rtype	rstory	rprice	rdesc	rpic
主键	类型	楼层	价格	备注	图片信息

- rid作为唯一主键
 - 类型有标准件（单、双人），大床房和总统套房
 - 楼层2,3,4
- 对于该表的操作
 - 客房查询：可以全部查询也可以只查部分表格（为办理入住做准备）注意每次查询需要先重置再查询
 - 办理入住：输入客户信息和房间信息，退房时间必须晚于当前时间。这里也有预订信息，但是这里的是必须是前面已经预定过的
 - 办理预订：个人预订不行，团队预订可以
 - 办理退房：输入相关信息，可以正常运行，要求办理退房时，必须在checkin_表格里有内容；否则提示“没有入住信息”
 - 更新客房信息：增加修改都正常，可以像修改staff里面那样修改。修改不能修改图片位置。。。改代码参照
- 其他表格：
 - booking_client, booking_team, checkin_client, checkin_client。这四个表格互相制约，指在插入的时候会根据房间和起止时间来判断是否已经预约过了
 - client，客户表，用来存储住过的客户信息（包括预定的信息）
 - team，团队表，用来存储所有的团队客户信息，同上

客户管理

MainWindow

— □ ×

1

权限：2

*表示需要最高权

搜索

查询和删...

增加客户

查...户

对象：

☐ 个人
 ☐ 团队

客户姓名/团队名和

搜索员工姓名

入住次数>=:

入住次数

立即检索

删除

姓名	性别	注册时间	年龄
----	----	------	----

- client

维护与报表

1

权限: 2

*表示需要最高权

搜索

数据库备份

报表查看

数...份

关于表名

导出路径:

导出路径

浏览

导出数据库 (.sql)

表名:

请选择...

导出路径:

导出路径

浏览

导出数据表 (.sql)

表名:

请选择...

导出路径:

导出路径

浏览

导入数据表 (.csv)

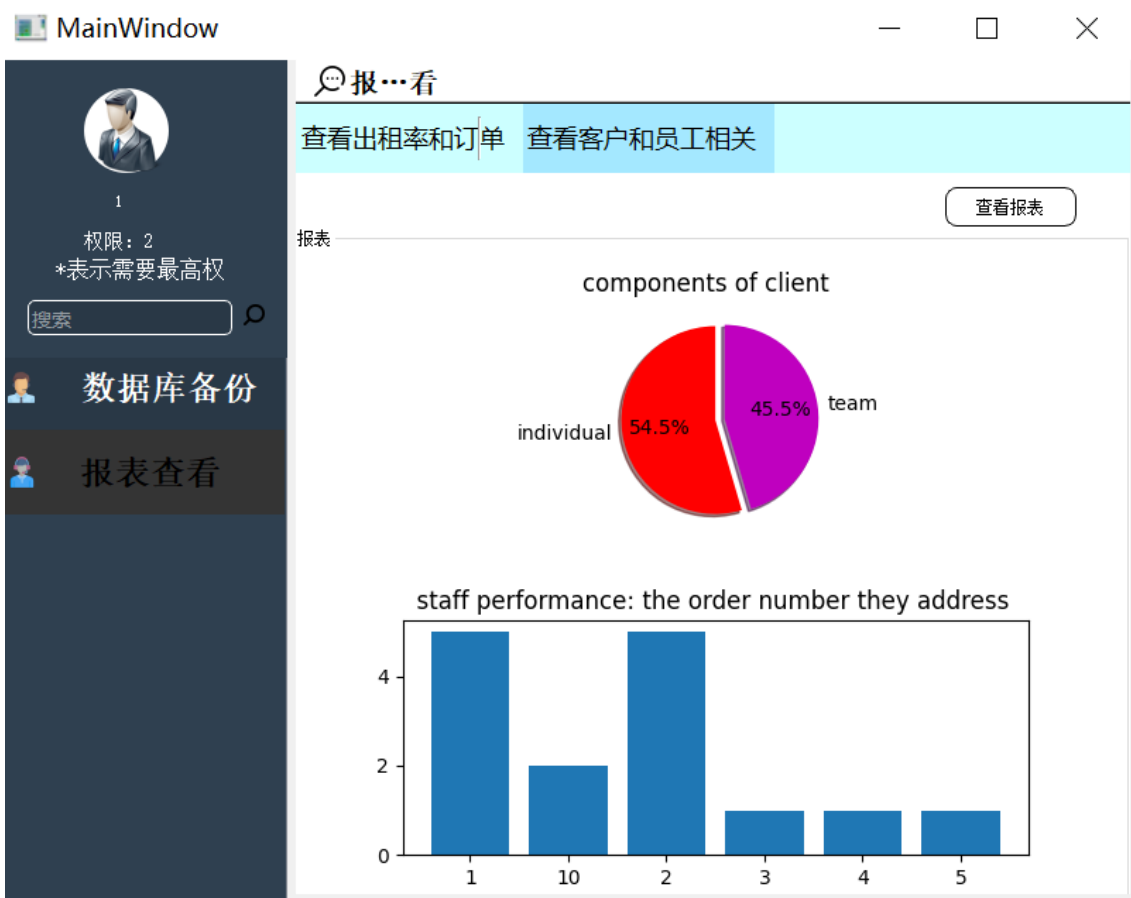
宾馆管理系统--Designed by sjy

version 1.0

- 数据库备份：
 - 导出database (sql)
 - 导出table (sql)
 - 导出table到excel

	A	B	C	D	E	F	G	H	I	J	K	L	M
1	sid	sname	ssex	stime	susername	spasswor	srole	sidcard	sphone				
2	1	张	男	43822	zs123	123456	1	31057219	13977766253				
3	10	1	男	43831	1	1	2	112233	112233				
4	11	李安	男	43831	la123	123456	2	11223344	11223344				
5	2	李四	女	43805	ls123	123456	2	32912319	13823209876				
6	3	张萍	女	43825	zp123	123456	1	33298719	13782765657				
7	4	赵六	女	40179	zl123	123456	1	33298719	13888909890				
8	5	王五	男	43831	wu123	123456	2	33298719	13988767890				
9	6	黄让	男	43831	hr123	123456	2	33298719	13962334343				
0	7	黄小平	女	43803	hxp123	123456	1	33298719	13962334222				
1	8	阿斯頓	男	43801	asd123	123456	1	33298719	13962334333				
2													
3													
4													

- 报表查看

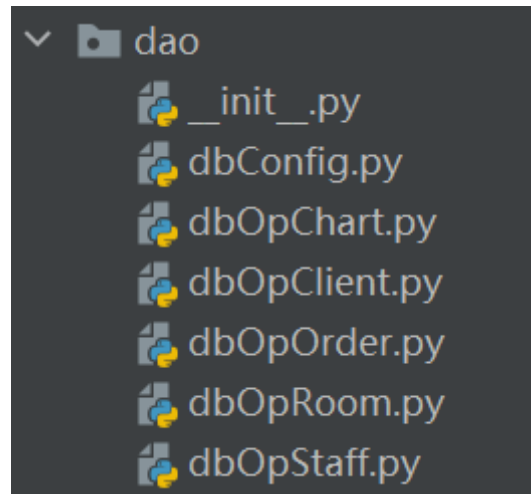


实现框架

dao

是比较底层比较基础的操作，具体到对某个表的增删改查，只做原子操作。负责与数据库进行联络的一些任务都封装在此，dao层的设计首先是设计dao层的接口，然后在Spring的配置文件中定义此接口的实现类，然后就可以再模块中调用此接口来进行数据业务的处理，而不用关心此接口的具体实现类是哪个类，显得结构非常清晰

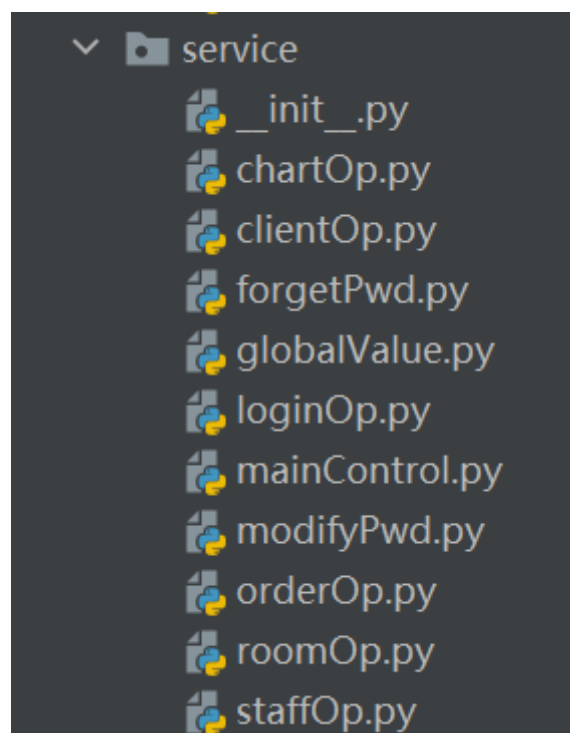
最基础的增删改查mysql语句都是在这里完成的



service

粗略的理解就是对一个或多个DAO进行的再次封装，封装成一个服务，所以这里也就不会是一个原子操作了，需要事物控制。service层主要负责业务模块的应用逻辑应用设计。同样是首先设计接口，再设计其实现类，接着再配置其实现的关联。这样我们就可以在应用中调用service接口来进行业务处理。service层的业务实，具体要调用已经定义的dao层接口，封装service层业务逻辑有利于通用的业务逻辑的独立性和重复利用性。程序显得非常简洁。

提供了基本的操作类，是连接底层数据库与ui界面的一层



ui

ui是用户交互层，负责请求转发，接受页面过来的参数，传给Service处理，接到返回值，再传给页面。在此层要调用service层的接口来控制业务流程，这样不仅使程序结构变得清晰，也大大减少了代码量。

前端显示的界面都是由ui来完成，是先用QtDesigner设计ui界面，在转化为py文件由python.exe执行



后端主要代码（sql语句）

本系统主要是实现客房管理，所以最主要的是实现客房管理功能

办理入住\办理预定

不管是办理入住还是办理预定，都必须事先查找该房间该时刻是否被预定，否则不能预定。所以需要在checkin_client,checkin_team,booking_client,booking_team四个表里面查找相关预定信息。分别存在data1,data2,data3,data4这四个变量中，只有这四个变量都为空，才说明这个房间这个时段可用

```
def
reserveCDB(self, cname, cid, cphone, cage, csex, crid, cstarttime, cendtime, cremark):
    """个人预约"""
    print(cstarttime)
    try:
        self.cursor.execute("select * from checkin_client as A where
(A.rid=%s) and (A.end_time>%s and A.start_time<=%s "
                            "or A.end_time>%s and A.start_time<=%s
A.start_time<=%s and A.end_time>%s or A.start_time>=%s and A.end_time<=%s)"
                            , (crid, cstarttime, cstarttime, cendtime, cendtime,
cstarttime, cendtime, cstarttime, cendtime))
        data1 = self.cursor.fetchall()
        self.cursor.execute("select * from checkin_team as A where
(A.rid=%s) and (A.end_time>%s and A.start_time<=%s "
                            "or A.end_time>%s and A.start_time<=%s or
A.start_time<=%s and A.end_time>%s or A.start_time>=%s and A.end_time<=%s)"
                            , (crid, cstarttime, cstarttime, cendtime, cendtime,
cstarttime, cendtime, cstarttime, cendtime))
        data2 = self.cursor.fetchall()
        self.cursor.execute("select * from booking_client as A where
(A.rid=%s) and (A.end_time>%s and A.start_time<=%s "
```

```

        "or A.end_time>%s and A.start_time<%s or
A.start_time<=%s and A.end_time>%s or A.start_time>=%s and A.end_time<=%s)"
        , (crid, cstarttime, cstarttime, cendtime, cendtime,
cstarttime, cendtime, cstarttime, cendtime))
        data3 = self.cursor.fetchall()
        self.cursor.execute("select * from booking_team as A where
(A.rid=%s) and (A.end_time>%s and A.start_time<%s "
        "or A.end_time>%s and A.start_time<%s or
A.start_time<=%s and A.end_time>%s or A.start_time>=%s and A.end_time<=%s)"
        , (crid, cstarttime, cstarttime, cendtime, cendtime,
cstarttime, cendtime, cstarttime, cendtime))
        data4 = self.cursor.fetchall()
    except Exception as e:
        print(e)
        return False
    if data1 != () or data2 != () or data3 != () or data4 != ():
        QMessageBox().information(None, "提示", "该时间段对应房间被占用（入住/预
约）！", QMessageBox.Yes)
        return False
    self.cursor.execute("select * from client where cid=%s", (cid))
    data = self.cursor.fetchall()
    if data == ():
        self.cursor.execute(
            "insert into
client(cname,cid,cphone,cage,csex,register_sid,accomodation_times)
values(%s,%s,%s,%s,%s,%s,%s)",
            (cname, cid, cphone, cage, csex, self.staff.sid, 0))
    try:
        self.cursor.execute("insert into
booking_client(cid,rid,start_time,end_time,remark) values(%s,%s,%s,%s,%s)",
            (cid,crid,cstarttime,cendtime,cremark))
        self.db.commit()
        return True
    except Exception as e:
        print(e)
        QMessageBox().information(None, "提示", "相关预约信息已存在！",
QMessageBox.Yes)
        return False

```

预定登记

首先要根据type找到在对应的表中进行查找和插入操作，以个人为例

需要先select booking_client，若是找到响应记录，则可以进行下面的操作，否则提示没有相关预定信息。找到相关信息后，需要删除booking_client里面的相关信息，并且插入checkin_client1的相关信息

```

def singleCheckinDB(self, cname, cid, cphone, cage, csex, crid, cendtime, remark):
    """个人入住"""
    # 查询预定表和入住表，判断该房间是否能租出去
    starttime = datetime.date.today()
    self.cursor.execute("select * from checkin_client as A where (A.rid=%s)
and (A.end_time>%s and A.start_time<%s "
        "or A.end_time>%s and A.start_time<%s or
A.start_time<=%s and A.end_time>=%s or A.start_time>=%s and A.end_time<=%s)"

```

```

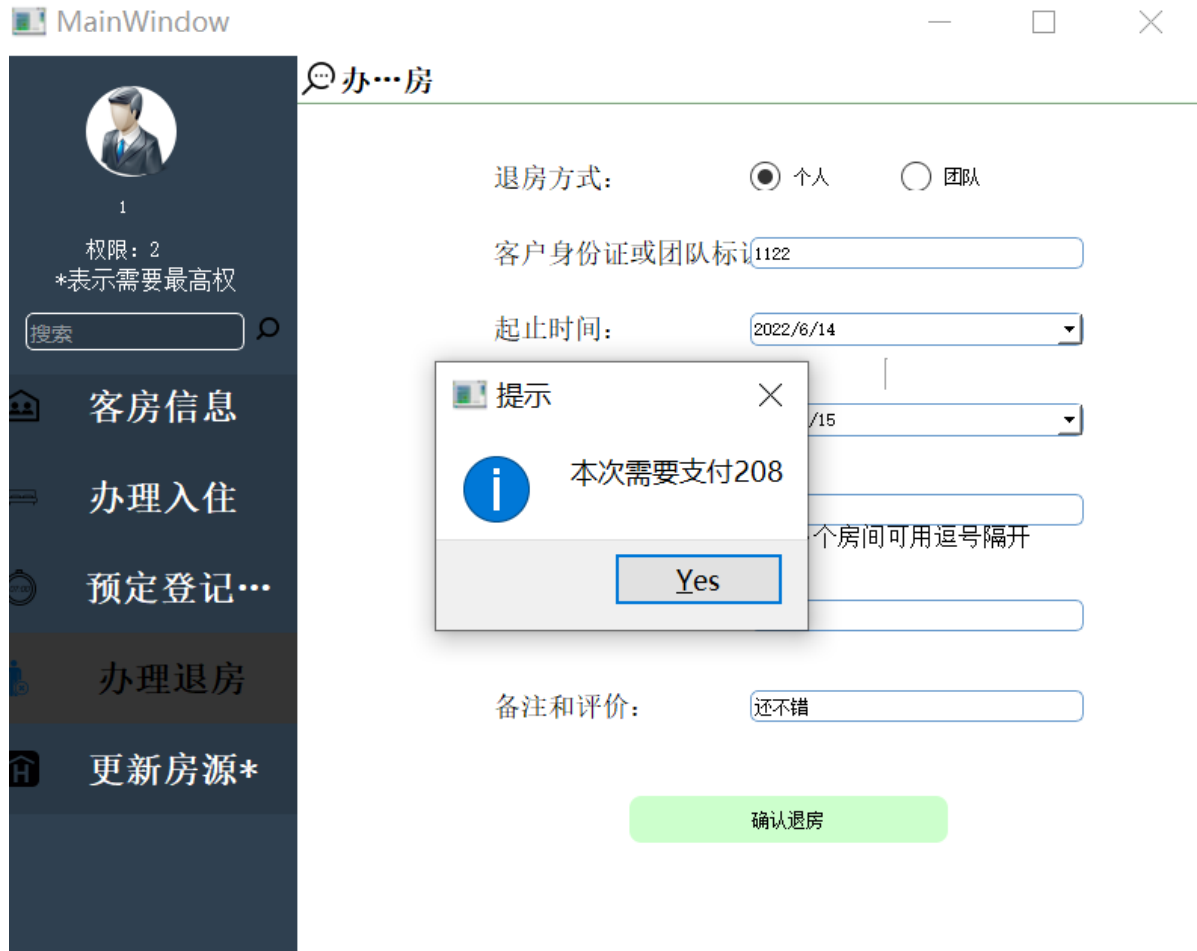
(crid,starttime,starttime,cendtime,cendtime,starttime,cendtime,starttime,cendtim
e))

    data1 = self.cursor.fetchall()
    self.cursor.execute("select * from checkin_team as A where (A.rid=%s)
and (A.end_time>%s and A.start_time<%s "
                        "or A.end_time>%s and A.start_time<%s or
A.start_time<=%s and A.end_time>=%s or A.start_time>=%s and A.end_time<=%s)"
                        , (crid, starttime, starttime, cendtime, cendtime,
starttime, cendtime,starttime,cendtime))
    data2 = self.cursor.fetchall()
    self.cursor.execute("select * from booking_client as A where (A.rid=%s)
and (A.end_time>%s and A.start_time<%s "
                        "or A.end_time>%s and A.start_time<%s or
A.start_time<=%s and A.end_time>=%s or A.start_time>=%s and A.end_time<=%s)"
                        , (crid, starttime, starttime, cendtime, cendtime,
starttime, cendtime,starttime,cendtime))
    data3 = self.cursor.fetchall()
    self.cursor.execute("select * from booking_team as A where (A.rid=%s)
and (A.end_time>%s and A.start_time<%s "
                        "or A.end_time>%s and A.start_time<%s or
A.start_time<=%s and A.end_time>=%s or A.start_time>=%s and A.end_time<=%s)"
                        , (crid, starttime, starttime, cendtime, cendtime,
starttime, cendtime,starttime,cendtime))
    data4 = self.cursor.fetchall()
    if data1 != () or data2 != () or data3 != () or data4 != ():
        QMessageBox().information(None, "提示", "该时间段对应房间被占用（入住/预
约）！", QMessageBox.Yes)
        return False
    self.cursor.execute("select * from client where cid=%s",(cid))
    data = self.cursor.fetchall()
    if data == ():
        self.cursor.execute("insert into
client(cname,cid,cphone,cage,csex,register_sid,accomodation_times) "
                            "values(%s,%s,%s,%s,%s,%s,%s)",
(cname,cid,cphone,cage,csex,self.staff.sid,0))
        self.cursor.execute("select * from room where rid=%s",(crid))
        data = self.cursor.fetchall()
        if data == ():
            QMessageBox().information(None, "提示", "没有对应房间号！",
QMessageBox.Yes)
            return False
        perPrice = data[0]['rprice']
        totalPrice = int(perPrice) * int((cendtime-starttime).days)
        try:
            self.cursor.execute("insert into checkin_client
values(%s,%s,%s,%s,%s,%s,%s,%s)",
(crid,cid,starttime,cendtime,totalPrice,self.staff.sid,remark))
            self.db.commit()
            return True
        except Exception as e:
            print(e)
            QMessageBox().information(None, "提示", "相关客户已入住，请勿重复插入",
QMessageBox.Yes)
            return False

```

退房

和预定登记一样，需要先在checkin表里面找相关数据，找到则删除信息，并且在hotelorder里面插入信息。并且根据现实生活中的例子，退房是涉及支付，系统会提醒相关的支付信息，并且一并存入数据库系统



```
def checkoutDB(self, flag, id, rid, payType, remark):
    """两种方式退房"""
    try:
        if flag == '个人':
            self.cursor.execute("select * from checkin_client where rid=%s
and cid=%s", (rid, id))
            data = self.cursor.fetchall()
            if data == ():
                QMessageBox().information(None, "提示", "没有相关入住信息!",
QMessageBox.Yes)
                return False
            else:
                rid_out = data[0]['rid']
                cid_out = data[0]['cid']
                stime_out = data[0]['start_time']
                etime_out = data[0]['end_time']
                money = data[0]['total_price']
                self.cursor.execute("insert into
hotelorder(id,ordertype,start_time,end_time,rid,pay_type,money,remark,register_s
id) values(%s,%s,%s,%s,%s,%s,%s,%s,%s,%s)",
(cid_out, flag, stime_out, etime_out, rid_out, payType, money, remark, self.staff.sid))
                self.cursor.execute("delete from checkin_client where rid=%s
and cid=%s", (rid_out, cid_out))
```

```

        self.db.commit()
        QMessageBox().information(None, "提示", "本次需要支付%s"
%money, QMessageBox.Yes)
        elif flag == '团队':
            sum = 0
            for r in re.split(',|,| ',rid):
                self.cursor.execute(
                    "select * from checkin_team where rid=%s and tid=%s",
(r, id))

                data = self.cursor.fetchall()
                if data == ():
                    QMessageBox().information(None, "提示", "没有相关入住信
息!", QMessageBox.Yes)
                    return False
                else:
                    rid_out = data[0]['rid']
                    tid_out = data[0]['tid']
                    stime_out = data[0]['start_time']
                    etime_out = data[0]['end_time']
                    money = data[0]['total_price']
                    self.cursor.execute(
                        "insert into
hotelorder(id,ordertype,start_time,end_time,rid,pay_type,money,remark,register_s
id) values(%s,%s,%s,%s,%s,%s,%s,%s,%s,%s)"
                        , (tid_out, flag, stime_out, etime_out, rid_out,
payType, money, remark,self.staff.sid))
                    self.cursor.execute("delete from checkin_team where
rid=%s and tid=%s", (rid_out, tid_out))
                    self.db.commit()
                    sum = sum + int(money)
                    QMessageBox().information(None, "提示", "本次需要支付%s" %str(sum),
QMessageBox.Yes)
                    return True
            except Exception as e:
                print(e)
                return False

```

客房管理

这里可以更新房源信息，包括修改和增加

展望

- 删除员工信息（因为外键的约束）。若没有外键约束，则删除后直接查询时会没有删除，但是数据库里面是已经删除了的
- 个人预定（数据库语法错误）
- 删除--->需要点击cid字段进行删除，存在外键约束。
- 修改客户信息---->**尚未实现这个功能**