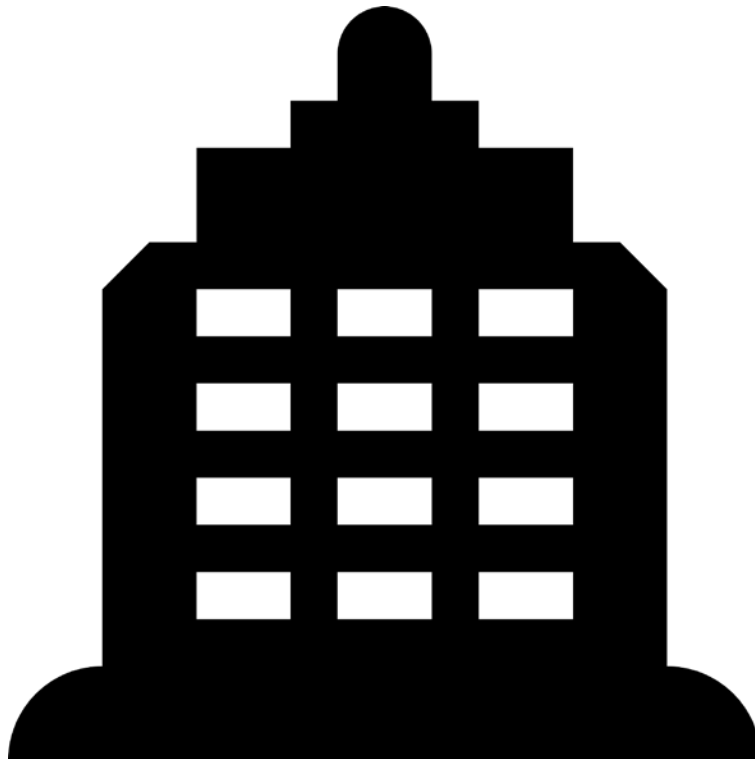


HOTEL RESERVATION SYSTEM

FALL 2017 CS 157A



TEAM MEMBERS

WANYI (EMMY) LI

MICHELLE LUONG

MARYAM MAJD

TEAM MyDB

Database schema in the form of SQL CREATE TABLE. Specify key constraints and foreign key constraints in the schema.

```
DROP DATABASE IF EXISTS hotelDB;  
CREATE DATABASE hotelDB;  
USE hotelDB;
```

```
DROP TABLE IF EXISTS user;  
CREATE TABLE user (  
    firstName VARCHAR(20) NOT NULL,  
    lastName VARCHAR(20) NOT NULL,  
    username VARCHAR(20) NOT NULL,  
    password VARCHAR(20) NOT NULL,  
    age INT NOT NULL,  
    gender ENUM('M', 'F'),  
    userRole ENUM('Customer', 'Manager', 'Room Attendant'),  
    PRIMARY KEY(username)  
);
```

```
DROP TABLE IF EXISTS room;  
CREATE TABLE room (  
    roomID INT(10) NOT NULL AUTO_INCREMENT,  
    costPerNight DOUBLE(10,2) NOT NULL,  
    roomType VARCHAR(20) NOT NULL,  
    PRIMARY KEY(roomID)  
);
```

```
DROP TABLE IF EXISTS reservation;  
CREATE TABLE reservation (  
    reservationID INT AUTO_INCREMENT,  
    roomID INT(10) NOT NULL,  
    customerName VARCHAR(20) NOT NULL,  
    startDate DATE NOT NULL,  
    endDate DATE NOT NULL,  
    totalNumOfDays INT(10),  
    totalCost DOUBLE(10,2),  
    cancelled BOOLEAN NOT NULL DEFAULT FALSE,  
    updateOn TIMESTAMP DEFAULT current_timestamp ON UPDATE  
current_timestamp,
```

```

        PRIMARY KEY(reservationID),
        FOREIGN KEY(roomID) references room(roomID),
        FOREIGN KEY(customerName) references user(username)
    );
ALTER table reservation auto_increment = 1000;

DROP TABLE IF EXISTS roomService;
CREATE TABLE roomService (
    taskID INT(10) NOT NULL AUTO_INCREMENT,
    username VARCHAR(20) NOT NULL,
    task VARCHAR(150) NOT NULL,
    completedBy VARCHAR(20),
    reservationID INT(10),
    updateOn TIMESTAMP NOT NULL DEFAULT current_timestamp ON UPDATE
current_timestamp,
    PRIMARY KEY(taskID),
    FOREIGN KEY(completedBy) references user(username),
    FOREIGN KEY(username) references user(username),
    FOREIGN KEY(reservationID) references reservation(reservationID)
);

DROP TABLE IF EXISTS complaint;
CREATE TABLE complaint(
    complaintID INT(20) NOT NULL AUTO_INCREMENT,
    customer VARCHAR(20) NOT NULL,
    complaint VARCHAR(150) NOT NULL,
    time DATETIME NOT NULL DEFAULT CURRENT_TIMESTAMP,
    resolvedBy VARCHAR(20),
    solution VARCHAR(150),
    updateOn TIMESTAMP NOT NULL DEFAULT current_timestamp ON UPDATE
current_timestamp,
    PRIMARY KEY(complaintID),
    FOREIGN KEY(customer) references user(username),
    FOREIGN KEY(resolvedBy) references user(username)
);

DROP TABLE IF EXISTS ratingFeedback;
CREATE TABLE ratingFeedback (
    ratingID INT(20) NOT NULL AUTO_INCREMENT,
    customer VARCHAR(20) NOT NULL,

```

```

        rating INT NOT NULL,
        PRIMARY KEY(ratingID),
        FOREIGN KEY(customer) references user(username)
    );

DROP TABLE IF EXISTS reservationArchive;
CREATE TABLE reservationArchive (
    reservationID INT(10) NOT NULL AUTO_INCREMENT,
    roomID INT(10) NOT NULL,
    customerName VARCHAR(20) NOT NULL,
    startDate DATE NOT NULL,
    endDate DATE NOT NULL,
    totalNumOfDays INT(10),
    totalCost DOUBLE(10,2),
    cancelled BOOLEAN NOT NULL DEFAULT FALSE,
    updateOn TIMESTAMP DEFAULT current_timestamp ON UPDATE
current_timestamp,
    PRIMARY KEY(reservationID),
    FOREIGN KEY(roomID) references room(roomID),
    FOREIGN KEY(customerName) references user(username)
);

```

```

DROP TABLE IF EXISTS roomServiceArchive;
CREATE TABLE roomServiceArchive (
    taskID INT(10) NOT NULL AUTO_INCREMENT,
    username VARCHAR(20) NOT NULL,
    task VARCHAR(150) NOT NULL,
    completedBy VARCHAR(20),
    updateOn TIMESTAMP NOT NULL DEFAULT current_timestamp ON UPDATE
current_timestamp,
    PRIMARY KEY(taskID),
    FOREIGN KEY(completedBy) references user(username),
    FOREIGN KEY(username) references user(username)
);

```

```

DROP TABLE IF EXISTS complaintArchive;
CREATE TABLE complaintArchive(
    complaintID INT(20) NOT NULL AUTO_INCREMENT,
    customer VARCHAR(20) NOT NULL,

```

```

        complaint VARCHAR(150) NOT NULL,
        time DATETIME NOT NULL DEFAULT CURRENT_TIMESTAMP,
        resolvedBy VARCHAR(20),
        solution VARCHAR(150),
        updateOn TIMESTAMP NOT NULL DEFAULT current_timestamp ON UPDATE
current_timestamp,
        PRIMARY KEY(complaintID),
        FOREIGN KEY(customer) references user(username),
        FOREIGN KEY(resolvedBy) references user(username)
);

```

Screenshots of all relations after populating initial data. Label each relation clearly.

In our inputHotelDB.sql file, we populated the 'ROOM' table with the cost of the room per night as well as the room type. Here is the following SQL statement to populate data into the 'ROOM' table:

```

insert into room(costPerNight, roomType)
values (89, "Single Room"),(89, "Single Room"),(89, "Single Room"),
(89, "Single Room"),(89, "Single Room"),(89, "Single Room"),
(89, "Single Room"),(89, "Single Room"),(89, "Single Room"),
(89, "Single Room"),(89, "Single Room"),(89, "Single Room"),
(112, "Double Room"),(112, "Double Room"),(112, "Double Room"),
(112, "Double Room"),(112, "Double Room"),(112, "Double Room"),
(112, "Double Room"),(112, "Double Room"),(112, "Double Room"),
(112, "Double Room"),(112, "Double Room"),(112, "Double Room"),
(164, "Suite Room"),(164, "Suite Room"),(164, "Suite Room"),
(164, "Suite Room"),(164, "Suite Room"),(164, "Suite Room"),
(164, "Suite Room"),(164, "Suite Room"),(164, "Suite Room"),
(164, "Suite Room"),(164, "Suite Room"),(164, "Suite Room"),
(200, "Platinum Suite"),(200, "Platinum Suite"),(200, "Platinum Suite"),
(200, "Platinum Suite"),(200, "Platinum Suite"),(200, "Platinum Suite");

```

Here is the screenshot of the relation after the initial population of 'ROOM' data:

```
Command Prompt - mysql -u root -p

mysql> select * from room;
+-----+-----+-----+
| roomID | costPerNight | roomType |
+-----+-----+-----+
| 1 | 89.00 | Single Room |
| 2 | 89.00 | Single Room |
| 3 | 89.00 | Single Room |
| 4 | 89.00 | Single Room |
| 5 | 89.00 | Single Room |
| 6 | 89.00 | Single Room |
| 7 | 89.00 | Single Room |
| 8 | 89.00 | Single Room |
| 9 | 89.00 | Single Room |
| 10 | 89.00 | Single Room |
| 11 | 89.00 | Single Room |
| 12 | 89.00 | Single Room |
| 13 | 112.00 | Double Room |
| 14 | 112.00 | Double Room |
| 15 | 112.00 | Double Room |
| 16 | 112.00 | Double Room |
| 17 | 112.00 | Double Room |
| 18 | 112.00 | Double Room |
| 19 | 112.00 | Double Room |
| 20 | 112.00 | Double Room |
| 21 | 112.00 | Double Room |
| 22 | 112.00 | Double Room |
| 23 | 112.00 | Double Room |
| 24 | 112.00 | Double Room |
| 25 | 164.00 | Suite Room |
| 26 | 164.00 | Suite Room |
| 27 | 164.00 | Suite Room |
| 28 | 164.00 | Suite Room |
| 29 | 164.00 | Suite Room |
| 30 | 164.00 | Suite Room |
| 31 | 164.00 | Suite Room |
| 32 | 164.00 | Suite Room |
| 33 | 164.00 | Suite Room |
| 34 | 164.00 | Suite Room |
| 35 | 164.00 | Suite Room |
| 36 | 164.00 | Suite Room |
| 37 | 200.00 | Platinum Suite |
| 38 | 200.00 | Platinum Suite |
| 39 | 200.00 | Platinum Suite |
| 40 | 200.00 | Platinum Suite |
| 41 | 200.00 | Platinum Suite |
| 42 | 200.00 | Platinum Suite |
+-----+-----+-----+
42 rows in set (0.00 sec)

mysql>
```

Then we populated 'USER' relation. What we wanted to do was to have initial data for each user role, such as the manager, customer, and room attendant. Here are the following SQL statements populating the 'USER' relation:

```
insert into user(username, firstName, lastName, password, age, gender, userRole)
values('admin1', 'TestAdminF', 'TestAdminL', '123', 26, 'F', 'Manager');
```

```
insert into user(username, firstName, lastName, password, age, gender, userRole)
values('customer1', 'TestCustF', 'TestCustL', '2345', 21, 'F', 'Customer');
```

```
insert into user(username, firstName, lastName, password, age, gender, userRole)
values('roomattent1', 'RoomAttentF', 'RoomAttentL', '12345', 22, 'F', 'Room Attendant');
```

Here is the screenshot of the relation after the initial population of ‘USER’ data:

```
mysql> select * from user;
```

firstName	lastName	username	password	age	gender	userRole
TestAdminF	TestAdminL	admin1	123	26	F	Manager
TestCustF	TestCustL	customer1	2345	21	F	Customer
RoomAttentF	RoomAttentL	roomattent1	12345	22	F	Room Attendant

```
3 rows in set (0.00 sec)

mysql>
```

List of all the functions

1. Add account to the database
2. Get account information based on username
3. Get the user’s account based on username
4. Update/Change the user’s account information
5. Add reservations to the database
6. Get all the reservations made
7. Get a specific reservation based on the minimum or maximum total cost. This function is for the manager/administrator user role.
8. Cancel the reservation
9. Get all the users, and order them by userRole
10. Get all the users based on the number of reservations
11. Check to see if the user exists in the database. In other words, does the user have an account or not.
12. Get the user’s password to their account
13. Check the user’s password
14. Delete user from database
15. Get all available rooms based on the check-in time and check-out time
16. Add a complaint
17. Get all the complaints

18. Get a specific complaint based on the complaint ID
19. Update the complaint
20. Add room service based on username and task
21. Get room service
22. Resolve the room service task based on taskID
23. Update the rating
24. Get the receipt. This is basically the hotel's statistics of total number of users in the system, total number of customers, total number of managers, average age of the users, average number of reservations, average cost, and average customer rating. This is for the manager/administrator user role.

SQL SELECT STATEMENTS

❑ SELECT *
FROM USER
WHERE username = 'username';

❑ SELECT username, firstName, lastName, userRole, age, gender
FROM USER WHERE username = 'username';

SELECT customerName, cancelled, reservationID, ROOM.roomID, startDate, endDate,
totalNumOfDays, totalCost, costPerNight, roomType
FROM ROOM right outer join RESERVATION ON ROOM.roomID =
RESERVATION.roomID
WHERE customer = 'username';

❑ SELECT*
FROM ROOM
where roomID = roomID;

❑ SELECT cancelled, customerName, reservationID, ROOM.roomID, startDate, endDate,
totalNumOfDays, totalCost, costPerNight, roomType
FROM ROOM right outer join RESERVATION
ON ROOM.roomID = RESERVATION.roomID;

❑ SELECT cancelled, customerName, reservationID, ROOM.roomID, startDate, endDate,
totalNumOfDays, totalCost, costPerNight, roomType
FROM ROOM right outer join RESERVATION
ON ROOM.roomID = RESERVATION.roomID
HAVING totalCost >= min AND totalCost <= max
ORDER BY orderBy;

- ❑ SELECT *
FROM USER
ORDER by userRole;
- ❑ SELECT username FROM USER;
right outer join (
 SELECT customerName, reservationID, ROOM.roomID, startDate, endDate,
 totalNumOfDats, totalCost, costPerNight, roomType
 FROM ROOM right outer join RESERVATION
 ON ROOM.roomID = RESERVATION.roomID
 GROUP BY customerName
 HAVING COUNT(*) >= numOfReservations AS reservations
 ON USER.username = RESERVATION.customerName;
- ❑ SELECT username
FROM USER
- ❑ SELECT password
FROM USER
WHERE username = 'username';
- ❑ SELECT *
FROM ROOM
WHERE roomID NOT IN
 (SELECT distinct ROOM.roomID
 FROM ROOM LEFT OUTER JOIN RESERVATION
 ON ROOM.roomID = RESERVATION.roomID
 WHERE = RESERVATION.startDate
 OR = RESERVATION.endDate
 OR = RESERVATION.startDate
 OR = RESERVATION.endDate
 OR (RESERVATION.startDate < AND RESERVATION.endDate >)
 OR (< RESERVATION.startDate and > RESERVATION.startDate)
 OR (< RESERVATION.endDate and > RESERVATION.endDate));
- ❑ SELECT *
FROM COMPLAINT;
- ❑ SELECT *

```
FROM COMPLAINT
WHERE complaintID = id;
```

- ❑ SELECT *
FROM roomService
WHERE completedBy is NULL;
- ❑ SELECT * FROM ratingFeedback WHERE customer = 'username';
- ❑ SELECT COUNT(*) FROM USER;
SELECT COUNT(*) FROM USER WHERE userRole = 'Customer';
SELECT COUNT(*) FROM USER WHERE userRole = 'Manager';
SELECT AVG(age) FROM USER;
SELECT AVG(reservationCount) FROM (SELECT COUNT(*) AS reservationCount;
FROM RESERVATION GROUP BY customerName) counts;
SELECT AVG(totalCost) FROM RESERVATION;
SELECT AVG(rating) FROM ratingFeedback;

SQL UPDATE STATEMENTS

- ❑ UPDATE USER
SET password = 'password', firstName = 'firstName', lastName = 'lastName'
age = age, gender = 'gender'
WHERE username = 'username';
- ❑ UPDATE RESERVATION
SET cancelled = true
WHERE reservationID = 'reservationID';
- ❑ UPDATE complaint
SET resolvedBy = resolvedBy, solution = solution
WHERE complaintID = id;
- ❑ UPDATE roomService
SET completedBy = 'username'
WHERE taskID = taskID;
- ❑ UPDATE ratingFeedback
SET rating = rate
WHERE customer = 'username'

SQL DELETE STATEMENTS

- ❑ DELETE
FROM USER
WHERE username = 'username';
- ❑ delete
from roomService
where username = 'username';
- ❑ delete
from ratingFeedback
where customer = 'username';
- ❑ delete
from complaint
where customer = 'username';
- ❑ delete
from reservation
where customerName = 'username';

SQL INSERT STATEMENTS

- ❑ INSERT INTO USER(username, firstName, lastName, userRole, age, gender)
VALUES('%s','%s','%s','%s',%d,'%s','%s');
- ❑ INSERT INTO Reservation(roomID, customerName, startDate, endDate,
totalNumOfDays, totalCost)
VALUES(%d,'%s',%s,%s,%d,%f);
- ❑ INSERT INTO COMPLAINT(customer, complaint)
VALUES('%s', '%s');
- ❑ INSERT INTO roomService(username, task)
VALUES('%s', '%s');
- ❑ INSERT into ratingFeedback (customer, rating)
VALUES('%s', '%d');

SQL TRIGGERS

- ❑ DROP TRIGGER IF EXISTS InsertReservation;
delimiter //
CREATE TRIGGER InsertReservation
AFTER INSERT ON reservation
FOR EACH ROW
BEGIN
Delete From reservation
where new.reservationID = reservationID and exists (select * From reservation where
new.roomID = roomID and
((new.startDate >= startDate and new.startDate<= endDate) or(new.endDate >=
startDate and new.endDate <= endDate)));
END;
- ❑ delimiter ;
DROP TRIGGER IF EXISTS InsertReservation;
delimiter //
CREATE TRIGGER DeleteRoomService
AFTER UPDATE ON reservation
FOR EACH ROW
BEGIN
IF NEW.cancelled = TRUE THEN
DELETE FROM roomService
where reservationID = NEW.reservationID;
END IF;
END;

SQL STORED PROCEDURE

```
DROP PROCEDURE IF EXISTS archiveAll;  
DELIMITER //  
CREATE PROCEDURE archiveAll (IN cutoffDate TIMESTAMP)  
BEGIN  
    START TRANSACTION;  
    INSERT INTO reservationArchive(reservationId, roomId, customerName,  
        startDate, endDate, totalNumOfDays, totalCost, cancelled, updateOn)  
    SELECT reservationId, roomId, customerName, startDate, endDate,  
        totalNumOfDays, totalCost, cancelled, updateOn
```

```
FROM reservation
WHERE DATE(updateOn) <= cutoffDate;

INSERT INTO roomServiceArchive(taskId, username, task,
    completedBy, updateOn)
SELECT taskId, username, task, completedBy, updateOn
FROM roomservice
WHERE DATE(updateOn) <= cutoffDate;

INSERT INTO complaintArchive(complaintId, customer, complaint,time,
    resolvedBy ,solution, updateOn)
SELECT complaintId, customer, complaint, time, resolvedBy, solution, updateOn
FROM complaint
WHERE DATE(updateOn) <= cutoffDate;

DELETE FROM RESERVATION WHERE DATE(updateOn) <= cutoffDate;
DELETE FROM ROOMSERVICE WHERE DATE(updateOn) <= cutoffDate;
DELETE FROM COMPLAINT WHERE DATE(updateOn) <= cutoffDate;
COMMIT;
END;
```

CUSTOMER

Hotel Reservation System

Welcome!

Username

Password

Login

Register

User log in if customer has existing account or choose to register a new account

Registration

First name:

Last name:

Username:

Password:

Confirm Password:

Age:

Gender:

Back **Register**

Add account to the database: If user does not have existing account, the customer can fill out the registration page to create a new account.

The screenshot shows a web application window titled "Hotel Reservation System". At the top, it displays the user's login information: "Username: emmyli Name: emmy li Role: Customer". To the right of this information is a "Sign Out" button. Below the login information, there is a vertical list of six buttons: "Book a reservation", "View/Cancel Reservations", "Room Service Request", "File Complaint", "Rating", and "Setting".

After logging in with username and password, the customer's main menu will show.

The screenshot shows a web application window titled "Hotel Reservation System" with a sub-header "Reserve a Room". The form contains two input fields for dates: "Check-in (MM/DD/YYYY):" with the value "12/11/2017" and "Check-out (MM/DD/YYYY):" with the value "12/15/2017". Below these fields is a "Search for rooms" button. Underneath the search button is a large, empty rectangular box for displaying search results. At the bottom of the form, there are two buttons: "Book" and "Back to main menu".

Adding reservations to the database: Make reservation with check in date and check out date of user's choice

Hotel Reservation System

Reserve a Room

Check-in (MM/DD/YYYY): 12/11/2017 Check-out (MM/DD/YYYY): 12/15/2017

Search for rooms

- Single Room \$89.0 ight
- Single Room \$89.0 ight
- Single Room \$89.0 ight
- Single Room \$89.0 ight
- Double Room \$112.0 ight
- Double Room \$112.0 ight
- Double Room \$112.0 ight
- Double Room \$112.0 ight
- Double Room \$112.0 ight
- Double Room \$112.0 ight

Book

Back to main menu

When click ‘search for rooms’, it will display all available room types and prices per night at that time. Customer will click on desired room, and click ‘Book’.

Hotel Reservation System

Reserve a Room

Check-in (MM/DD/YYYY): 12/11/2017 Check-out (MM/DD/YYYY): 12/15/2017

Confirmation

Do you want to make this reservation?

否(N) 是(Y)

Single Room \$89.0 ight

Single Room \$89.0 ight

Single Room \$89.0 ight

Single Room \$89.0 ight

Double Room \$112.0 ight

Double Room \$112.0 ight

Double Room \$112.0 ight

Double Room \$112.0 ight

Double Room \$112.0 ight

Double Room \$112.0 ight

Book

Back to main menu

Make reservation for the current customer with a confirmation.

Hotel Reservation System

Receipt

Username: emmyli
 Name: emmy li
 Reservations made: 1

Reservation # 1
 12/11/2017
 to 12/15/2017
 Total Cost: 448.000000
 Total: \$448.00

[Back to main menu](#)

126 user = new User(rs.getString("username"), rs.getString("firstName"),

Print out receipt of the reservation with total price and start date and end date

Hotel Reservation System

File a Complaint

We apologize for any inconvenience.
 Please file your complaint here and a hotel manager
 will contact you as soon as possible.

[Submit](#)

[Back](#)

Hotel Reservation System

File a Complaint

We apologize for any inconvenience.
 Please file your complaint here and a hotel manager
 will contact you as soon as possible.

[Submit](#)

[Back](#)

Result
 Your complaint has been filed.
[确定](#)

Customer can add complaint of their services

View or Cancel a Reservation

Below are all your reservations.

To cancel a reservation, select the one you with the cancel. Press cancel.

If the list is empty, then you do not have any reservations

12/20/2017	to 12/21/2017	Total Cost: 0.000000
12/11/2017	to 12/12/2017	Total Cost: 0.000000
12/11/2017	to 12/13/2017	Total Cost: 0.000000
12/10/2017	to 12/11/2017	Total Cost: 89.000000
12/11/2017	to 12/15/2017	Total Cost: 448.000000

Cancel

Back to main menu

View or Cancel a Reservation

Below are all your reservations.

To cancel a reservation, select the one you with the cancel. Press cancel.

If the list is empty, then you do not have any reservations

12/11/2017	to 12/12/2017	Total Cost: 0.000000
12/11/2017	to 12/13/2017	Total Cost: 0.000000
12/10/2017	to 12/11/2017	Total Cost: 89.000000
12/11/2017	to 12/15/2017	Total Cost: 448.000000

Cancel

Back to main menu

View or Cancel a Reservation


Below are all your reservations.

To cancel a reservation, select the one you with the cancel. Press cancel.

If the list is empty, then you do not have any reservations

12/20/2017	to 12/21/2017	Total Cost: 0.000000
12/11/2017	to 12/12/2017	Total Cost: 0.000000
12/11/2017	to 12/13/2017	Total Cost: 0.000000
12/10/2017	to 12/11/2017	Total Cost: 89.000000
12/11/2017	to 12/15/2017	Total Cost: 448.000000

Confirmation



Are you sure you want to cancel this reservation?

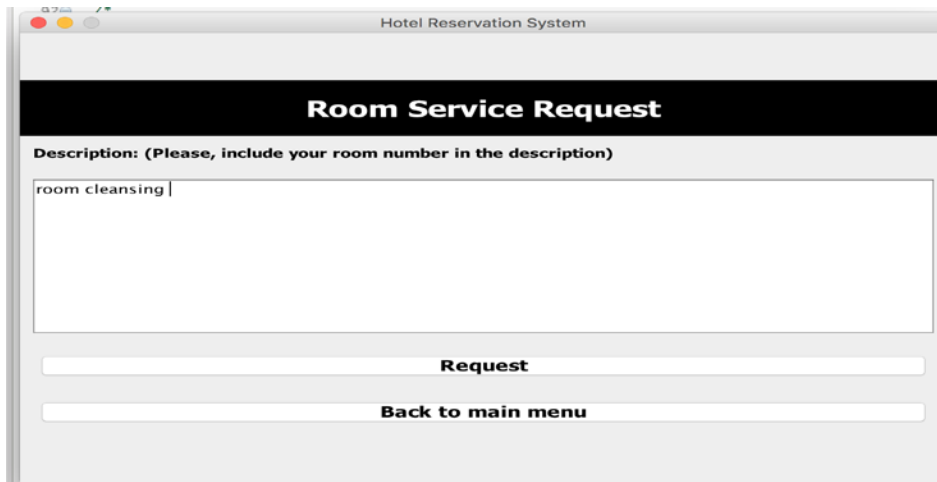
否(N)

是(Y)

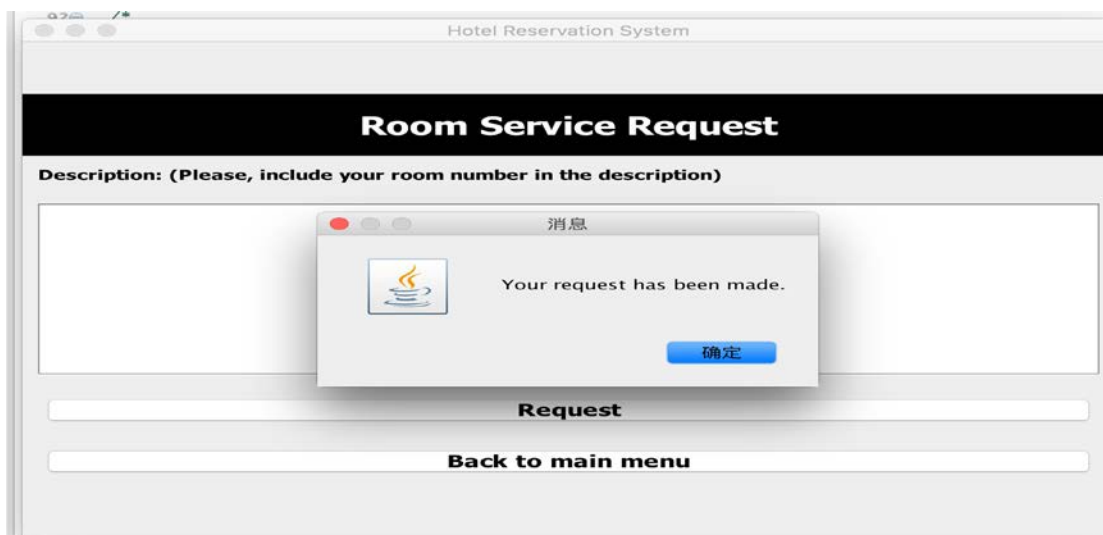
Cancel

Back to main menu

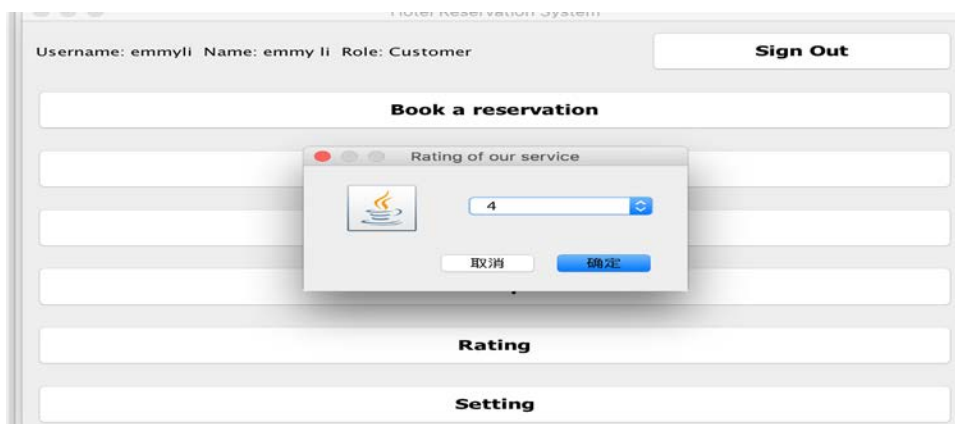
Customer can view all the reservations he or she have made. The customer can also cancel an existing reservation.



Customer can request a room service. This request will be sent to the room attendant account.



Confirmation that service request has been made



Customers can give a rating of the hotel (Rating from 1 - 5).

Hotel Reservation System

Account Settings

First Name:

Last Name:

Password:

Confirm Password:

Age:

Gender:

126 user = new User(rs.getString("username"), rs.getString("firstName"),

If customers wish to make changes to their existing account information, they can do so in Account Settings.

ADMINISTRATOR/MANAGER

Hotel Reservation System

Welcome!

Username

Password

Hotel Reservation System

Username: admin1 Name: TestAdminF TestAdminL Role: Manager

-
-
-
-
-
-

Administrator login; Main Menu for Admin.

Reservations

Enter a min and max and sort by room or customer.

Min cost (optional)
Max cost (optional)

Total reservations: 4

Reservation # 1017
 Username: emmyli
 Room: Double Room
 Start: 12/11/2017
 End: 12/13/2017
 Cost: \$0.0

Reservation # 1019
 Username: emmyli
 Room: Single Room
 Start: 12/10/2017

Manager can view reservations sorted by room with minimal cost of 0 and max cost of 400.

Reservations

Enter a min and max and sort by room or customer.

Min cost (optional)
Max cost (optional)

Username: emmyli
 Room: Single Room
 Start: 12/10/2017
 End: 12/13/2017
 Cost: \$267.0
 This reservation has been cancelled

Reservation # 1019
 Username: emmyli
 Room: Single Room
 Start: 12/10/2017
 End: 12/11/2017
 Cost: \$89.0

Manager can also view reservations sorted by customers with minimal cost of 0 and max cost of 400

Hotel Reservation System

Customer Complaints

Number of complaints: 1
 Username: emmyli
 Complaint ID: 3
 Filed on: 12/10/2017
 Complaint: the uncomplete cleansing of the room

Enter a complaint ID and solution to resolve the complaint.

Complaint ID:

3

Solution:

told the attendant and cleaned the room

Back to main menu **Resolve Complaint**

Customer Complaints

Number of complaints: 1
 Username: emmyli
 Complaint ID: 3
 Filed on: 12/10/2017
 Complaint: the uncomplete cleansing of the room
 Resolved By: admin1
 Solution: told the attendant and cleaned the room

Enter a complaint ID and solution to resolve the complaint.

Complaint ID:

Solution:

Back to main menu **Resolve Complaint**

Result
 Complaint resolved
 确定

Managers can take care of resolving existing complaints made by customers.

Manage System Users

admin1 TestAdminF TestAdminL Manager 26 F
 attendant attent t Room Attendant 21 M
 attendant2 bjh xdfdf Room Attendant 23 M
 customer1 TestCustF TestCustL Customer 21 F
 emmyli emmy li Customer 22 M
 roomattent1 RoomAttentF RoomAttentL Room Attendant 23 M

Add Employee **Delete**

Back to main menu

Manager can also manage system users (administrator, attendant, customer). This will show all the users with existing accounts. Administrators are in magenta, Customers are in blue, and Room Attendants are in green.

Hotel Reservation System

Add Employee

Account type:

Select Role

First name:

Last name:

Username:

Password:

Confirm Password:

Age:

Select Age

Gender:

Select Gender

Add

Back

Hotel Reservation System

Add Employee

Account type:

Manager

First name:

admin

Last name:

ad

Username:

admin2

Password:

...

Confirm Password:

...

Age:

30

Gender:

Female

Add

Back

Manage System Users

admin1 TestAdminF TestAdminL Manager 26 F

admin2 admin ad Manager 30 M

attendant attent t Room Attendant 21 M

attendant2 bjh xdfdf Room Attendant 23 M

customer1 TestCustF TestCustL Customer 21 F

emmyli emmy li Customer 22 M

roomattent1 RoomAttentF RoomAttentL Room Attendant 23 M

test test test Customer 19 M

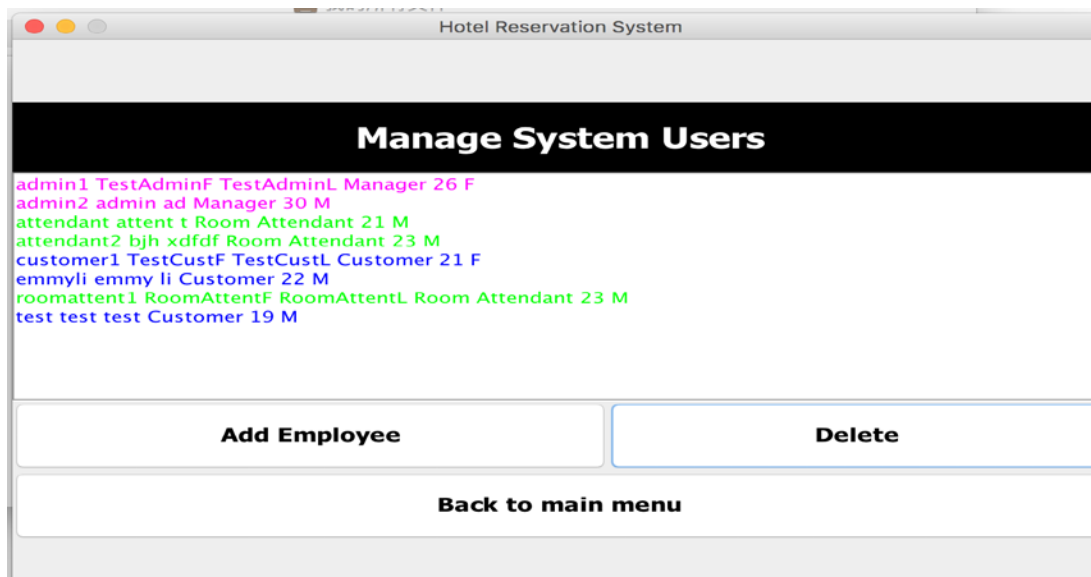
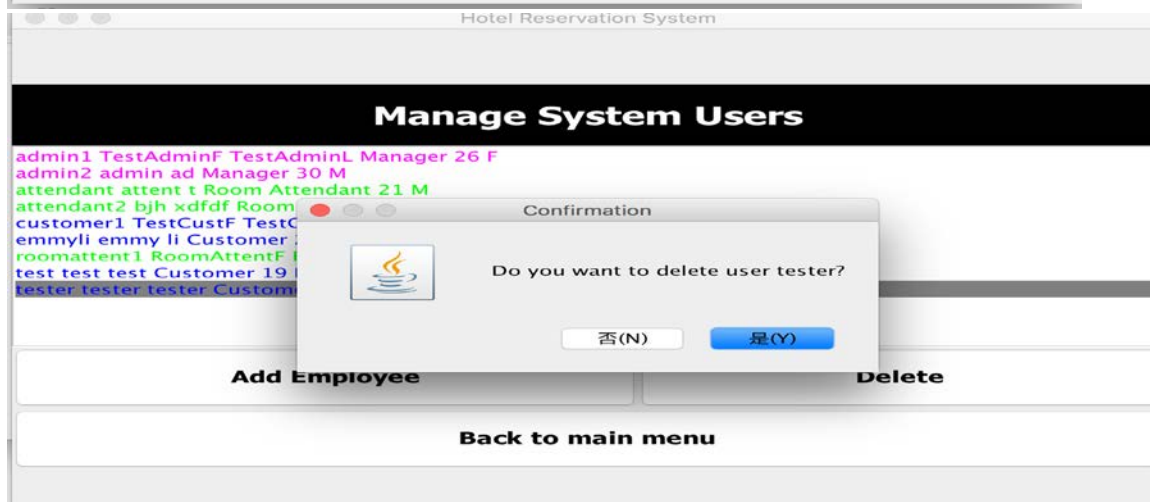
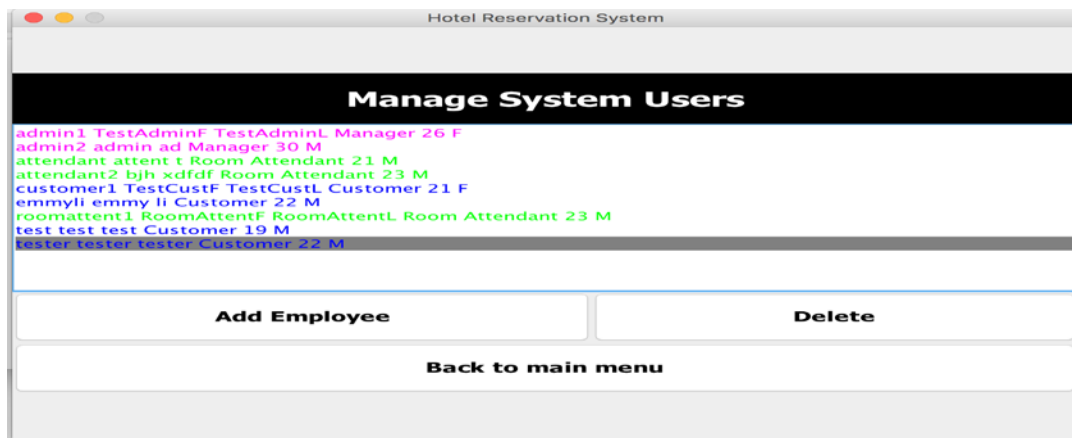
tester tester tester Customer 22 M

Add Employee

Delete

Back to main menu

Manager can add a new employee



Manager can also delete an existing employee

Hotel Reservation System

Archive

Enter a date. Reservations, room service requests, and complaints will be archived from this date.

Date to archive from (MM/DD/YYYY):

Result

Archive Successful

确定

Back to main menu

Archive data in database

```
bin — mysql -u root — 80x24
```

```
[mysql> select * from reservationArchive;
```

reservationID	roomID	customerName	startDate	endDate	totalNumOfDays	totalCost	cancelled	updateOn
1015	2	emmyli	2017-12-20	2017-12-21				
1016	8	emmyli	2017-12-11	2017-12-12				
1017	24	emmyli	2017-12-11	2017-12-13				
1018	3	emmyli	2017-12-10	2017-12-13				
1019	6	emmyli	2017-12-10	2017-12-11				
1020	13	emmyli	2017-12-11	2017-12-15				

```
6 rows in set (0.00 sec)
```

Here is what is populated into the reservationArchive relation

```
0 rows in set (0.00 sec)
```

```
[mysql> select * from roomServiceArchive;
```

taskID	username	task	completedBy	updateOn
5	emmyli	room cleansing	attendant	2017-12-09 18:56:25

```
1 row in set (0.01 sec)
```

```
mysql>
```

Here is what is populated into the roomServiceArchive relation

```

7 rows in set (0.01 sec)

mysql> select * from complaintArchive;
+-----+-----+-----+-----+
| complaintID | customer | complaint | time |
| resolvedBy | solution | updateOn |
+-----+-----+-----+-----+
| 3 | emmyli | the uncompelete cleansing of the room | 2017-12-10 15:51:42 |
| admin1 | told the attendant and cleaned the room | 2017-12-10 15:53:50 |
| 4 | emmyli | testing | 2017-12-10 15:55:48 |
| NULL | NULL | 2017-12-10 15:55:48 |
+-----+-----+-----+-----+
2 rows in set (0.01 sec)

mysql>

```

Here is what is populated into the complaintArchive relation

Statistics	
Total employee: 4 Number of manager: 1 Number of room attendant: 3 Number of registered customer: 3 Average age of users: 22.57 Average number of reservations per customer: 6.00 Average cost of a reservation: 160.80 Average rating from customer: 4.00	
Back to main menu	

Statistics for hotel system

Hotel Reservation System

Account Settings

First Name:

TestAdminF

Last Name:

TestAdmin change

Password:

...

Comfirm Password:

...

Age:

27

Gender:

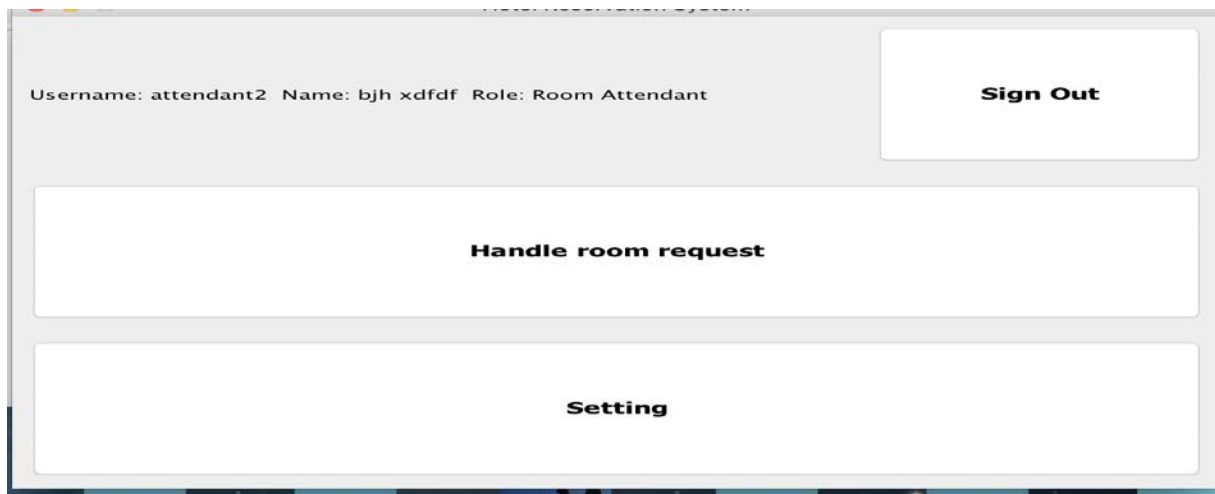
M

Back

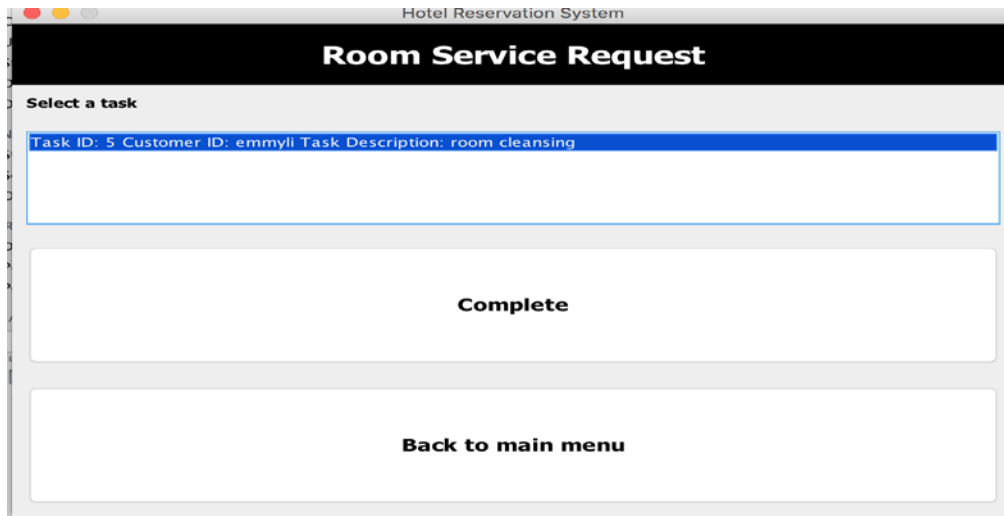
Change

Manager can also go into Settings to update and change account information.

ROOM ATTENDANT



Attendant log in, main menu



Check room service request and complete the request

SCREENSHOTS OF POPULATED TABLES

```

[mysql> select * from Complaint;
Empty set (0.02 sec)

[mysql> select * from complaintArchive;
+-----+
| complaintID | customer | complaint | resolvedBy | solution | time |
+-----+
| 3 | emmyli | the uncomplete cleansing of the room | 2017-12-10 15:51:42 | admin1 | told the attendant and cleaned the room | 2017-12-10 15:53:50 |
| 4 | emmyli | testing | NULL | NULL | 2017-12-10 15:55:48 |
+-----+

```

Complaint and complaintArchive.

Currently Complaint table is empty, because all the complaint were archived into complaintArchive

```

2 rows in set (0.00 sec)

[mysql> select * from ratingFeedback;
+-----+
| ratingID | customer | rating |
+-----+
| 3 | emmyli | 4 |
+-----+
1 row in set (0.01 sec)

mysql>

```

ratingFeedback

```

[mysql> select * from reservation;
Empty set (0.00 sec)

[mysql> select * from reservationArchive;
+-----+
| reservationID | roomID | customerName | startDate | endDate | totalNumOfDays | totalCost | cancelled | updateOn |
+-----+
| 1015 | 2 | emmyli | 2017-12-20 | 2017-12-21 | 2 | NULL | 0 | 2017-12-10 16:04:07 |
| 1016 | 8 | emmyli | 2017-12-11 | 2017-12-12 | 2 | 0.00 | 0 | 2017-12-08 01:14:58 |
| 1017 | 24 | emmyli | 2017-12-11 | 2017-12-13 | 3 | 0.00 | 0 | 2017-12-08 01:16:03 |
| 1018 | 3 | emmyli | 2017-12-10 | 2017-12-13 | 4 | 267.00 | 1 | 2017-12-08 11:52:59 |
| 1019 | 6 | emmyli | 2017-12-10 | 2017-12-11 | 2 | 89.00 | 0 | 2017-12-08 11:51:40 |
| 1020 | 13 | emmyli | 2017-12-11 | 2017-12-15 | 5 | 448.00 | 0 | 2017-12-09 11:28:33 |
+-----+

```

Reservation and reservationArchive

Currently reservation table is empty because all data are archived into reservationArchive

6 rows in set (0.00 sec)

mysql> select * from room;

roomID	costPerNight	roomType
1	89.00	Single Room
2	89.00	Single Room
3	89.00	Single Room
4	89.00	Single Room
5	89.00	Single Room
6	89.00	Single Room
7	89.00	Single Room
8	89.00	Single Room
9	89.00	Single Room
10	89.00	Single Room
11	89.00	Single Room
12	89.00	Single Room
13	112.00	Double Room
14	112.00	Double Room
15	112.00	Double Room
16	112.00	Double Room
17	112.00	Double Room
18	112.00	Double Room
19	112.00	Double Room
20	112.00	Double Room
21	112.00	Double Room
22	112.00	Double Room
23	112.00	Double Room
24	112.00	Double Room
25	164.00	Suite Room
26	164.00	Suite Room
27	164.00	Suite Room
28	164.00	Suite Room
29	164.00	Suite Room
30	164.00	Suite Room
31	164.00	Suite Room
32	164.00	Suite Room
33	164.00	Suite Room
34	164.00	Suite Room
35	164.00	Suite Room
36	164.00	Suite Room
37	200.00	Platinum Suite
38	200.00	Platinum Suite
39	200.00	Platinum Suite
40	200.00	Platinum Suite
41	200.00	Platinum Suite
42	200.00	Platinum Suite

42 rows in set (0.01 sec)

Room

Empty set (0.00 sec)

mysql> select * from roomService;

Empty set (0.00 sec)

mysql> select * from roomServiceArchive

taskID	username	task	completedBy	updateOn
5	emmyli	room cleansing	attendant	2017-12-09 18:56:25

1 row in set (0.00 sec)

mysql> █

roomService and roomServiceArchive

Currently all roomService is empty all data is in the roomServiceArchive

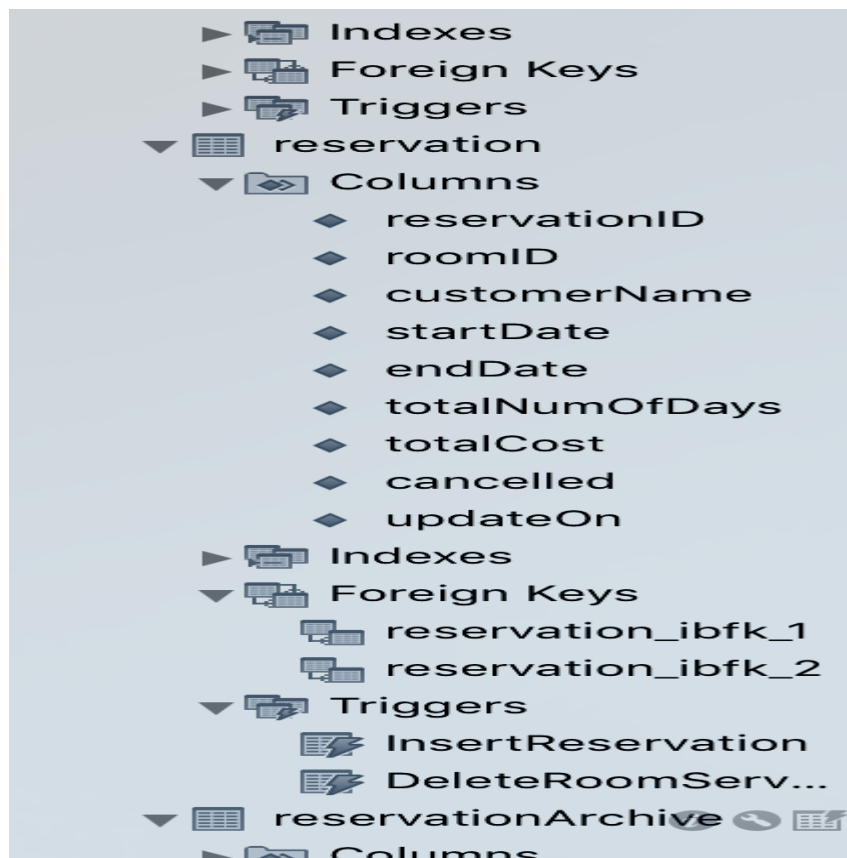
```
1 row in set (0.00 sec)

mysql> select * from user;
```

firstName	lastName	username	password	age	gender	userRole
TestAdminF	change	admin1	123	27	M	Manager
admin	ad	admin2	123	30	M	Manager
attendant	t	attendant	12345	21	M	Room Attendant
bjh	xdfdf	attendant2	12345	23	M	Room Attendant
TestCustF	TestCustL	customer1	2345	21	F	Customer
emmy	li	emmyli	12345	22	M	Customer
RoomAttentF	RoomAttentL	roomattent1	12345	23	M	Room Attendant

user

Screenshots of key constraint and foreign key constraint violations



```

9 rows in set (0.01 sec)

mysql> INSERT INTO reservation(roomID,customerName, startDate, endDate, totalNum
OfDays, totalCost, cancelled)
-> values ( 37, 'customer1',( SELECT STR_TO_DATE('03,12,2017','%d,%m,%Y')), (
SELECT STR_TO_DATE('04,12,2017','%d,%m,%Y')),1,200,FALSE);
Query OK, 1 row affected (0.04 sec)

mysql> INSERT INTO reservation(roomID,customerName, startDate, endDate, totalNum
OfDays, totalCost, cancelled)
-> values ( 37, 'customer1',( SELECT STR_TO_DATE('03,12,2017','%d,%m,%Y')), (
SELECT STR_TO_DATE('04,12,2017','%d,%m,%Y')),1,200,FALSE);
ERROR 1644 (45000): Date insert conflicts with another date
mysql>

```

Trigger in reservation and Trigger of time conflict