2018

Hotel Management System

REPORT

何钰佳 DOLLY 1530005008

李万水 WESLEY 1530005017

李玥 LILIAN 1530005018

Listing

Hotel Management System

1. Propose

As a rapidly growing economic industry, the hotel industry has a huge potential for development and hotel competition is pretty fierce. Traditional manual recording hotel information is not convenient and fast enough for people to query what they want, which cannot meet the actual demands, especially for five-star hotels. Based on the hotels' actual demands (from some professional hotels' references), we designed a hotel management system, which is not only simple to operate, but also can reduce the consumption of human resources and save money. It can help the hotel to improve the ability of service level, establish a good image and have an advantage on the competition.

2. Functionalities

This database is designed both for hotel reception and manager:

- **For hotel reception**

2.1.1 rooms query: search the current occupancy, customers who have already checked, and oversee the appropriate rooms. This application humanely provides the description of each certain room so that satisfying the special requirement of customer.

2.1.2 check in, check out, and update the state of room: this hotel database can achieve customer check in and al so when customer check out, at the same time database update the current state of room, which make it easy to inform the staff of corresponding room to clean.

2.1.3 special VIP rules: for customer's convenient, special vip rules can be achieve by this database. By recording the accumulated consumption of each arrived customer, and automatically upgrade the customer with high consumption to VIP which can regularly enjoy discounts.

- **For management:**

2.2 management of employees

2.2.1 query information about staffs: checking the personal information, income and performance of all staffs

2.2.2 update new information of the staff: Register the fresh staffs and remove resign staffs

2.2.3 timely know rooms waiters are responsible for: in case some responsibility investigation from customers' complaints.

2.3 management of customers information

2.3.1 insert information about the new customers (registering)

2.3.2 query information about the customers

2.3.3 change the information about customers (update)

2.4 **analyze the benefit performance**:

by searching the customer flow volume and checking the occupancy at present.

3. Assumption you made:

3.1 The accumulated consumption is larger than 8000, customers could become VIP.

3.2 Only users input correct ID and password then he could enter the system and use the functions.

3.3 When customer check in or check out the state of room should be updated so that corresponding cleaner could clean timely

3.4 When customer select room, may want to know the room type, scene out of window, whether or not the room price including the breakfast.

Hotel should have some good discounts for old customer which can benefit hotel in a long run.

3.5 Hotel should keep all the information of arrived customer, and automatically updated the total spent when customer repeatedly arrived.

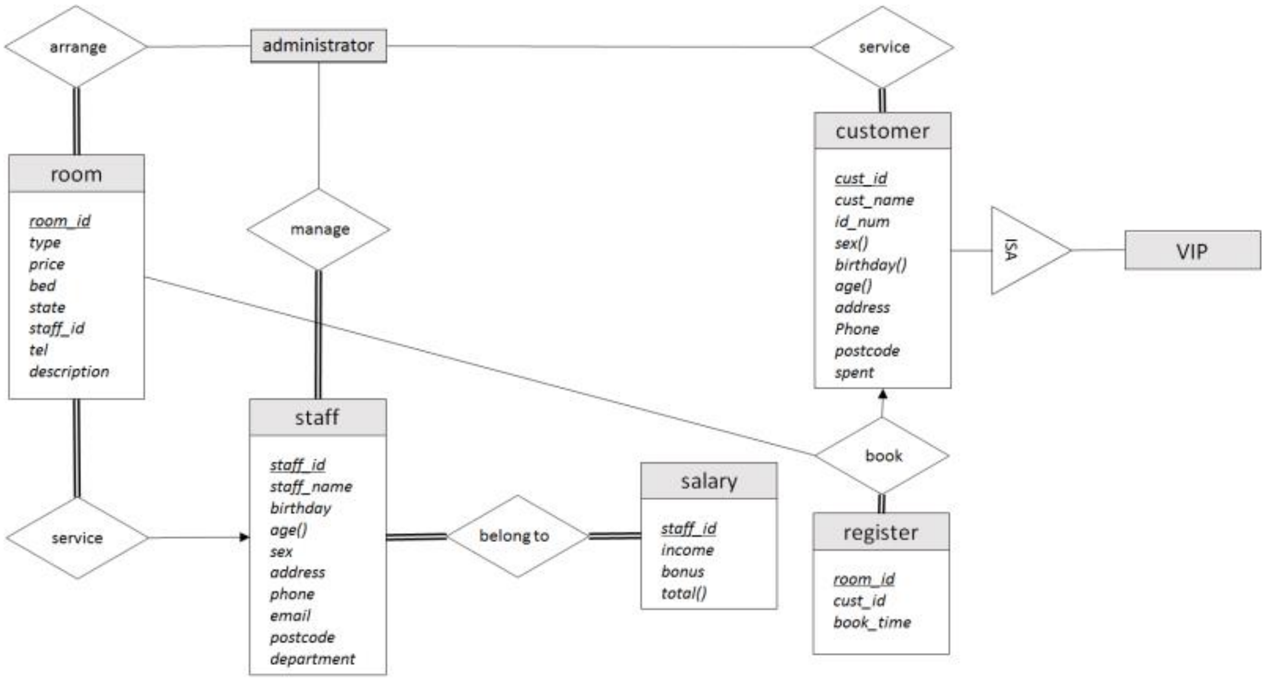
3.6 Each room should have a staff are responsible for so that reduce the management pressure of manager.

3.7 As this database should also be used by manager, a record to all staffs' information, salary and deserved bonus is necessary.

3.8 When being used by reception and manager, some personal information from staff and customer should not be viewed by reception, and some information which is unnecessary to manage also should not viewed by manager.

3.9 Each customer could book multiple rooms, but when come to check in, one customer can only register one room. Also customer's booking time should also by recorded for checking the booking order in case two customers come at the same time.

4. ER diagram



Customer (<u>cust_id</u>,cust_name,id_num,address,birthday,phone,postcode,spent)

Register(<u>room_id</u>,cust_id,book_time)

Room(<u>room_id,</u>staff_id,tel,type,price,bed,state,description)

Salary(<u>staff_id</u>,income,bonus)

Staff (<u>staff_id</u>,staff_name,birthday,sex,address,phone,email,postcode,department)

Vip(<u>cust_id</u>)

Service(<u>room_id,staff_id,tel</u>,type,price,bed,state,description)

Belong to

(<u>st.staff_id,sa.staff_id</u> ,st.staff_name,st.birthday,st.sex,st.address,st.phone,st.email,st.postcode,st.department ,sa.income,sa.bonus)

Book(<u>ro.room_id</u> <u>re.room_id,</u>re.cust_id,book_time,c.cust_id)

4.1 Explaination:

In the ER diagram, customer entity has book relationship with register and one customer can register one or more rooms. And VIP is a type of customer, which including the customer those who meet certain condition.
The room entity has service relationship with staff, and each room should be served by a certain staff, while each staff can serve zero or more rooms.
The salary entity has belongs to staff information entity, each staff has one or more salary sources.
The administrator have arrange relation with all rooms and serve all customers also manage all staffs.

5. Functional Dependency:

5.1 customer:

Cust_id—cust_name,address,postcode,spent;

id_num—cust_id;

phone—id_num

5.2 room

Room_id—type,price,bed,state,descriptionc

Staff_id—room_id

tel—Staff_id

5.3 register

room_id—book_time ,cust_id，

5.4 staff

Staff_id—name,birthday,sex,address,postcode,department

phone—staff_id

email—phone

5.5 salary

Staff_id—income,bonus

5.6 vip

Cust_id

6. SQL and explanations

6.1 Manager used:

6.1.1 Function1: management of employees

/*query all staffs, to check whose income larger than 4500 yuan*/

SELECT * FROM staff,salary WHERE staff.`staff_id`=salary.`staff_id` AND income>4500;

/*ranking the income from the low to the high*/

SELECT * FROM salary,staff WHERE staff.`staff_id`=salary.`staff_id` ORDER BY salary.income ASC;

```
/*query the name and income for the staffs who income over 2200 yuan and from
department 1*/

SELECT staff_name,income FROM staff,salary WHERE
staff.staff_id=salary.staff_id AND income>2200 AND staff.department= 1;


/*To calculate the average income for department 1's staff*/

SELECT AVG(income) avg_income FROM salary,staff WHERE
staff.staff_id=salary.staff_id AND staff.department= 1;


/*query the lowest and the highest and lowest income from department 1*/

SELECT MAX(income) max_income,MIN(income) min_income FROM salary,staff
WHERE staff.staff_id=salary.staff_id AND staff.department= 1;


/*count the number of staff for each department*/

SELECT department, COUNT(*) number FROM staff GROUP BY department;


/*count the number of staff whose incomes larger than 2000 yuan*/

SELECT department, COUNT(*) number FROM staff,salary WHERE
staff.staff_id=salary.staff_id AND income>2000 GROUP BY department;
```

/*find the birthdays,ages and sexs for customers by ID number)(the method is also could use to staff)*/

DATE_FORMAT(CAST(SUBSTRING(id_num,7,8) AS DATE), '%m-%d') AS Birthday

SELECT id_num, CAST(SUBSTRING(id_num,7,8) AS DATE) AS 'Birthday',

DATE_FORMAT(NOW(), '%Y') - SUBSTRING(id_num,7,4) AS age,

IF(LEFT(SUBSTRING(id_num,17),1)%2=1,'M','F') AS sex FROM customer WHERE cust_id=100001;

/* calculate the total income of staff)*/

SELECT staff_id, income, bonus, SUM(income+bonus) AS TOTAL FROM salary GROUP BY staff_id;

/* calculate the total income of staff)*/

SELECT cust_id,spent FROM vip_info WHERE spent>8500;

/* Ranking the bonus from the high to the low,the bonus is based on the extra working hours for each staff*/

SELECT staff_id, staff_name, bonus FROM salary NATURAL JOIN staff ORDER BY bonus DESC;

/*count the number of today booking*/

SELECT COUNT(cust_id) FROM register GROUP BY book_time HAVING book_time=CURDATE();

/*sum the price of 10% discount（-10%） */

SELECT SUM(price*0.01) expense FROM room NATURAL JOIN register NATURAL JOIN VIP;

/*check the occupied proportion*/

SELECT (COUNT(room_id)/temp.tr) ocupy_proportion FROM room,(SELECT COUNT(room_id) tr FROM room) AS temp WHERE room.state='ocupy';

/*nest*/

/*find the ages and names of staff who from department 1 and are older than all of staff from department 2 */

SELECT staff_name FROM staff WHERE department=1 AND birthday <=ALL (SELECT birthday FROM staff WHERE department=2) ;

/*find the staff name whose income is higher than all staff from department 1*/

SELECT staff.staff_name FROM staff WHERE staff_id IN (SELECT salary.staff_id FROM salary WHERE income>ALL (SELECT income FROM salary WHERE salary.staff_id IN (SELECT staff.staff_id FROM staff WHERE department=1)));

6.1.2 Function2: manage customer

/* find the birthdays,ages and sexs for customers by ID number)(the method is also could use to staff)*/   DATE_FORMAT(CAST(SUBSTRING(id_num,7,8) AS DATE), '%m-%d') AS Birthday

SELECT id_num, CAST(SUBSTRING(id_num,7,8) AS DATE) AS 'Birthday',

DATE_FORMAT(NOW(), '%Y') - SUBSTRING(id_num,7,4) AS age,

IF(LEFT(SUBSTRING(id_num,17),1)%2=1,'M','F') AS sex FROM customer WHERE cust_id=100001;

/* query the spent larger than 8500 from VIP customers*/

SELECT cust_id,spent FROM vip_info WHERE spent>8500;

/* count   the number of today booking*/

SELECT count(cust_id) from register GROUP BY book_time HAVING book_time=”2018-4-1”

6.1.3 Function3: analyze the benefit performance

/*nest*/

/*calculate the total income in today（only can calculate in occupied rooms）*/

```
SELECT sum(room.price) FROM room,register where
room.room_id=register.room_id and room.book_time=”2018-4-1”;


/*sum the price of 10% discount（-10%） */

SELECT sum(room.price) FROM room,register,VIP WHERE
register.cust_id=VIP.cust_id and register.room_id=room.room_id and
room.price*0.01;
```

6.2 For reception:

6.2.1 Function1: room,customer,register VIP information query

```
/*create view of query*/


/*view of query customers information*/

CREATE VIEW cust_info AS SELECT * FROM customer;

# select * from cust_info;


/* view of query rooms information */

CREATE VIEW room_info AS SELECT * FROM room;

# select * from room_info;


/* view of query check in information */
```

```
CREATE VIEW register_info AS SELECT * FROM register;

CREATE VIEW regis_info AS SELECT
register.`room_id`,cust_name,register.`cust_id`,room.`type`,price FROM
room,customer,register WHERE room.`room_id`=register.`room_id` AND
register.`cust_id`=customer.`cust_id`;

# select * from register_info;

# select * from regis_info;


/* view of query VIP customers information */

CREATE VIEW vip_info AS SELECT * FROM vip NATURAL JOIN customer;

# select * from vip_info;
```

6.2.2 Function2: room state, VIP state update, check in and check out

```
/* update the state and spent of rooms in time */

delete from register where room_id=44599;

insert into register values (44599,'100018','2018-05-23')

select * from customer where cust_id=100018;

select * from register where room_id=44599;

select * from room where room_id=44599;

select * from vip where cust_id=100018;
```

```
/*check in and check out and room state update*/

DELETE FROM register WHERE room_id=88888

SELECT * FROM room WHERE room_id=88888;

INSERT INTO register VALUES (88888,100001,'2018-05-22');

select * from customer where cust_id=100001;


/*check in& check out&room state update*/

insert into customer values('150001', 'ht Ghuui', '4321241972453302743',
'FuTianDistrict, SheRoad-NO.472330', '1989-02-27', '13451902926', '300');

delete from register where room_id=10001;

SELECT * FROM customer WHERE cust_id=150001;

INSERT INTO register VALUES (10001,150001,'2018-05-23');
```

/* build up index*/

```
/* index of room table */

CREATE UNIQUE INDEX ur ON room(room_id,price);


/* index of customer table */

CREATE UNIQUE INDEX uc ON customer(cust_id,cust_name);
```

```
/* index of register table*/

CREATE UNIQUE INDEX ure    ON register(cust_id,room_id);

/* index of VIP table */

CREATE UNIQUE INDEX uv     ON vip(cust_id);

/*build up stored procedure*/



/*stored procedure of rooms*/

    /*stored procedure of rooms of insert rooms information*/

DELIMITER $$

CREATE PROCEDURE    insert_room_info (IN v_room_id VARCHAR(20),IN
v_type VARCHAR(20),IN v_price DOUBLE,IN v_bed INT,IN v_state
VARCHAR(20),IN v_staff_id VARCHAR(20),IN v_tel VARCHAR(20),IN
v_description VARCHAR(255))

    BEGIN

        INSERT INTO room VALUES
(v_room_id,v_type,v_price,v_bed,v_state,v_staff_id,v_tel,v_description);

    END$$

DELIMITER ;

/*call insert_room_info(666666,'bigbig',23333,3,'available',535000,6666666,'full
services');*/



    /* stored procedure of rooms of modification rooms information */
```

```
DELIMITER $$

CREATE PROCEDURE    alter_room_info (IN v_room_id VARCHAR(20),IN
v_type VARCHAR(20),IN v_price DOUBLE,IN v_bed INT,IN v_state
VARCHAR(20),IN v_staff_id VARCHAR(20),IN v_tel VARCHAR(20),IN
v_description VARCHAR(255))

    BEGIN

        UPDATE room SET
TYPE=v_type,price=v_price,bed=v_bed,state=v_state,staff_id=v_staff_id,tel=v_tel,d
escription=v_description WHERE room_id=v_room_id;

    END$$

DELIMITER ;

/*call alter_room_info(666666,'smallsmall',23333,3,'available',535000,6666666,'full
services');*/


    /* stored procedure of rooms of delete rooms information */

DELIMITER $$

CREATE PROCEDURE    delete_room_info (IN v_room_id VARCHAR(20))

    BEGIN

        DELETE FROM room WHERE room_id=v_room_id;

    END$$

DELIMITER ;

/*call delete_room_info(666666);*/
```

```
/* stored procedure of customers*/

        /*insert information of customers */

DELIMITER $$

CREATE PROCEDURE insert_cust_info (IN v_cust_id VARCHAR(20), IN
v_cust_name VARCHAR(30),IN v_id_num VARCHAR(20),IN v_address
VARCHAR(255),IN v_phone VARCHAR(20),IN v_postcode VARCHAR(20),IN
v_spent DOUBLE)

        BEGIN

                INSERT INTO customer VALUES
(v_cust_id,v_cust_name,v_id_num,v_address,v_phone,v_postcode,v_spent);

        END$$

DELIMITER ;

/*call insert_cust_info (666666,'xxx
xxxx',330821199610200048,'UnitedInternationalCollege, lalala-
No.3933',13938573857,510000,0);*/


        /* modification information of customers */

DELIMITER $$

CREATE PROCEDURE alter_cust_info (IN v_cust_id VARCHAR(20), IN
v_cust_name VARCHAR(30),IN v_id_num VARCHAR(20),IN v_address
VARCHAR(255),IN v_phone VARCHAR(20),IN v_postcode VARCHAR(20),IN
v_spent DOUBLE)
```

```
    BEGIN
        UPDATE customer SET
cust_name=v_cust_name,id_num=v_id_num,address=v_address,phone=v_phone,post
code=v_postcode,spent=v_spent WHERE cust_id=v_cust_id;
    END$$
DELIMITER ;
/*call alter_cust_info (666666,'ooo
oooo',330821199610200048,'UnitedInternationalCollege, lalala-
No.3933',13938573857,510510,0);*/


    /*delete information of customers */
DELIMITER $$
CREATE PROCEDURE delete_cust_info (IN v_cust_id VARCHAR(20))
    BEGIN
        DELETE FROM customer WHERE cust_id=v_cust_id;
    END$$
DELIMITER ;
/*call delete_cust_info (666666);*/


/*create information of check in */
    /* insert information of check in */
```

```
DELIMITER $$

CREATE PROCEDURE insert_regis_info (IN v_room_id VARCHAR(20), IN
v_cust_id VARCHAR(20),IN v_book_time DATE)

    BEGIN

        INSERT INTO register VALUES(v_room_id,v_cust_id,v_book_time);

    END$$

DELIMITER ;

/*call insert_regis_info();*/



    /*modification information of check in */

DELIMITER $$

CREATE PROCEDURE alter_regis_info (IN v_room_id VARCHAR(20), IN
v_cust_id VARCHAR(20),IN v_book_time DATE)

    BEGIN

        UPDATE register SET
room_id=v_kfb,cust_id=v_cust_id,book_time=v_book_time;

    END$$

DELIMITER ;

/*call alter_regis_info ();*/



    /* delete information of check in */
```

```
DELIMITER $$

CREATE PROCEDURE delete_regis_info (IN v_cust_id VARCHAR(20))

    BEGIN

        DELETE FROM register WHERE cust_id=v_cust_id;

    END$$

DELIMITER ;

/*call delete_regis_info ()*/



/*stored procedure of query */

    /*query rooms numbers */

DELIMITER $$

CREATE PROCEDURE room_id_query (IN v_room_id VARCHAR(20),OUT
v_type VARCHAR(20),OUT v_price DOUBLE,OUT v_bed INT,OUT v_state
VARCHAR(20),OUT v_staff_id VARCHAR(20),OUT tel VARCHAR(20),OUT
description VARCHAR(255))

    BEGIN

        SELECT
v_type=TYPE,v_price=price,v_bed=bed,v_state=state,v_staff_id=staff_id,v_tel=tel,v
_description=description FROM room WHERE room_id=v_room_id;

    END$$

DELIMITER ;

/*call room_id_query ()*/
```

```
    /* query customers numbers */
DELIMITER $$
CREATE PROCEDURE cust_id_query (IN v_cust_id VARCHAR(20),OUT
v_cust_name VARCHAR(30),OUT v_id_num VARCHAR(20),OUT v_address
VARCHAR(255),v_birthday DATE,v_phone VARCHAR(20),v_postcode
VARCHAR(20),v_spent DOUBLE)
    BEGIN
        SELECT
v_cust_name=cust_name,v_id_num=id_num,v_address=address,v_birthday=birthday,
v_phone=phone,v_postcode=postcode,v_spent=spent FROM customer WHERE
cust_id=v_cust_id;
    END$$
DELIMITER ;
/*call cust_id_query*/


    /* query customer id, who has checked in */
DELIMITER $$
CREATE PROCEDURE regis_cust_id_query (OUT v_room_id VARCHAR(20),IN
v_cust_id VARCHAR(20),OUT v_book_time DATE)
    BEGIN
        SELECT v_cust_id=cust_id,v_book_time=book_time FROM register
WHERE room_id=v_room_id;
```

```
    END$$
DELIMITER ;
/*call regis_cust_id_query ()*/


/* bulid up trigger */


    /* update the state and spent of rooms in time */
DELIMITER $$
CREATE TRIGGER check_in AFTER INSERT ON register FOR EACH ROW
    BEGIN
        DECLARE s INT;
        DECLARE p INT;
        UPDATE room SET state='occupy' WHERE room_id=new.room_id;
        SET s=(SELECT spent FROM customer WHERE cust_id=new.cust_id);
        SET p=(SELECT price FROM room WHERE room.room_id=new.room_id);
        UPDATE customer SET spent=s+p WHERE cust_id=new.cust_id;
    END$$
DELIMITER ;


    /* to judge that when to delete the information of check in */
```

```
DELIMITER $$

CREATE TRIGGER check_out AFTER DELETE ON register FOR EACH ROW

    BEGIN

        UPDATE room SET state='available' WHERE room_id=old.room_id;

    END$$

DELIMITER ;



    /*update VIP*/

DELIMITER $$

CREATE TRIGGER vip_tri AFTER UPDATE ON customer FOR EACH ROW

    BEGIN

        INSERT INTO vip SELECT cust_id FROM customer WHERE spent>=8000
AND cust_id NOT IN (SELECT cust_id FROM vip);

    END$$

DELIMITER ;
```

7. JAVA for function achieved (for hotel reception used)

7.1 Check in and check out;

7.2 Add new customer's information

7.3 Query specific type of available rooms

7.4 Limit the number of display rows

JAVA Code:

1. （check out）

call delete_regis_info(?)

2. （check in）

call insert_regis_info(?,?,?)

3. （add new customer's information）

call insert_cust_info(?,?,?,?,?,?,?)

4. （query specific type of available room）

select * from room where type like ? and state='available' order by room_id

5. （limit the number of display rows）

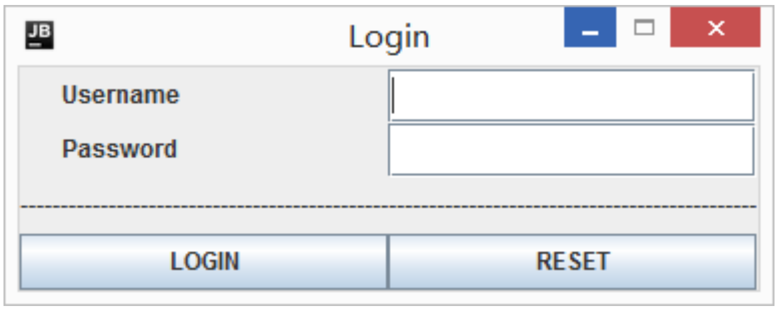
select * from room limit ?,?

8. Difficulties

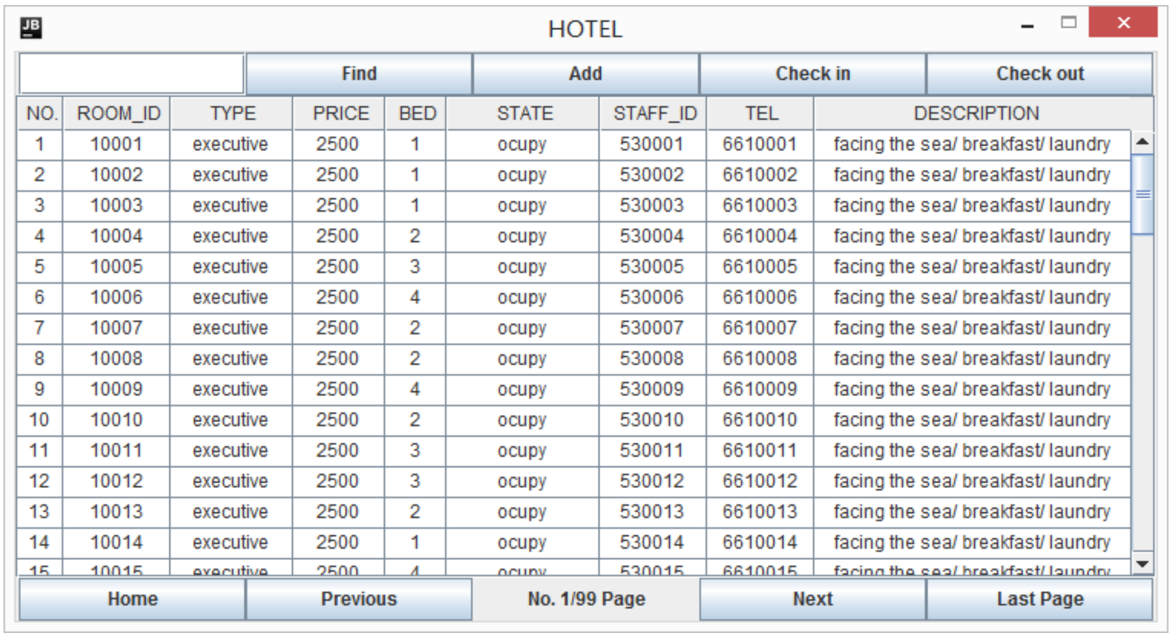
During the process, firstly, we meet problem when select a suitable topic, as initially we have unclear recognize to database we will design and relevant software that might be useful. At the same time, it’s not easy to find a topic which satisfy the project requirement such as 7 entities and all of them including 5000 plus rows. Secondly, when drawing ER-diagram we want it as much complex as possible, however, later we understand the complex one is also hard to achieve. Thirdly, we thought is difficult to collect enough data and we choose to simulate data which is also a hard work. After importing data to improve the speed we create index and create trigger to make needed state updated, which may be a little bit difficulty. Fourthly, we have searched a lot of from websites and books to study how to do to improve the convenience for using and finally we use JAVA.

Appendix

The application of the management hotel for hotel reception:

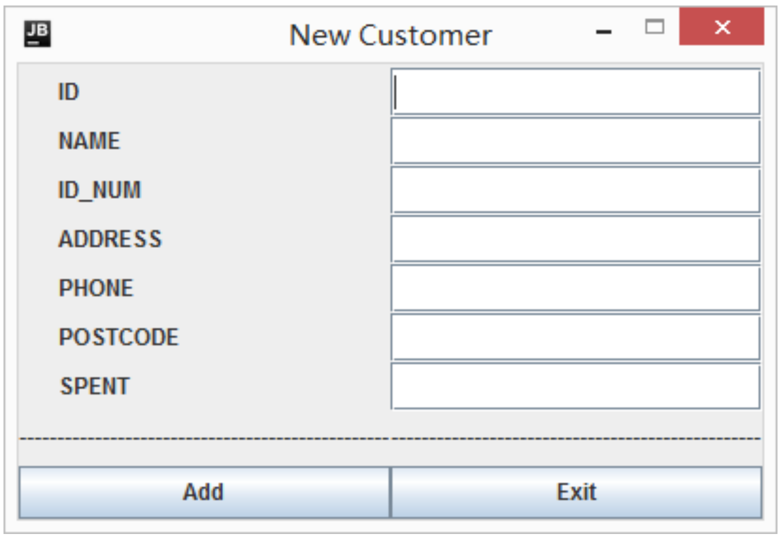


Only users input correct ID and password then he could enter the system and use the functions.



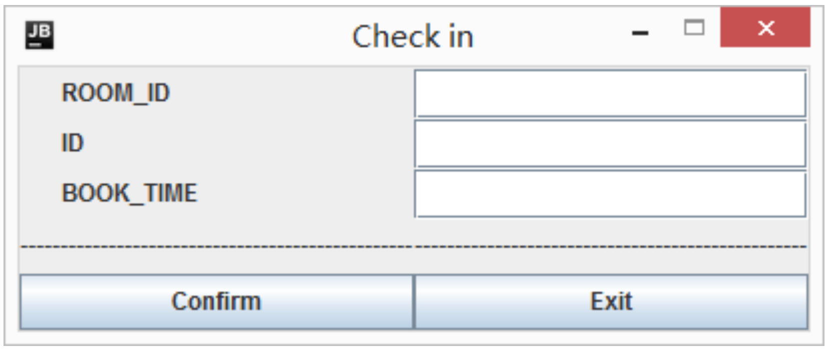
NO.	ROOM_ID	TYPE	PRICE	BED	STATE	STAFF_ID	TEL	DESCRIPTION
1	10001	executive	2500	1	ocupy	530001	6610001	facing the sea/ breakfast/ laundry
2	10002	executive	2500	1	ocupy	530002	6610002	facing the sea/ breakfast/ laundry
3	10003	executive	2500	1	ocupy	530003	6610003	facing the sea/ breakfast/ laundry
4	10004	executive	2500	2	ocupy	530004	6610004	facing the sea/ breakfast/ laundry
5	10005	executive	2500	3	ocupy	530005	6610005	facing the sea/ breakfast/ laundry
6	10006	executive	2500	4	ocupy	530006	6610006	facing the sea/ breakfast/ laundry
7	10007	executive	2500	2	ocupy	530007	6610007	facing the sea/ breakfast/ laundry
8	10008	executive	2500	2	ocupy	530008	6610008	facing the sea/ breakfast/ laundry
9	10009	executive	2500	4	ocupy	530009	6610009	facing the sea/ breakfast/ laundry
10	10010	executive	2500	2	ocupy	530010	6610010	facing the sea/ breakfast/ laundry
11	10011	executive	2500	3	ocupy	530011	6610011	facing the sea/ breakfast/ laundry
12	10012	executive	2500	3	ocupy	530012	6610012	facing the sea/ breakfast/ laundry
13	10013	executive	2500	2	ocupy	530013	6610013	facing the sea/ breakfast/ laundry
14	10014	executive	2500	1	ocupy	530014	6610014	facing the sea/ breakfast/ laundry
15	10015	executive	2500	4	ocupy	530015	6610015	facing the sea/ breakfast/ laundry

Part of data from Room table.



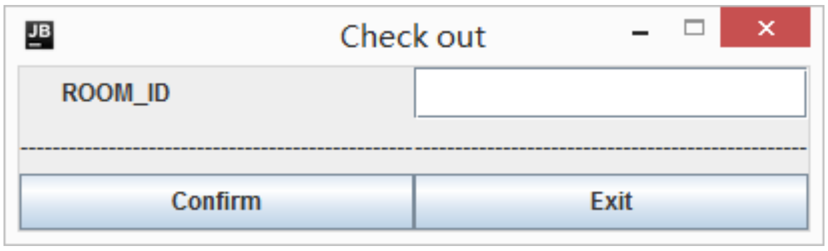
ID	
NAME	
ID_NUM	
ADDRESS	
PHONE	
POSTCODE	
SPENT	

When a customer first time lives in the hotel, staff need to insert those information.



ROOM_ID	
ID	
BOOK_TIME	

If a staff click the check-in key, the check-in interface will show in



ROOM_ID	

If a staff click the check-out key, the check-out interface will show in.

The staff only need to input the room id, then the system could change the state of room immediately.