CS 631-002-SPRING 2017 – Online Hotel Reservation System Project

Deliverable 3

GROUP3

LIN TANG

YUHAO SHI

XIAOXI YANG

PROFESSOR:

Dimitrios Theodoratos

Ananya Dass

INTRODUCTION

Goal

The goal of this part is translating our Relational schema to a real world database framework, populate the database, and create a website for Hulton.

Revisions

1. Adding a new attribute named c_id in table of credit_card, it is a foreign key to table customer c_id. Based on this action, every customer can store a lots of credit cards in our database, they can choose any card to pay their reservation.
2. Creating a new table named manager, to store the managers' account information. The recommendation attribute is used to give some good news to the customer, eg. discount information.

Problems Encountered

This is my first time to use eclipse to develop a website, too many things need to be dealt with. This is not a perfect website, but we already do our best.

1. Tomcat port default is 8080, but on our computer it always shows a error message that says this port is already in use, so we change to port of tomcat to 8081.
2. To develop a fancy interface for our website is too difficult.
3. It is hard to let a room reservation include multiple breakfast/service types and can have multiple orders for each breakfast type, and the total number of orders for different breakfast types for a room reservation should not exceed the number of persons that can stay in that room. By referring a lot of information on the internet, we solved this problem with using JQuery.

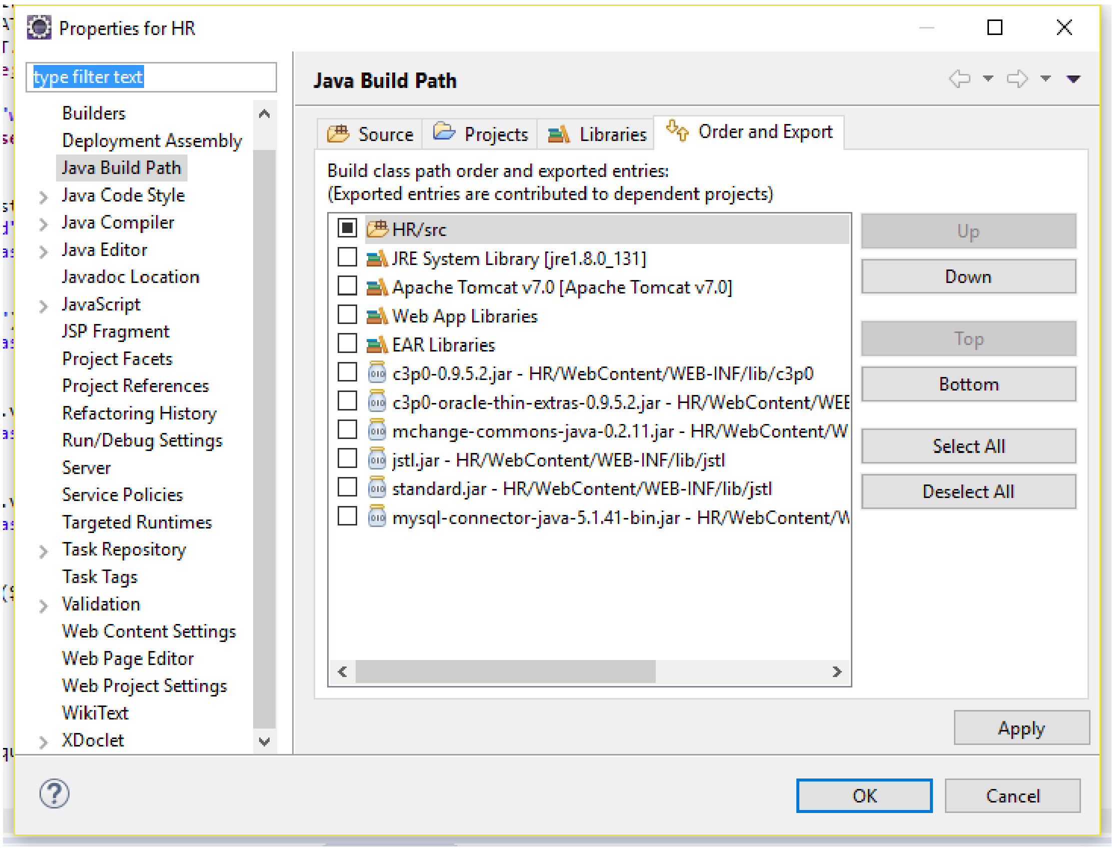
Application Implementation

Source code folder:
HR (using Java programming language, JSP and JQuery, JS)

Tools that we use:

1. JDK 1.8.0
2. Eclipse IDE for Java EE Developers
   Version: Neon.1a Release (4.6.1)
3. apache-tomcat-7.0.73
4. MySQL server 5.7

5. Navicat for MYSQL 10.0.11



Build class path order and exported entries:
(Exported entries are contributed to dependent projects)

■ HR/src
□ JRE System Library [jre1.8.0_131]
□ Apache Tomcat v7.0 [Apache Tomcat v7.0]
□ Web App Libraries
□ EAR Libraries
□ c3p0-0.9.5.2.jar - HR/WebContent/WEB-INF/lib/c3p0
□ c3p0-oracle-thin-extras-0.9.5.2.jar - HR/WebContent/WEB
□ mchange-commons-java-0.2.11.jar - HR/WebContent/W
□ jstl.jar - HR/WebContent/WEB-INF/lib/jstl
□ standard.jar - HR/WebContent/WEB-INF/lib/jstl
□ mysql-connector-java-5.1.41-bin.jar - HR/WebContent/W

Programming Concept

Our entire website is create as an mvc model, which is Model–view–controller. MVC is a software architectural pattern for implementing user interfaces on computers. It divides a given application into three interconnected parts in order to separate internal representations of information from the ways that information is presented to and accepted from the user. The MVC design pattern decouples these major components allowing for efficient code reuse and parallel development.

We divided the source code into 6 layers, which are dao, domain, action, service and util.

(1) Dao layer: it is responsible for the communication between the entire program and the database.

(2) Domain layer: it is contain some classes which is one by one corresponding to the database table, and some other classes. It packages some data, so that the parameters can easy to be passed. Some key database commands are as follows:
Compute the highest rated room type for each hotel:
Compute the 5 best customers:
Compute the highest rated breakfast type across all hotels:
Compute the highest rated service type across all hotels:

(3) Sevice layer: it is the glue that connects the dao layer and the action layer. It can get an available room that can be reserved by a customer, calculate the final total amount and

pay the reservation bill.

We create a class named FindARoom. Firstly, finding all the rooms in a hotel, then based on the room to find its reservation time in room reservation table, if the time is not conflict with the customer's reservation time, return the room number of this room. If conflict, then continue to find another avaliable room. If cannot find a available room, return 0.

We create a class named EvaluatePrice. It calculate the sum price of all the orders. Because some room has a discount period, we need to compare the check in and check out date of the customer with the discount date. Getting which days have the discount right. Then sum the room price, breakfast price and service price.

We create a class named Pay. This class inputed a room reservation bill, reservation amount, name of the customer and credit card number. Before input the room reservation information, we need store the invoice number of this bill into database reservation table (the invoice number is created in util layer). Then traversal the room reservation bill, store all the detail information into database.

(4) Action layer: this layer is equal to the control layer of MVC, it is responsible for calling the function of service layer and processing the data, then return the result to jsp.

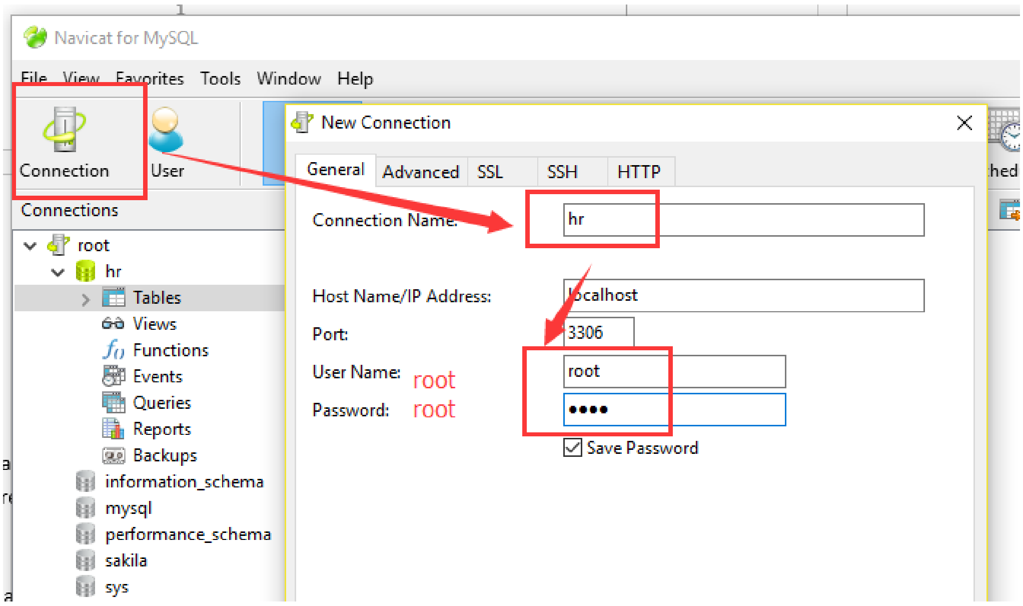
(5) Jsp layer: it is the same as the view layer of MVC. We store all the jsp files which can not see by the user in the WEB-INF folder. The page can use by the users we put them in webcontent folder. In jsp communicate with server in two ways. One is in the form, another is using href of a label.

(6) Util layer: it contains MyUtil.java and FillData.java. By running the FillData.java, can populate the tables in database. MyUtil.java is used to generate a invoice number of reservation.

Database Creation

Our database is based on Mysql, and we use Navicat for MySQL to create tables and manage data in the database. To see all SQL commands of table creation, please see the <SQL Commands of Table Creation> part.

1. Creating a connection between mysql and Navicat, click Connection -> type the connection information -> click OK. Then you can see connection icon appear on the left named root.



Connection Name: hr

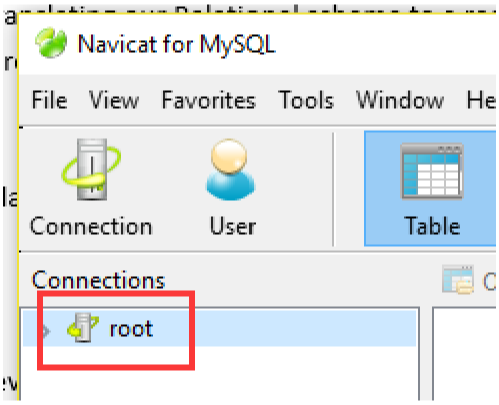
Host Name/IP Address: localhost

Port: 3306

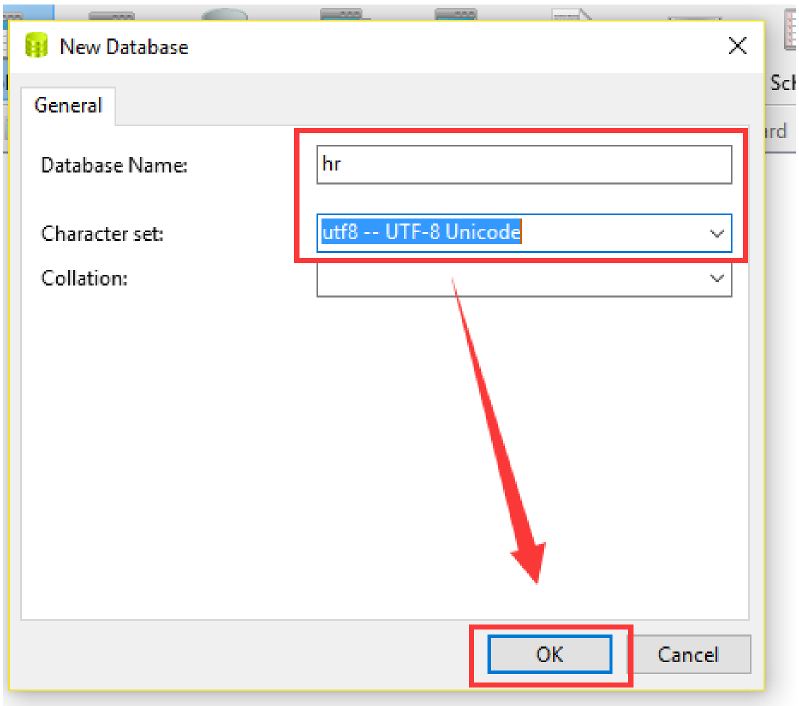
User Name: root root

Password: root ••••

☑ Save Password



2. Click the connection icon -> right click world -> click New Database… -> type information as follow -> click OK.

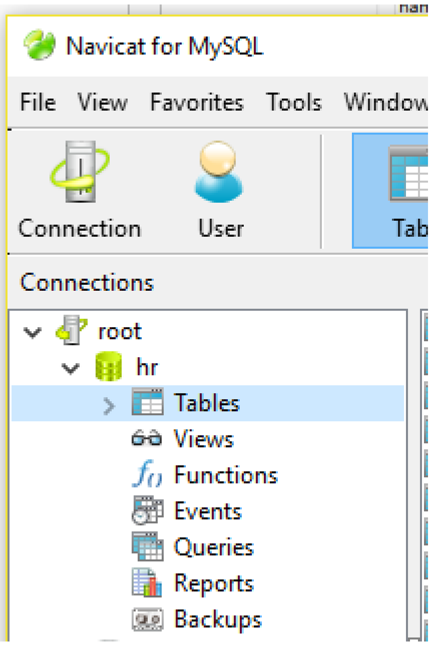


Database Name: hr

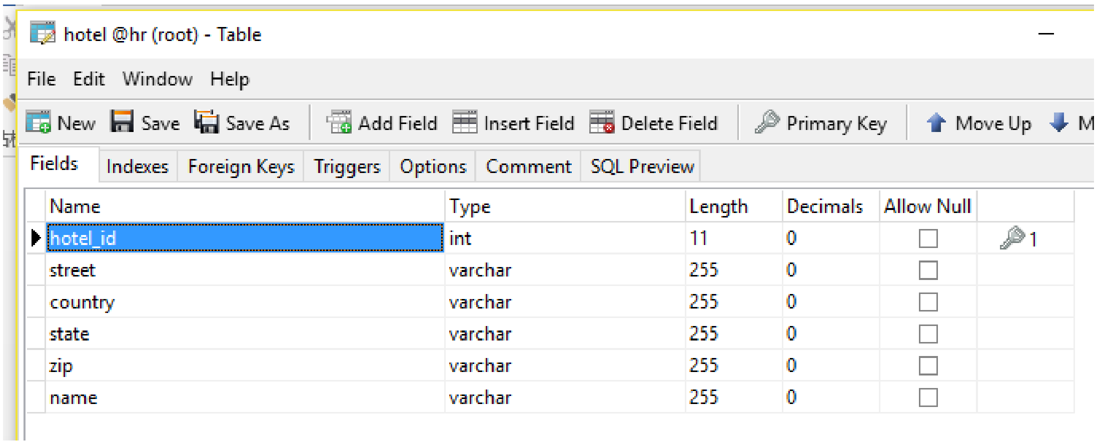
Character set: utf8 -- UTF-8 Unicode

Collation:

3. Click hr database, right click Tables -> New Table.

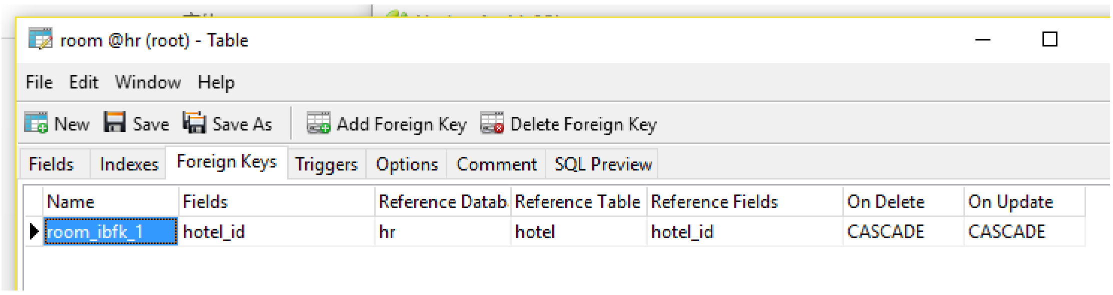


4. Based our relation schema, insert the attribute names, primary keys, foreign keys and so on


hotel @hr (root) - Table
File Edit Window Help
New Save Save As Add Field Insert Field Delete Field Primary Key Move Up M
Fields Indexes Foreign Keys Triggers Options Comment SQL Preview

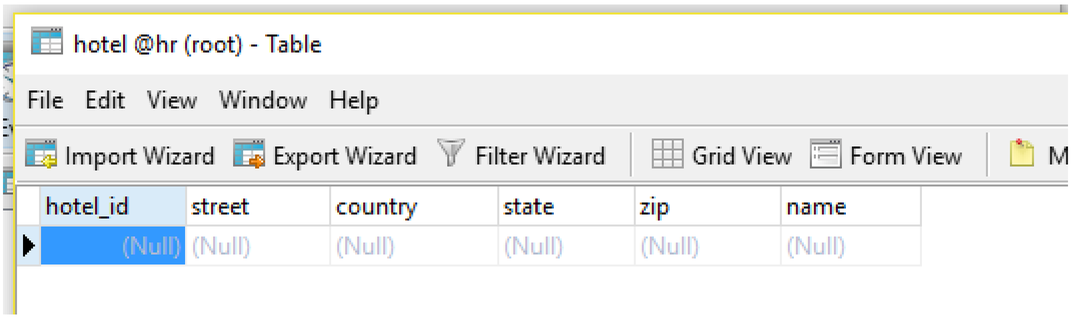
Name	Type	Length	Decimals	Allow Null	
hotel_id	int	11	0	□	🔑1
street	varchar	255	0	□	
country	varchar	255	0	□	
state	varchar	255	0	□	
zip	varchar	255	0	□	
name	varchar	255	0	□	

5. Click Foreign Keys to create foreign key, this is an example of room table

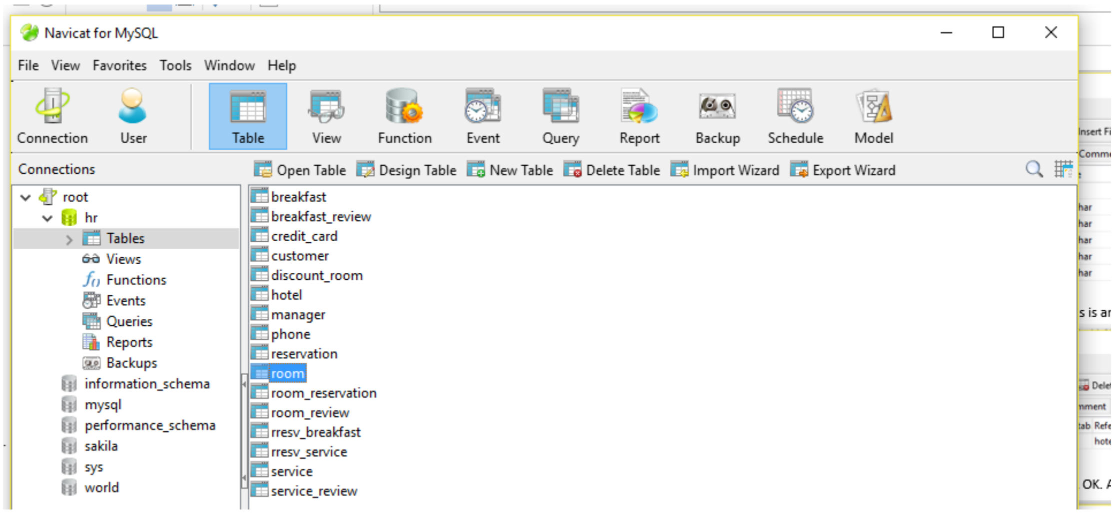

room @hr (root) - Table
File Edit Window Help
New Save Save As Add Foreign Key Delete Foreign Key
Fields Indexes Foreign Keys Triggers Options Comment SQL Preview

Name	Fields	Reference Datab	Reference Table	Reference Fields	On Delete	On Update
room_ibfk_1	hotel_id	hr	hotel	hotel_id	CASCADE	CASCADE

6. Click Save, and type the table’s name, click OK. A new table is created.

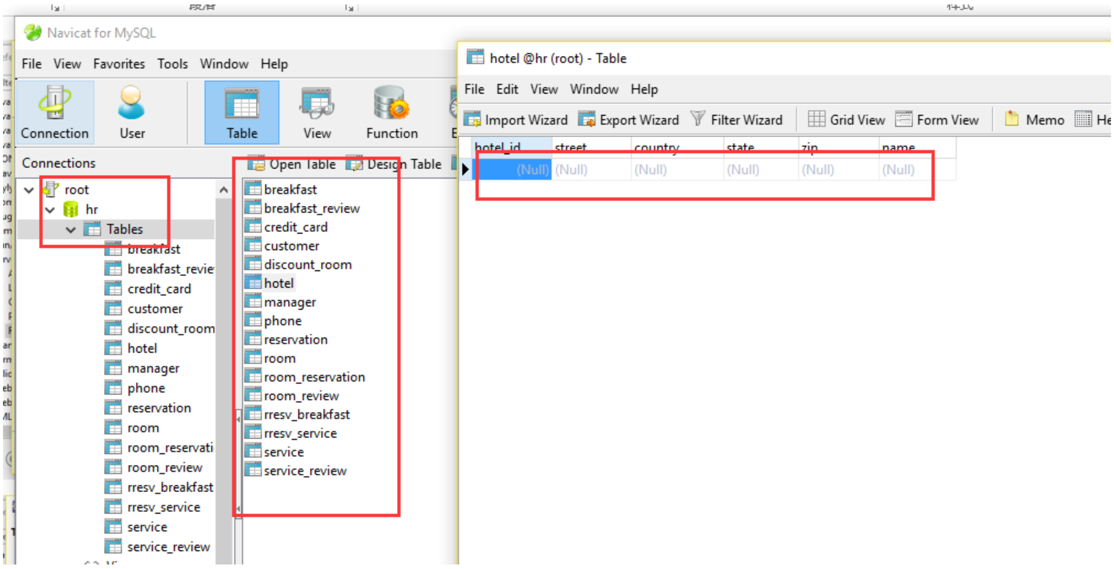


hotel_id	street	country	state	zip	name
(Null)	(Null)	(Null)	(Null)	(Null)	(Null)



- root
  - hr
    - Tables
    - Views
    - Functions
    - Events
    - Queries
    - Reports
    - Backups
  - information_schema
  - mysql
  - performance_schema
  - sakila
  - sys
  - world

breakfast
breakfast_review
credit_card
customer
discount_room
hotel
manager
phone
reservation
room
room_reservation
room_review
rresv_breakfast
rresv_service
service
service_review

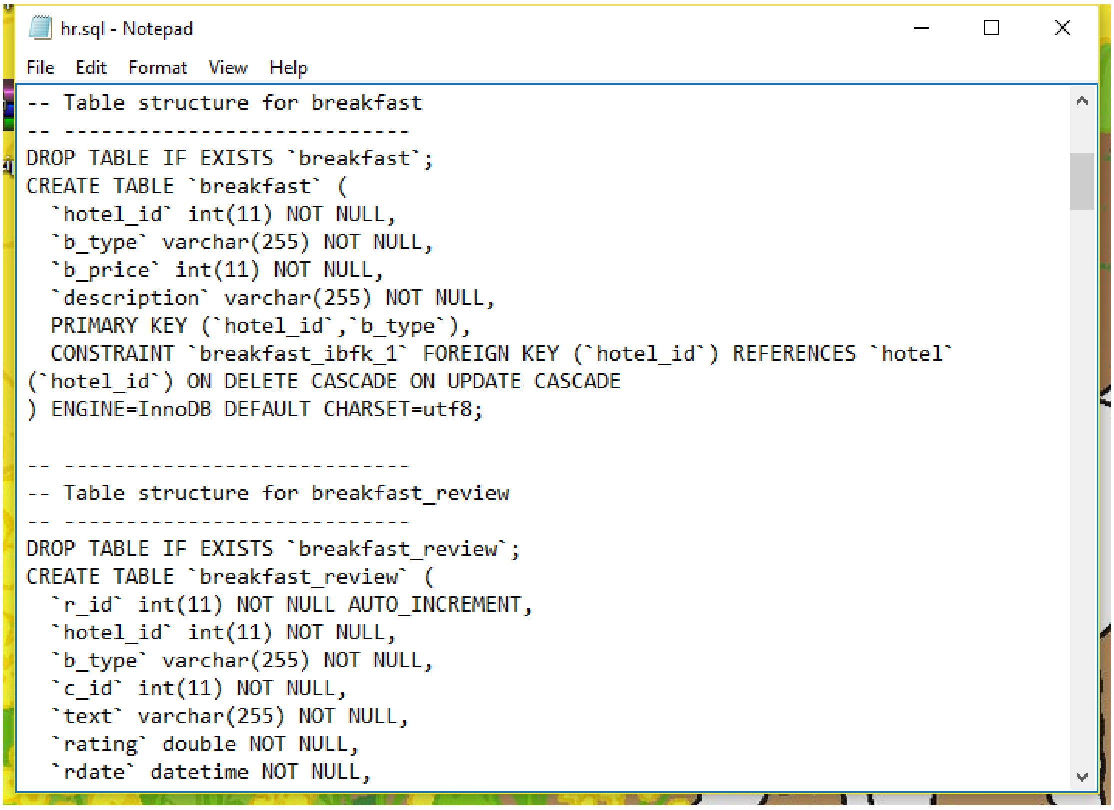


- root
  - hr
    - Tables
      - breakfast
      - breakfast_revie
      - credit_card
      - customer
      - discount_room
      - hotel
      - manager
      - phone
      - reservation
      - room
      - room_reservati
      - room_review
      - rresv_breakfast
      - rresv_service
      - service
      - service_review

breakfast
breakfast_review
credit_card
customer
discount_room
hotel
manager
phone
reservation
room
room_reservation
room_review
rresv_breakfast
rresv_service
service
service_review

hotel_id	street	country	state	zip	name
(Null)	(Null)	(Null)	(Null)	(Null)	(Null)

7. To generate a sql file which is include all create table SQL commands, right click hr -> Dump SQL File… -> Save. Here is partial of the table creation SQL commands.

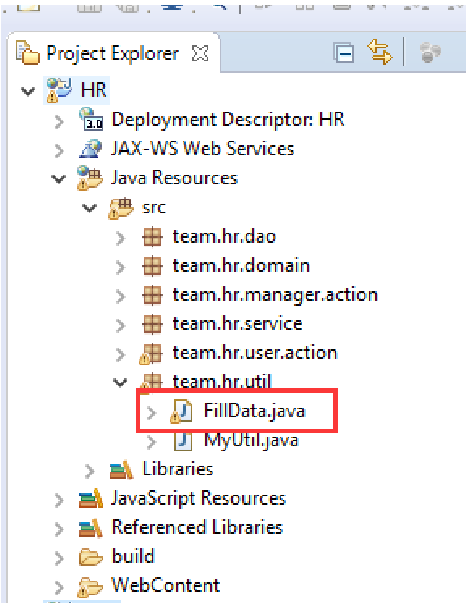


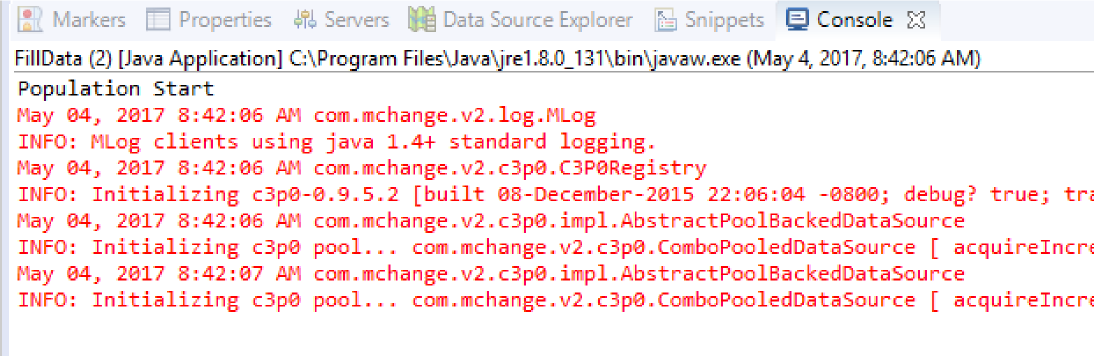
```
-- Table structure for breakfast
-- ----------------------------
DROP TABLE IF EXISTS `breakfast`;
CREATE TABLE `breakfast` (
  `hotel_id` int(11) NOT NULL,
  `b_type` varchar(255) NOT NULL,
  `b_price` int(11) NOT NULL,
  `description` varchar(255) NOT NULL,
  PRIMARY KEY (`hotel_id`,`b_type`),
  CONSTRAINT `breakfast_ibfk_1` FOREIGN KEY (`hotel_id`) REFERENCES `hotel`
(`hotel_id`) ON DELETE CASCADE ON UPDATE CASCADE
) ENGINE=InnoDB DEFAULT CHARSET=utf8;

-- ----------------------------
-- Table structure for breakfast_review
-- ----------------------------
DROP TABLE IF EXISTS `breakfast_review`;
CREATE TABLE `breakfast_review` (
  `r_id` int(11) NOT NULL AUTO_INCREMENT,
  `hotel_id` int(11) NOT NULL,
  `b_type` varchar(255) NOT NULL,
  `c_id` int(11) NOT NULL,
  `text` varchar(255) NOT NULL,
  `rating` double NOT NULL,
  `rdate` datetime NOT NULL,
```

Table Population

We use java program to populate all of our tables. <SQL Commands of Table Population (partial)> part of our report is giving partial SQL commands of table population. There are 200 hotels in the hotel chain.

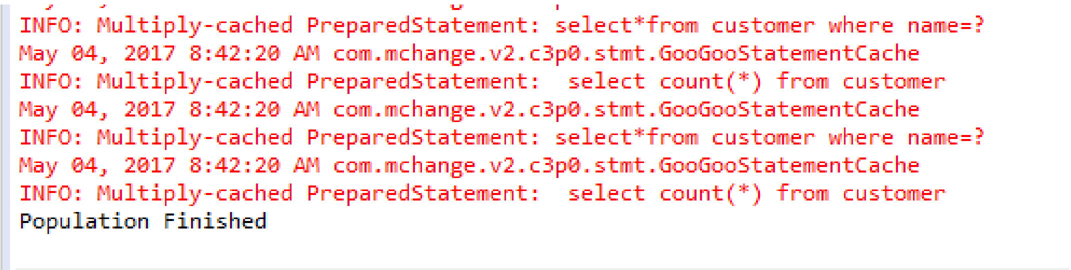
1. Just right click the FillData.java on Eclipse, the click run as -> java application. You will see a Population start message in the console panel:





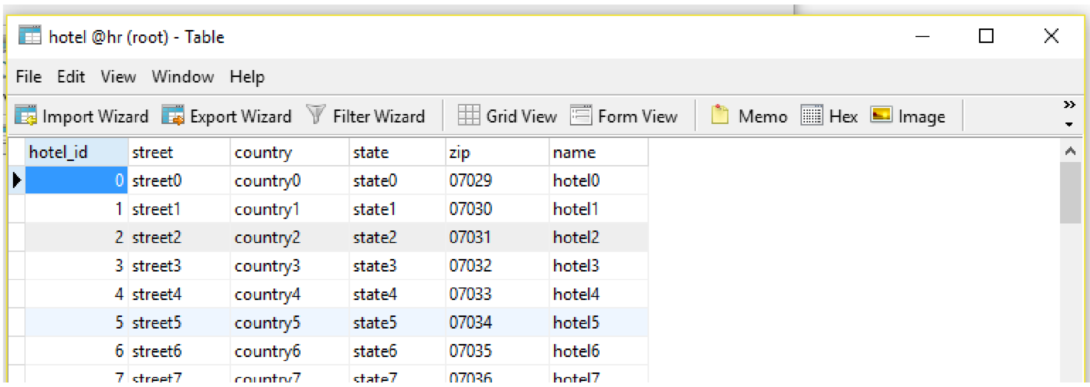
```
FillData (2) [Java Application] C:\Program Files\Java\jre1.8.0_131\bin\javaw.exe (May 4, 2017, 8:42:06 AM)
Population Start
May 04, 2017 8:42:06 AM com.mchange.v2.log.MLog
INFO: MLog clients using java 1.4+ standard logging.
May 04, 2017 8:42:06 AM com.mchange.v2.c3p0.C3P0Registry
INFO: Initializing c3p0-0.9.5.2 [built 08-December-2015 22:06:04 -0800; debug? true; tra
May 04, 2017 8:42:06 AM com.mchange.v2.c3p0.impl.AbstractPoolBackedDataSource
INFO: Initializing c3p0 pool... com.mchange.v2.c3p0.ComboPooledDataSource [ acquireIncre
May 04, 2017 8:42:07 AM com.mchange.v2.c3p0.impl.AbstractPoolBackedDataSource
INFO: Initializing c3p0 pool... com.mchange.v2.c3p0.ComboPooledDataSource [ acquireIncre
```

2. When a Population Finished message shows up, it means tables population is done.



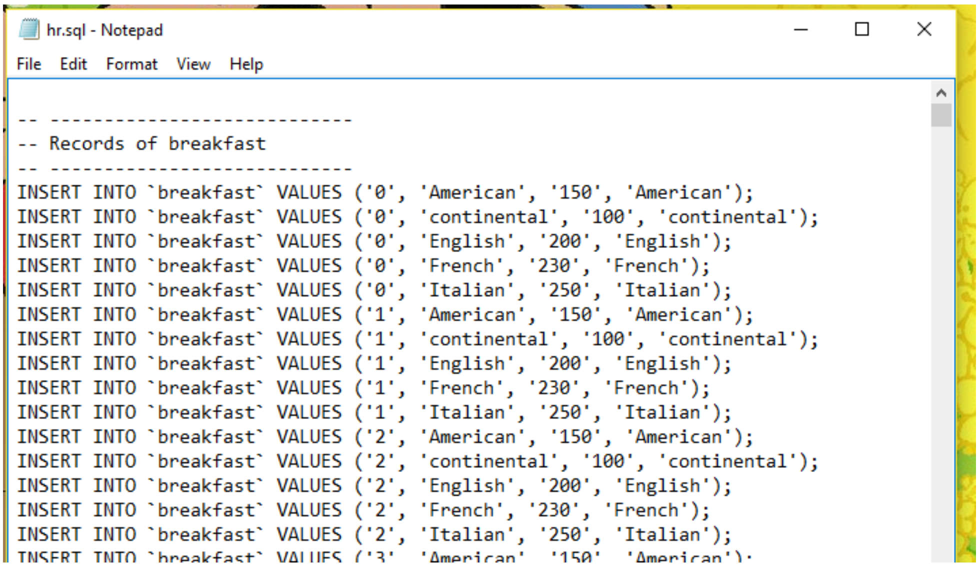
```
INFO: Multiply-cached PreparedStatement: select*from customer where name=?
May 04, 2017 8:42:20 AM com.mchange.v2.c3p0.stmt.GooGooStatementCache
INFO: Multiply-cached PreparedStatement:  select count(*) from customer
May 04, 2017 8:42:20 AM com.mchange.v2.c3p0.stmt.GooGooStatementCache
INFO: Multiply-cached PreparedStatement: select*from customer where name=?
May 04, 2017 8:42:20 AM com.mchange.v2.c3p0.stmt.GooGooStatementCache
INFO: Multiply-cached PreparedStatement:  select count(*) from customer
Population Finished
```

3. Then you will see all tables are full of data.



hotel_id	street	country	state	zip	name
0	street0	country0	state0	07029	hotel0
1	street1	country1	state1	07030	hotel1
2	street2	country2	state2	07031	hotel2
3	street3	country3	state3	07032	hotel3
4	street4	country4	state4	07033	hotel4
5	street5	country5	state5	07034	hotel5
6	street6	country6	state6	07035	hotel6
7	street7	country7	state7	07036	hotel7

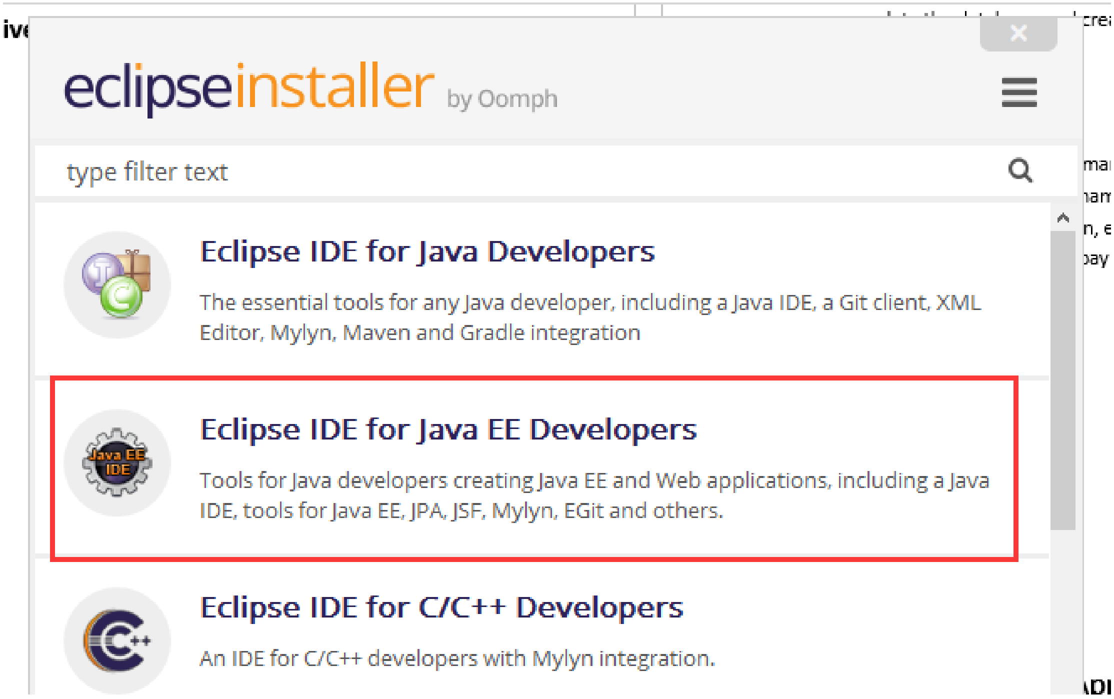
4. To generate a sql file which is include all SQL commands that populate the tables, after run the FillData.java in Eclipse, then right click hr -> Dump SQL File... -> Save in Navicat.



```
-- ----------------------------
-- Records of breakfast
-- ----------------------------
INSERT INTO `breakfast` VALUES ('0', 'American', '150', 'American');
INSERT INTO `breakfast` VALUES ('0', 'continental', '100', 'continental');
INSERT INTO `breakfast` VALUES ('0', 'English', '200', 'English');
INSERT INTO `breakfast` VALUES ('0', 'French', '230', 'French');
INSERT INTO `breakfast` VALUES ('0', 'Italian', '250', 'Italian');
INSERT INTO `breakfast` VALUES ('1', 'American', '150', 'American');
INSERT INTO `breakfast` VALUES ('1', 'continental', '100', 'continental');
INSERT INTO `breakfast` VALUES ('1', 'English', '200', 'English');
INSERT INTO `breakfast` VALUES ('1', 'French', '230', 'French');
INSERT INTO `breakfast` VALUES ('1', 'Italian', '250', 'Italian');
INSERT INTO `breakfast` VALUES ('2', 'American', '150', 'American');
INSERT INTO `breakfast` VALUES ('2', 'continental', '100', 'continental');
INSERT INTO `breakfast` VALUES ('2', 'English', '200', 'English');
INSERT INTO `breakfast` VALUES ('2', 'French', '230', 'French');
INSERT INTO `breakfast` VALUES ('2', 'Italian', '250', 'Italian');
INSERT INTO `breakfast` VALUES ('3', 'American', '150', 'American');
```

Run Program

1. First we should setup the environment of JDK, consider this is a very easy thing to be solve, we will not show the steps here. After done of the setup enviroment of JDK, install Eclipse IDE for Java EE Developers:



Eclipse IDE for Java Developers

The essential tools for any Java developer, including a Java IDE, a Git client, XML Editor, Mylyn, Maven and Gradle integration

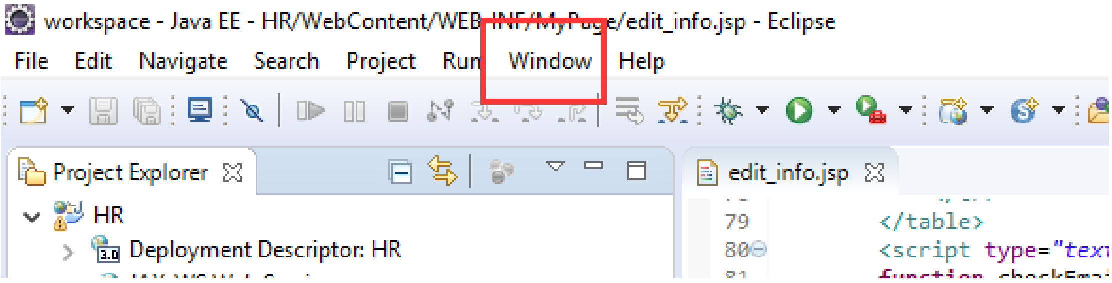
Eclipse IDE for Java EE Developers

Tools for Java developers creating Java EE and Web applications, including a Java IDE, tools for Java EE, JPA, JSF, Mylyn, EGit and others.

Eclipse IDE for C/C++ Developers

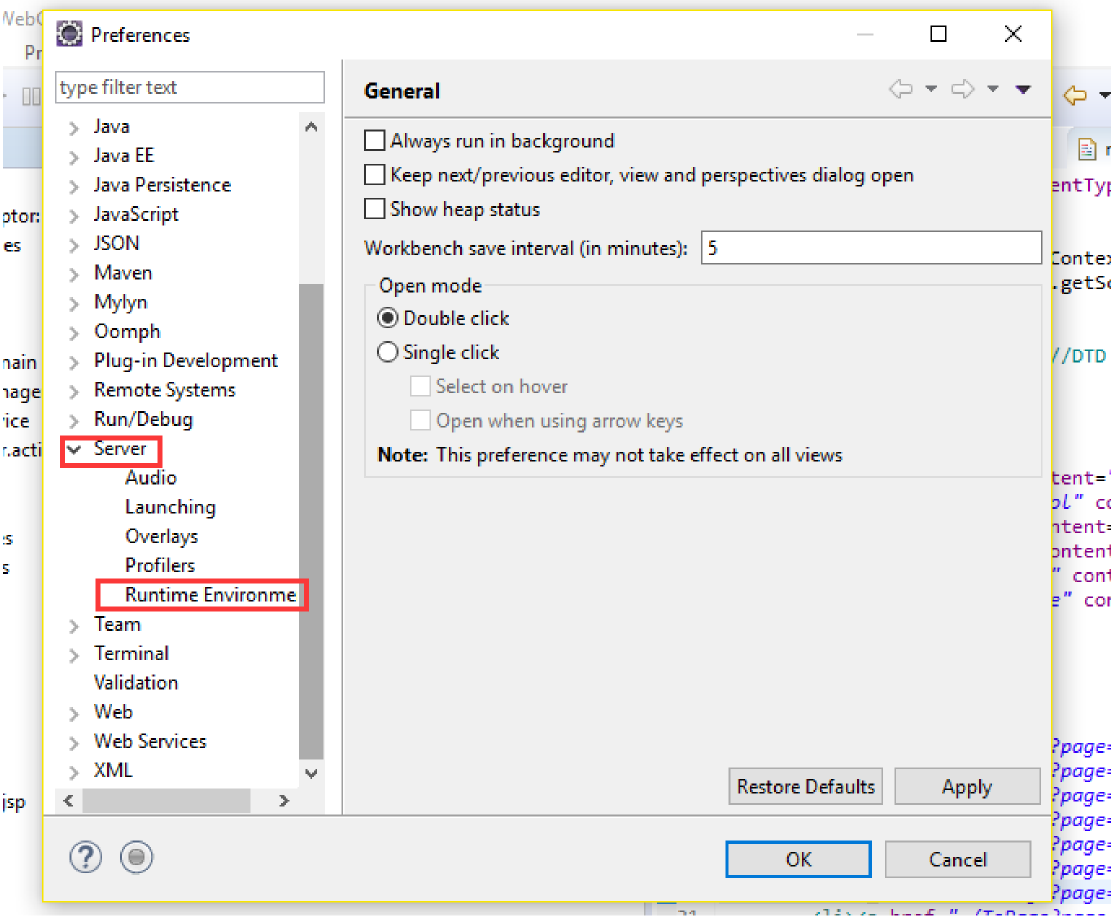
An IDE for C/C++ developers with Mylyn integration.

2. Add tomcat to Eclipse, our version of tomat is apache-tomcat-7.0.73:



workspace - Java EE - HR/WebContent/WEB-INF/MyPage/edit_info.jsp - Eclipse
File
Edit
Navigate
Search
Project
Run
Window
Help
Project Explorer
HR
Deployment Descriptor: HR
edit_info.jsp

```
        </table>
        <script type="text
```



□ Always run in background

□ Keep next/previous editor, view and perspectives dialog open

□ Show heap status

Workbench save interval (in minutes): 5

Open mode

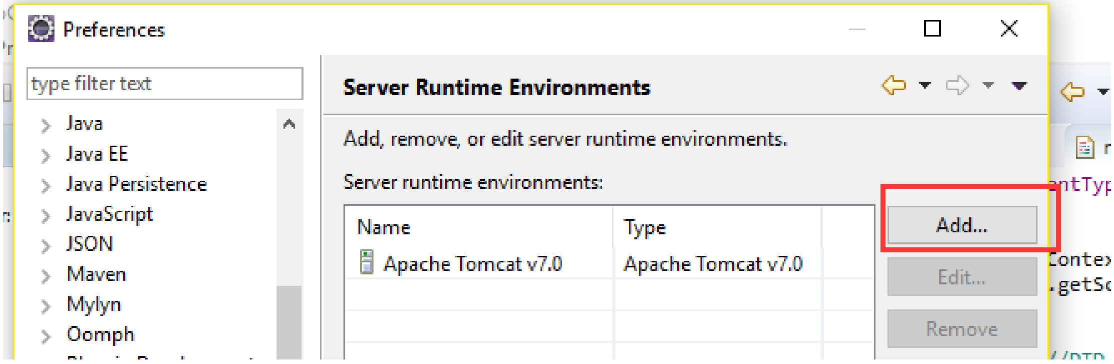
● Double click

○ Single click

□ Select on hover

□ Open when using arrow keys

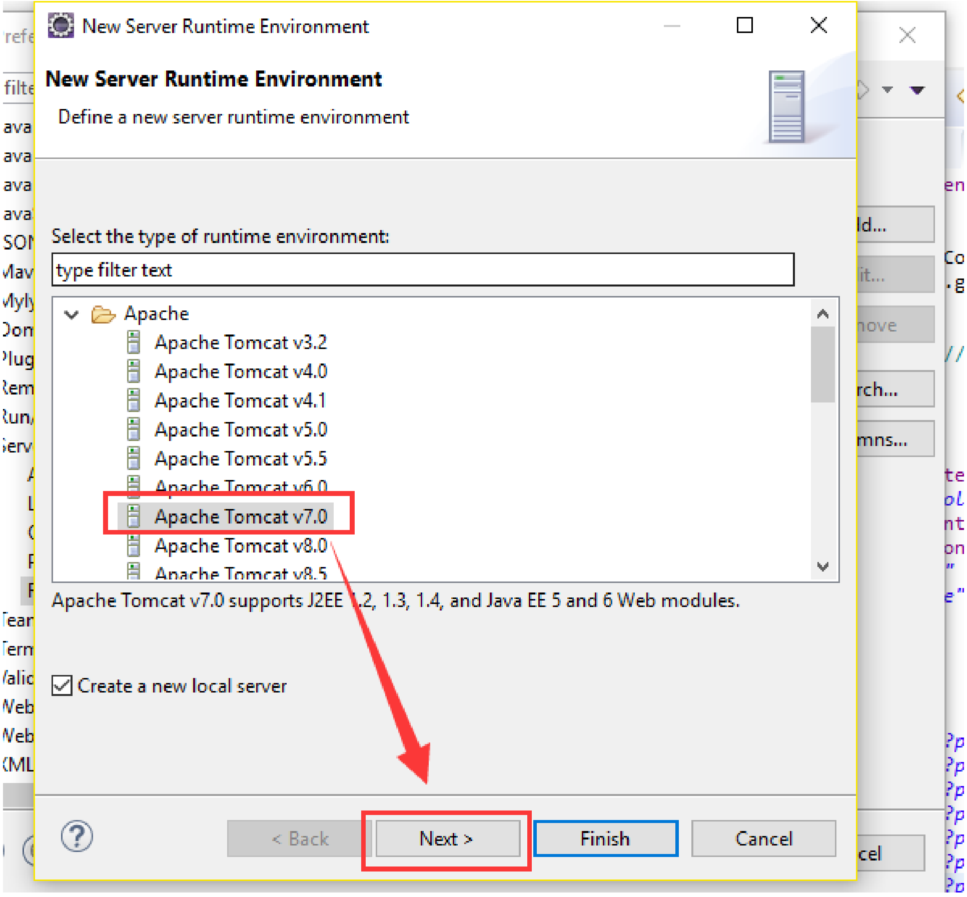
Note: This preference may not take effect on all views



Add, remove, or edit server runtime environments.

Server runtime environments:

Name	Type
Apache Tomcat v7.0	Apache Tomcat v7.0



New Server Runtime Environment

New Server Runtime Environment

Define a new server runtime environment

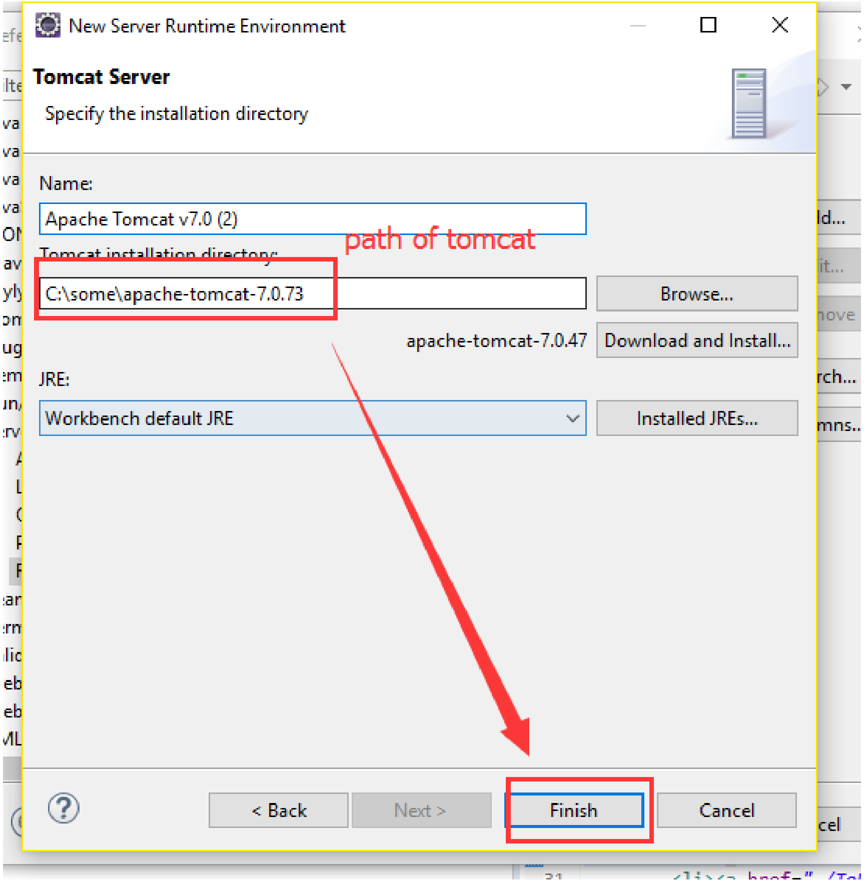
Select the type of runtime environment:

type filter text

- Apache
  - Apache Tomcat v3.2
  - Apache Tomcat v4.0
  - Apache Tomcat v4.1
  - Apache Tomcat v5.0
  - Apache Tomcat v5.5
  - Apache Tomcat v6.0
  - Apache Tomcat v7.0
  - Apache Tomcat v8.0
  - Apache Tomcat v8.5

Apache Tomcat v7.0 supports J2EE 1.2, 1.3, 1.4, and Java EE 5 and 6 Web modules.

☑ Create a new local server



New Server Runtime Environment

Tomcat Server

Specify the installation directory

Name:

Apache Tomcat v7.0 (2)

Tomcat installation directory:

C:\some\apache-tomcat-7.0.73

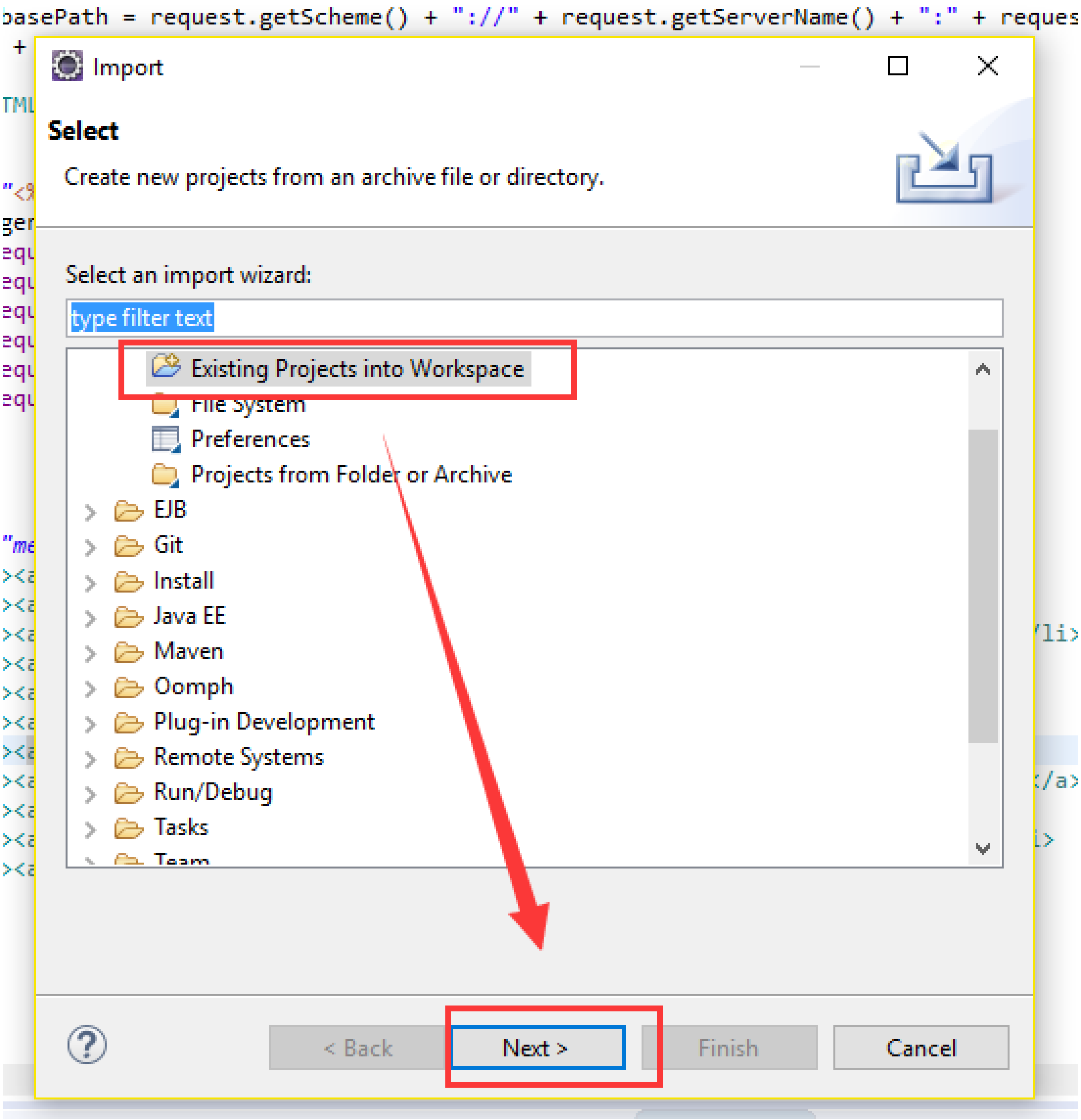
path of tomcat

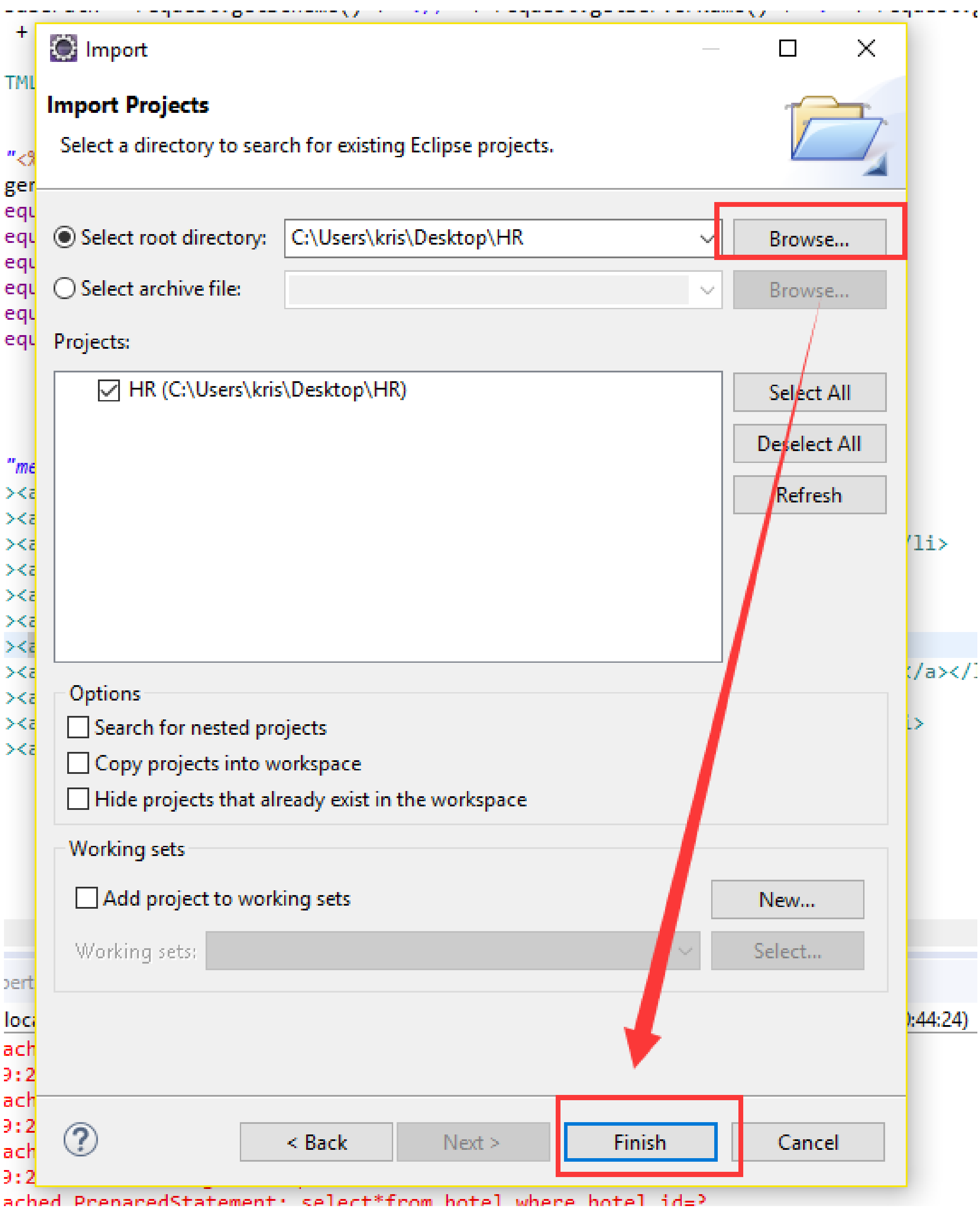
apache-tomcat-7.0.47

JRE:

Workbench default JRE

3. To run our source code, just simply import the HR file by click File -> Import, run the fillData.java by click Run As Java Application to populate the tables in database.





Import Projects

Select a directory to search for existing Eclipse projects.

● Select root directory: C:\Users\kris\Desktop\HR

○ Select archive file:

Projects:

☑ HR (C:\Users\kris\Desktop\HR)

Options

□ Search for nested projects

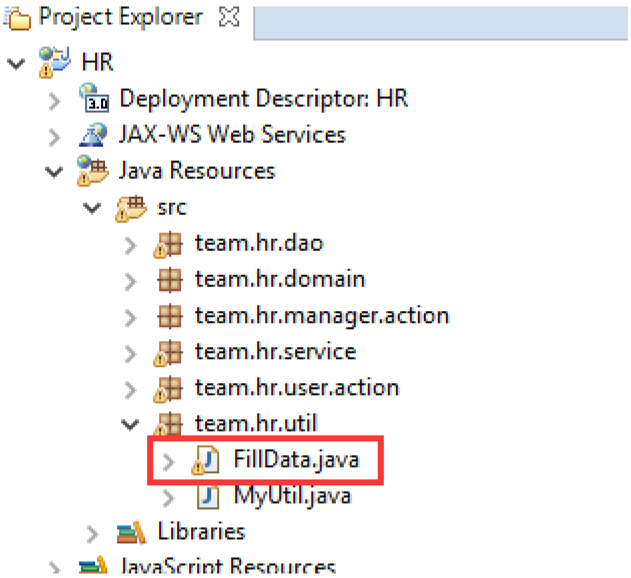
□ Copy projects into workspace

□ Hide projects that already exist in the workspace

Working sets

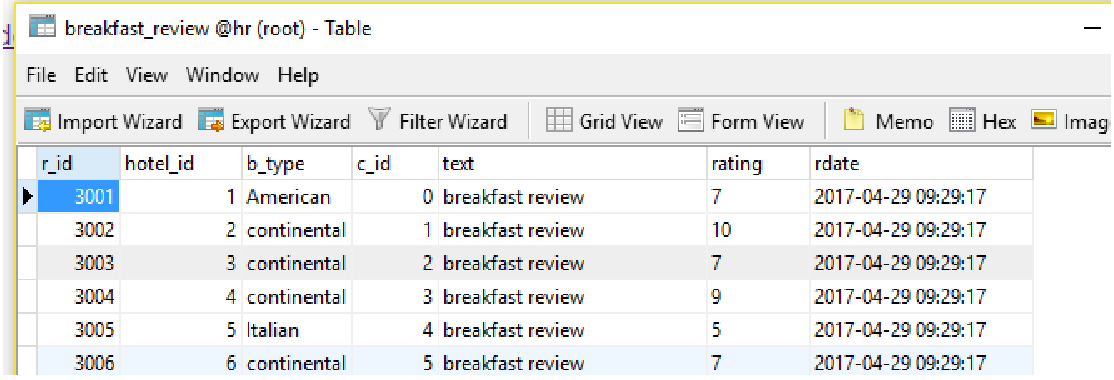
□ Add project to working sets

Working sets:



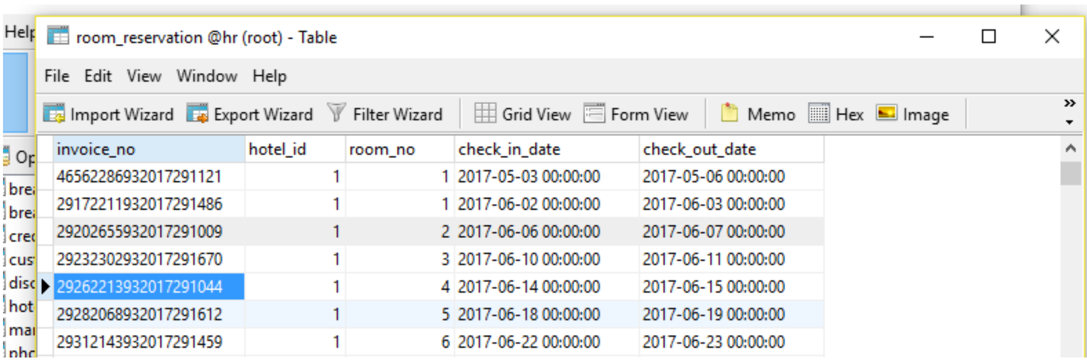
```
Project Explorer
v HR
  > Deployment Descriptor: HR
  > JAX-WS Web Services
  v Java Resources
    v src
      > team.hr.dao
      > team.hr.domain
      > team.hr.manager.action
      > team.hr.service
      > team.hr.user.action
      v team.hr.util
        > FillData.java
        > MyUtil.java
    > Libraries
  > JavaScript Resources
```

4. After populate all the tables, there are 200 hotels in the Hulton hotel chain, the following is part of the breakfast review table and part of the room reservation table:



breakfast_review @hr (root) - Table
File Edit View Window Help
Import Wizard Export Wizard Filter Wizard Grid View Form View Memo Hex Imag

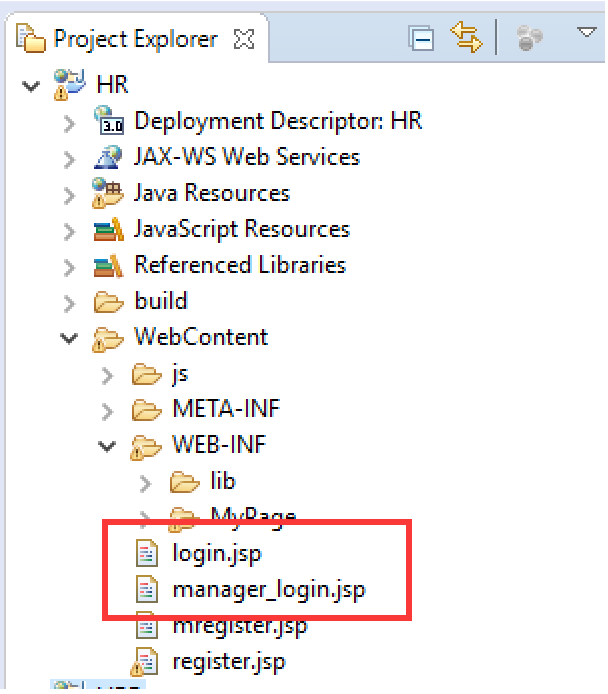
r_id	hotel_id	b_type	c_id	text	rating	rdate
3001	1	American	0	breakfast review	7	2017-04-29 09:29:17
3002	2	continental	1	breakfast review	10	2017-04-29 09:29:17
3003	3	continental	2	breakfast review	7	2017-04-29 09:29:17
3004	4	continental	3	breakfast review	9	2017-04-29 09:29:17
3005	5	Italian	4	breakfast review	5	2017-04-29 09:29:17
3006	6	continental	5	breakfast review	7	2017-04-29 09:29:17



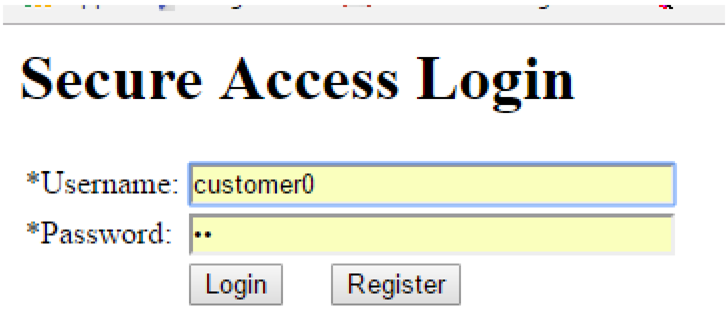
Help
room_reservation @hr (root) - Table
File Edit View Window Help
Import Wizard Export Wizard Filter Wizard Grid View Form View Memo Hex Image
Op
brea
brea
cred
cus
disc
hot
mai
phc

invoice_no	hotel_id	room_no	check_in_date	check_out_date
4656228693201729 1121	1	1	2017-05-03 00:00:00	2017-05-06 00:00:00
2917221193201729 1486	1	1	2017-06-02 00:00:00	2017-06-03 00:00:00
2920265593201729 1009	1	2	2017-06-06 00:00:00	2017-06-07 00:00:00
2923230293201729 1670	1	3	2017-06-10 00:00:00	2017-06-11 00:00:00
2926221393201729 1044	1	4	2017-06-14 00:00:00	2017-06-15 00:00:00
2928206893201729 1612	1	5	2017-06-18 00:00:00	2017-06-19 00:00:00
2931214393201729 1459	1	6	2017-06-22 00:00:00	2017-06-23 00:00:00

5. For customers, run login.jps by click Run on Server. For managers, run manager_login.jsp by click Run on Server.



6. After run login.jps, a website will pop up, Type customer0 as username and p0 as password to do a test, then click login in:

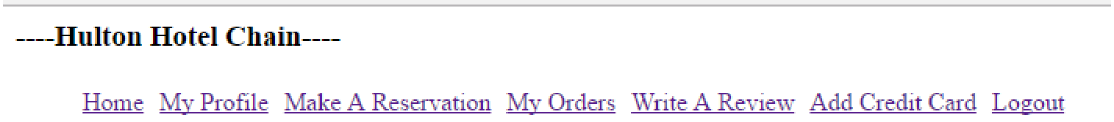


Secure Access Login

*Username: customer0

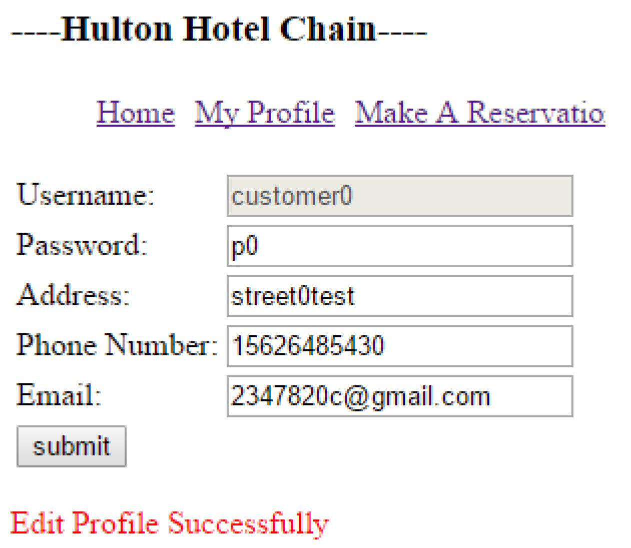
*Password: ••

7. Then you will see the web page for customers. On the top-left you can see a discount information for one of a hotel:



----Hulton Hotel Chain----

8. Click My Profile to change user’s information, after click submit button, you will see the new information is already in the database.

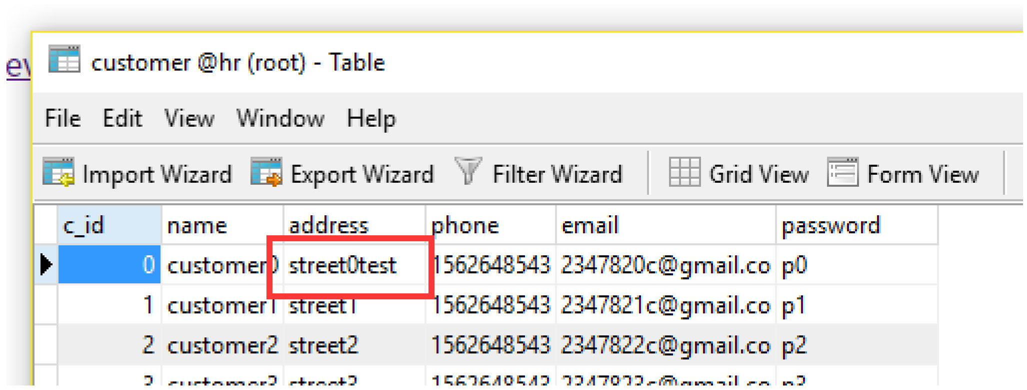


----Hulton Hotel Chain----

Home My Profile Make A Reservatio

Username:	customer0
Password:	p0
Address:	street0test
Phone Number:	15626485430
Email:	2347820c@gmail.com

Edit Profile Successfully

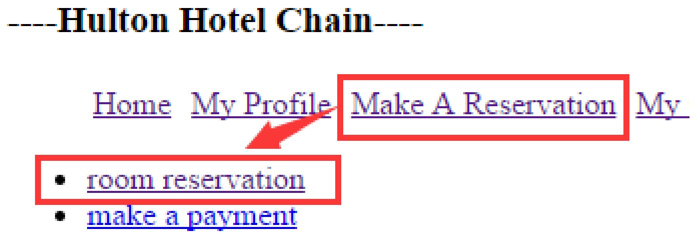


customer @hr (root) - Table

File Edit View Window Help
Import Wizard Export Wizard Filter Wizard Grid View Form View

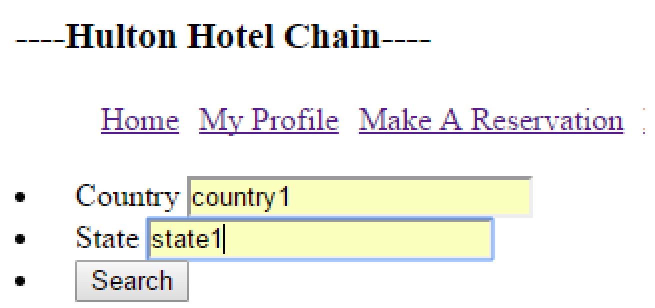
c_id	name	address	phone	email	password
0	customer0	street0test	1562648543	2347820c@gmail.co	p0
1	customer1	street1	1562648543	2347821c@gmail.co	p1
2	customer2	street2	1562648543	2347822c@gmail.co	p2
3	customer3	street3	1562648543	2347823c@gmail.co	p3

9. To make a reservation, just simply click the Make A Reservation button, choose the country and state to find a hotel, use coutry1 and state1 to do a test.



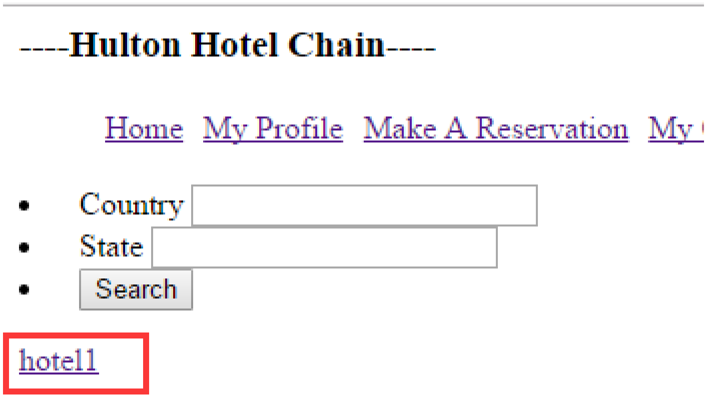
----Hulton Hotel Chain----

- room reservation
- make a payment



----Hulton Hotel Chain----

- Country country1
- State state1
- Search



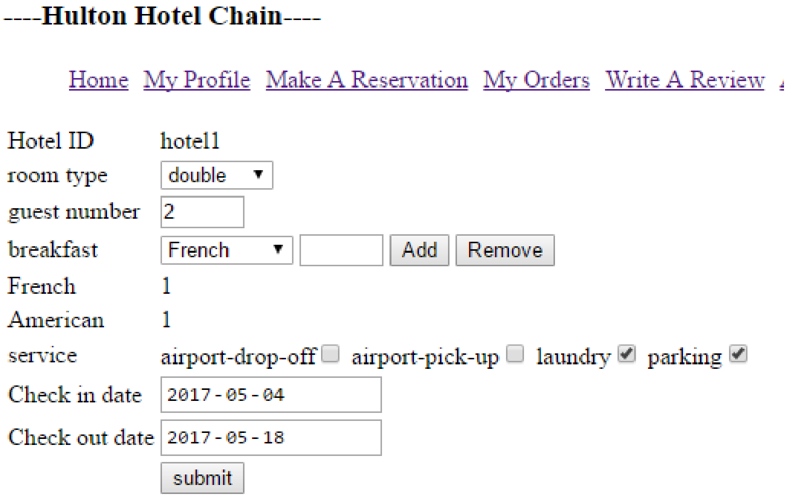
----Hulton Hotel Chain----

Home My Profile Make A Reservation My

- Country
- State
- Search

hotel1

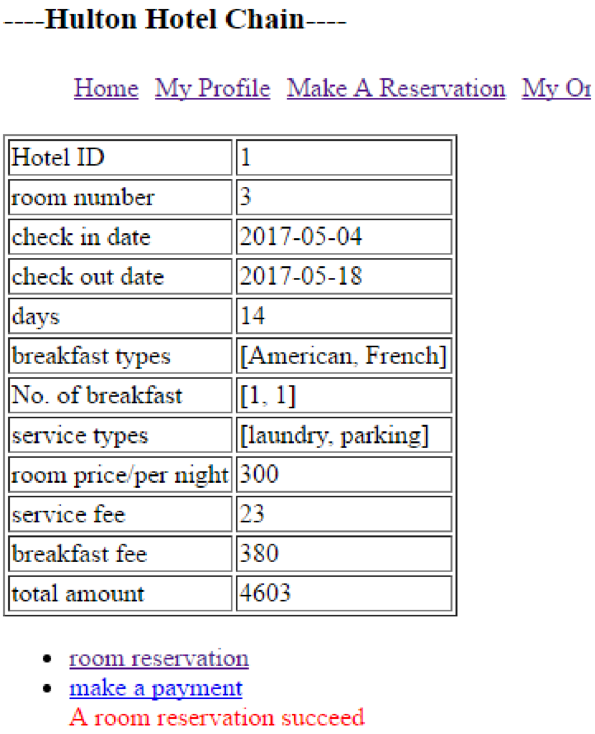
10. Reserve a room:



----Hulton Hotel Chain----

Home My Profile Make A Reservation My Orders Write A Review

Hotel ID	hotel1
room type	double ▼
guest number	2
breakfast	French ▼ Add Remove
French	1
American	1
service	airport-drop-off □ airport-pick-up □ laundry ☑ parking ☑
Check in date	2017-05-04
Check out date	2017-05-18
	submit



----Hulton Hotel Chain----

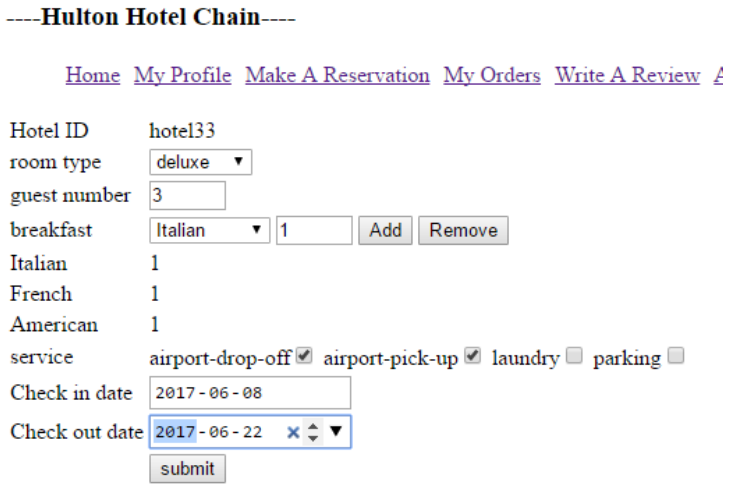
Home My Profile Make A Reservation My Or

Hotel ID	1
room number	3
check in date	2017-05-04
check out date	2017-05-18
days	14
breakfast types	[American, French]
No. of breakfast	[1, 1]
service types	[laundry, parking]
room price/per night	300
service fee	23
breakfast fee	380
total amount	4603

- room reservation
- make a payment

A room reservation succeed

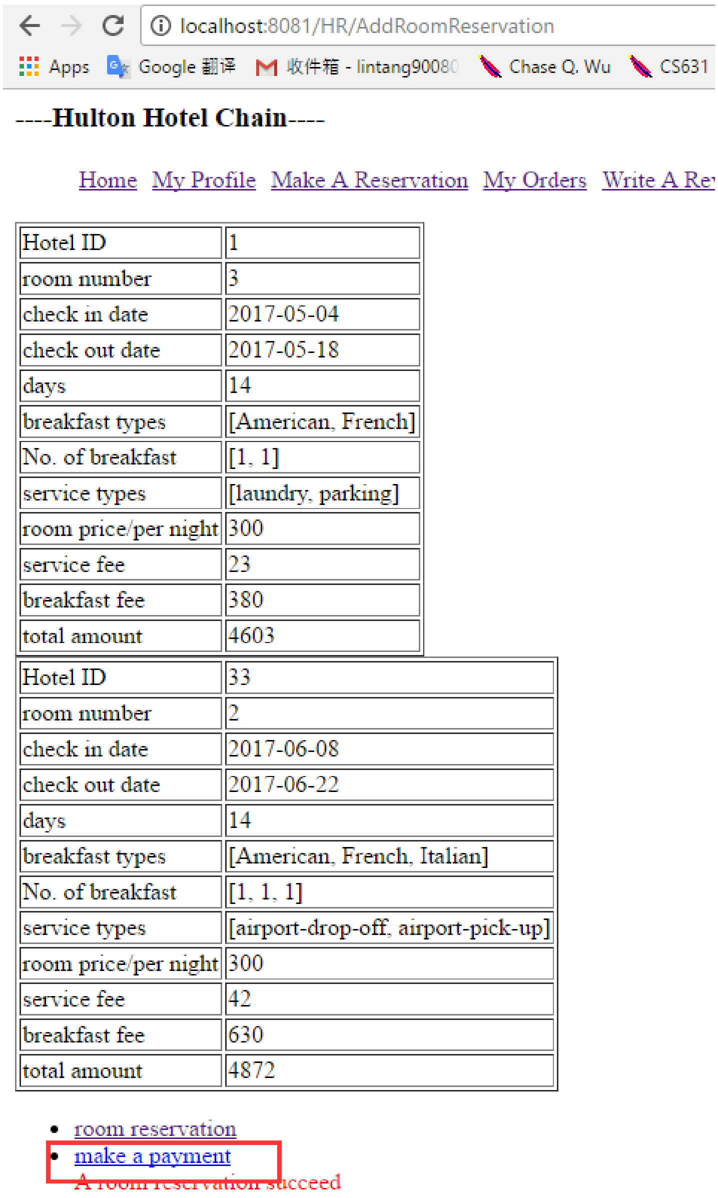
11. Reserve another room, also click room reservation:



----Hulton Hotel Chain----

Home My Profile Make A Reservation My Orders Write A Review A

Hotel ID	hotel33
room type	deluxe ▾
guest number	3
breakfast	Italian ▾ 1 Add Remove
Italian	1
French	1
American	1
service	airport-drop-off ☑ airport-pick-up ☑ laundry □ parking □
Check in date	2017-06-08
Check out date	2017-06-22



localhost:8081/HR/AddRoomReservation
Apps Google 翻译 收件箱 - lintang90080 Chase Q. Wu CS631

----Hulton Hotel Chain----

Home My Profile Make A Reservation My Orders Write A Re

Hotel ID	1
room number	3
check in date	2017-05-04
check out date	2017-05-18
days	14
breakfast types	[American, French]
No. of breakfast	[1, 1]
service types	[laundry, parking]
room price/per night	300
service fee	23
breakfast fee	380
total amount	4603

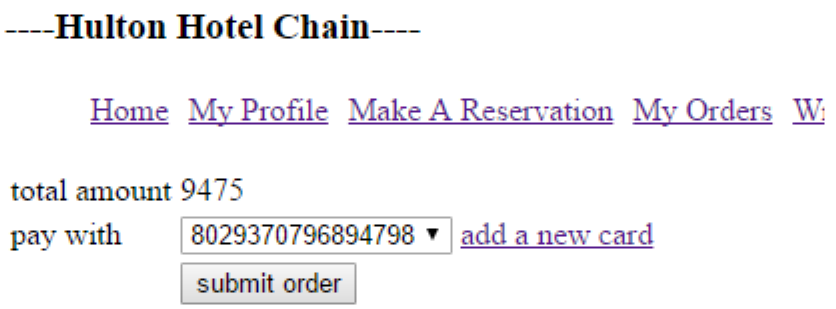
Hotel ID	33
room number	2
check in date	2017-06-08
check out date	2017-06-22
days	14
breakfast types	[American, French, Italian]
No. of breakfast	[1, 1, 1]
service types	[airport-drop-off, airport-pick-up]
room price/per night	300
service fee	42
breakfast fee	630
total amount	4872

- room reservation
- make a payment

A room reservation succeed

12. To pay the reservation, customer just need to click the make a payment button. Or if she

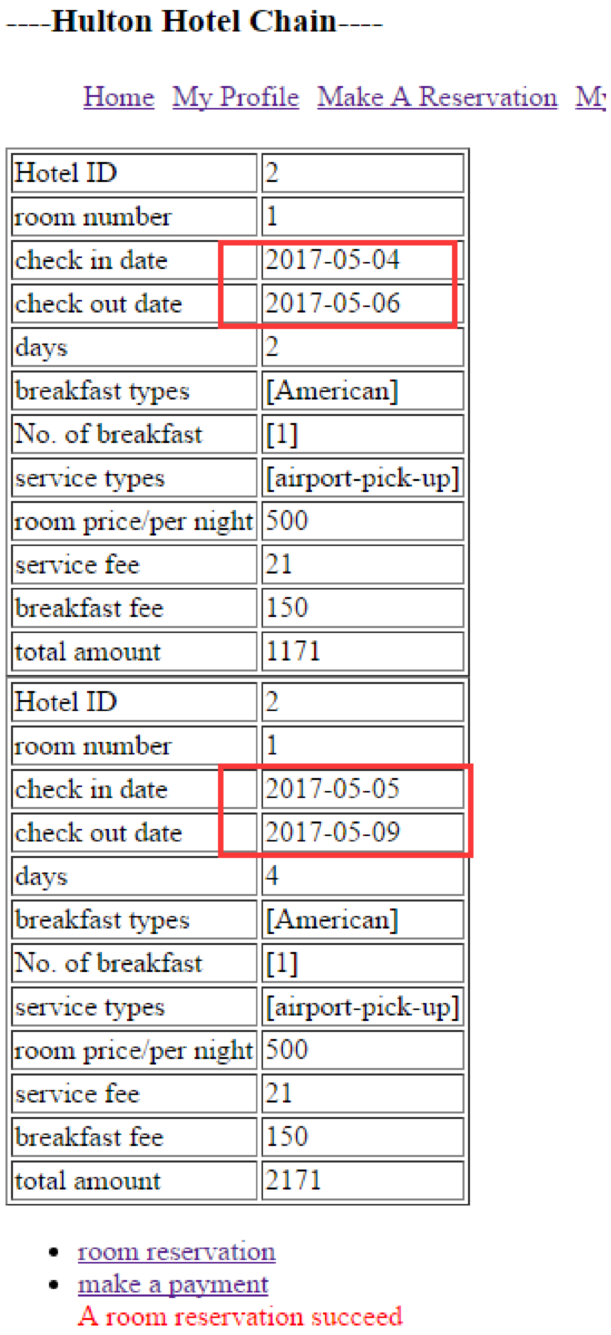
wants to add another card to pay the bill, she just need to click the Add a new card button.



total amount 9475

pay with 8029370796894798

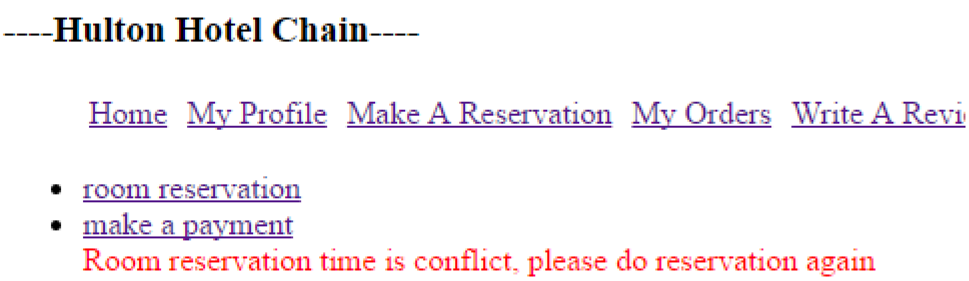
13. If there are same room's reservation date is overlape, it will give a error message, if the date is no conflict, then it will let the customer to pay the bill.



Hotel ID	2
room number	1
check in date	2017-05-04
check out date	2017-05-06
days	2
breakfast types	[American]
No. of breakfast	[1]
service types	[airport-pick-up]
room price/per night	500
service fee	21
breakfast fee	150
total amount	1171

Hotel ID	2
room number	1
check in date	2017-05-05
check out date	2017-05-09
days	4
breakfast types	[American]
No. of breakfast	[1]
service types	[airport-pick-up]
room price/per night	500
service fee	21
breakfast fee	150
total amount	2171

A room reservation succeed

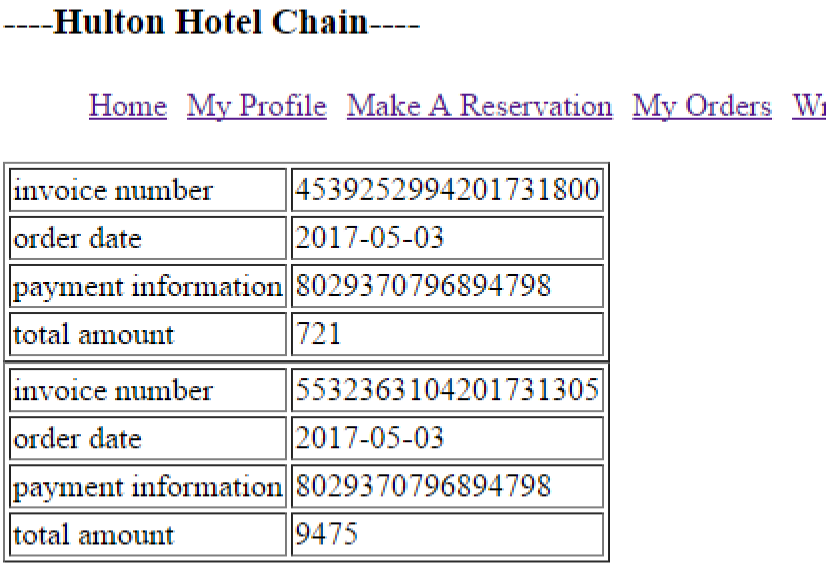


----Hulton Hotel Chain----

- room reservation
- make a payment

Room reservation time is conflict, please do reservation again

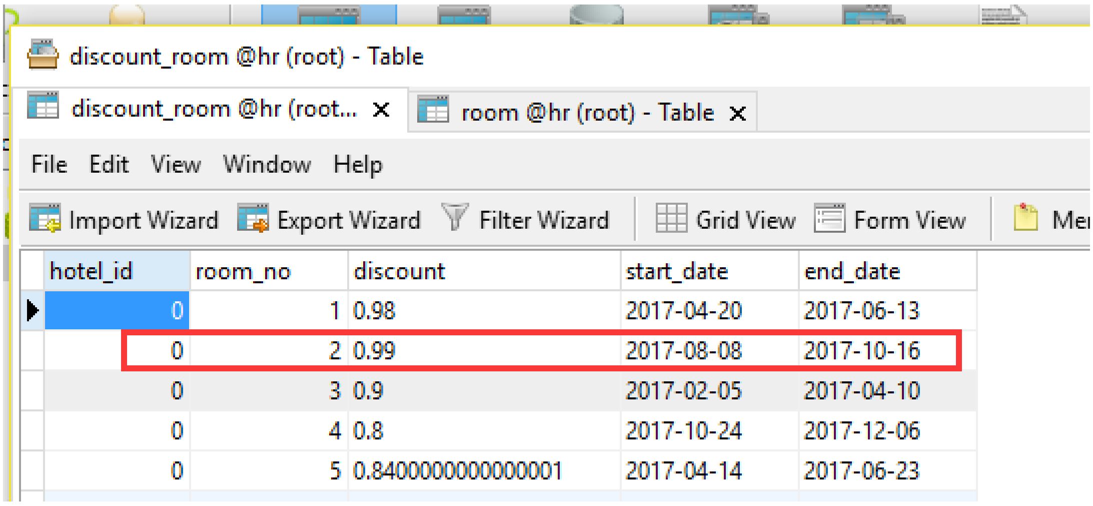
14. By clicking My Orders, customer can see the orders he made:



----Hulton Hotel Chain----

invoice number	4539252994201731800
order date	2017-05-03
payment information	8029370796894798
total amount	721
invoice number	5532363104201731305
order date	2017-05-03
payment information	8029370796894798
total amount	9475

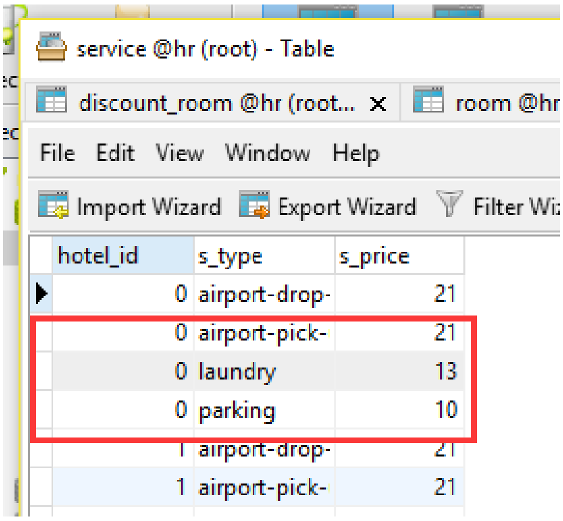
15. If the room check in and check out date is during the discount period, discounted prices for the room will be taken into account. Breakfast fee is the total fee of all the breakfast orders, service fee also too. For example: room 2 of hotel0's discount is 0.99 during 2017-08-08 to 2017-10-16, the amount is: (4 * 600) + (39 * 600 * 0.99) + (150 + 200) + (21 + 13 + 10) = 25960



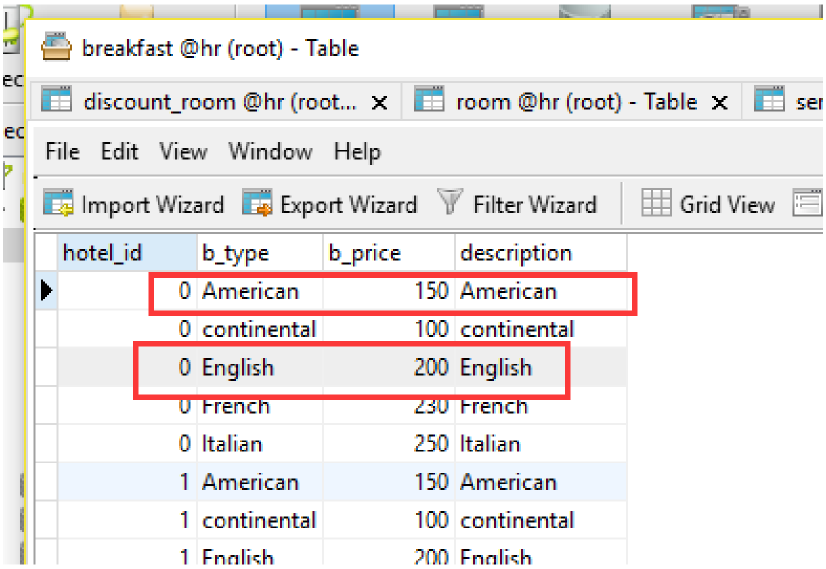
discount_room @hr (root) - Table

discount_room @hr (root... ×
room @hr (root) - Table ×
File Edit View Window Help
Import Wizard
Export Wizard
Filter Wizard
Grid View
Form View
Me

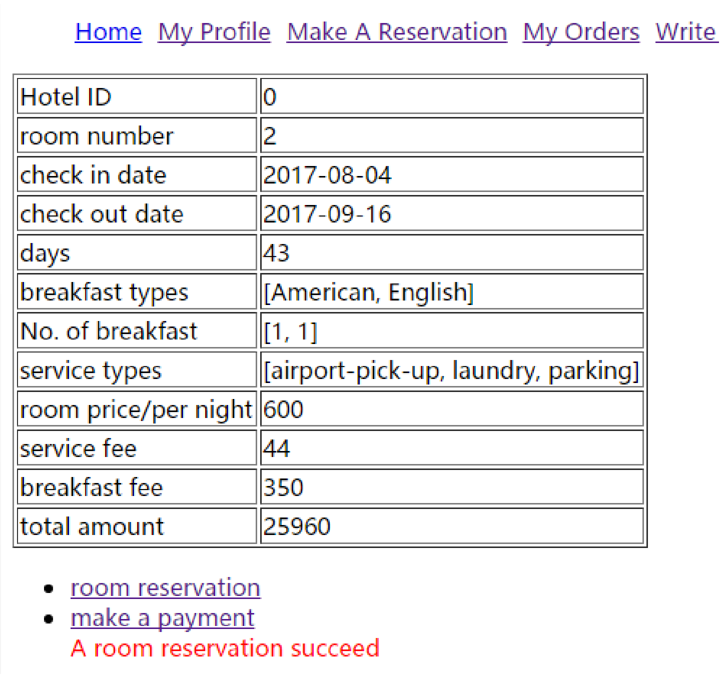
hotel_id	room_no	discount	start_date	end_date
0	1	0.98	2017-04-20	2017-06-13
0	2	0.99	2017-08-08	2017-10-16
0	3	0.9	2017-02-05	2017-04-10
0	4	0.8	2017-10-24	2017-12-06
0	5	0.8400000000000001	2017-04-14	2017-06-23


service @hr (root) - Table
discount_room @hr (root... ✕
room @hr
File Edit View Window Help
Import Wizard
Export Wizard
Filter Wi

hotel_id	s_type	s_price
0	airport-drop-	21
0	airport-pick-	21
0	laundry	13
0	parking	10
1	airport-drop-	21
1	airport-pick-	21


breakfast @hr (root) - Table
discount_room @hr (root... ✕
room @hr (root) - Table ✕
ser
File Edit View Window Help
Import Wizard
Export Wizard
Filter Wizard
Grid View

hotel_id	b_type	b_price	description
0	American	150	American
0	continental	100	continental
0	English	200	English
0	French	230	French
0	Italian	250	Italian
1	American	150	American
1	continental	100	continental
1	English	200	English



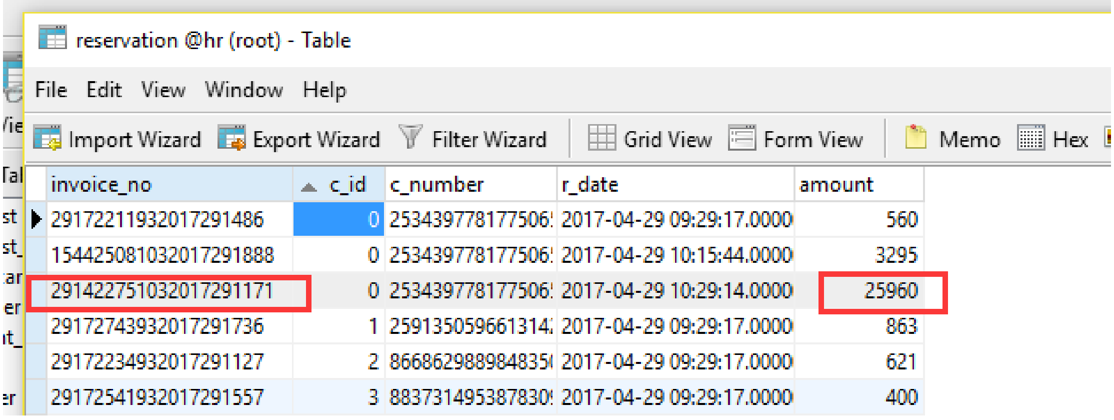
Home My Profile Make A Reservation My Orders Write

Hotel ID	0
room number	2
check in date	2017-08-04
check out date	2017-09-16
days	43
breakfast types	[American, English]
No. of breakfast	[1, 1]
service types	[airport-pick-up, laundry, parking]
room price/per night	600
service fee	44
breakfast fee	350
total amount	25960

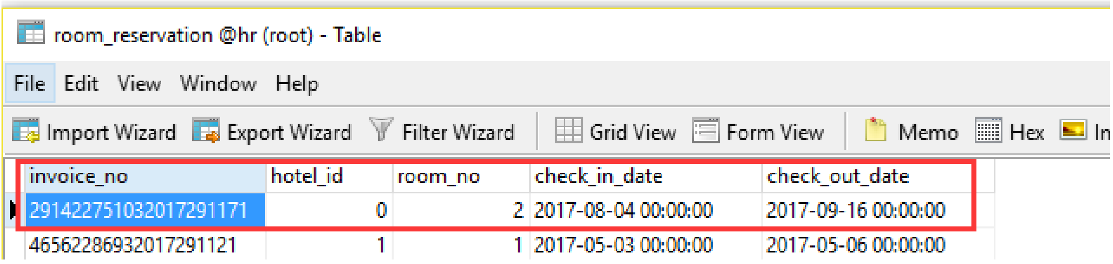
- room reservation
- make a payment

A room reservation succeed

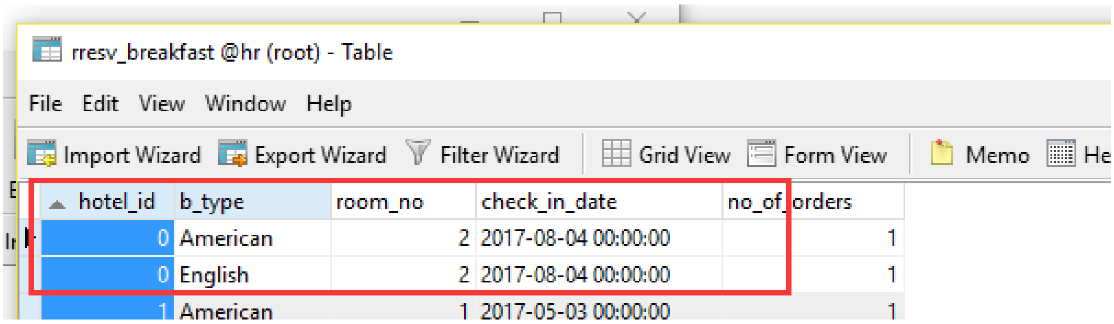
16. And you can find the reservation information in the database:


reservation @hr (root) - Table
File Edit View Window Help
Import Wizard Export Wizard Filter Wizard Grid View Form View Memo Hex

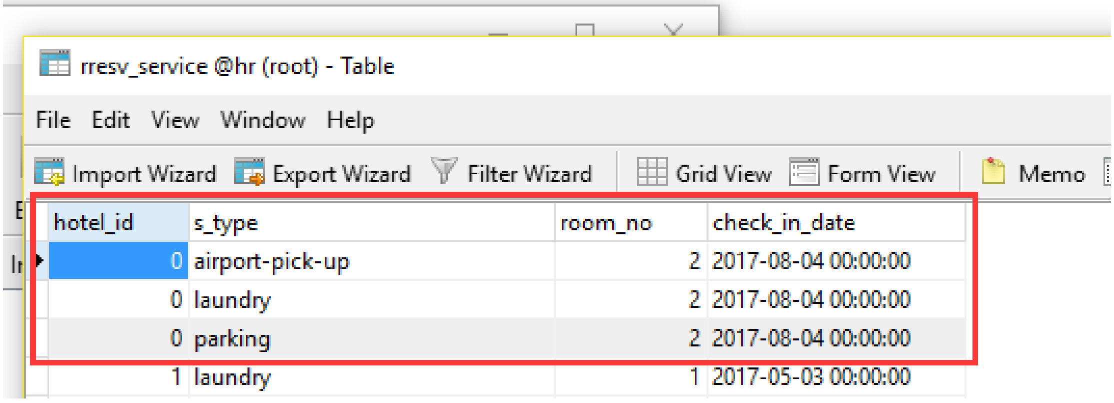
invoice_no	c_id	c_number	r_date	amount
29172211932017291486	0	253439778177506	2017-04-29 09:29:17.0000	560
15442508103201729188 8	0	253439778177506	2017-04-29 10:15:44.0000	3295
29142275103201729117 1	0	253439778177506	2017-04-29 10:29:14.0000	25960
29172743932017291736	1	259135059661314	2017-04-29 09:29:17.0000	863
29172234932017291127	2	866862988984835	2017-04-29 09:29:17.0000	621
29172541932017291557	3	883731495387830	2017-04-29 09:29:17.0000	400


room_reservation @hr (root) - Table
File Edit View Window Help
Import Wizard Export Wizard Filter Wizard Grid View Form View Memo Hex In

invoice_no	hotel_id	room_no	check_in_date	check_out_date
29142275103201729117 1	0	2	2017-08-04 00:00:00	2017-09-16 00:00:00
46562286932017291121	1	1	2017-05-03 00:00:00	2017-05-06 00:00:00

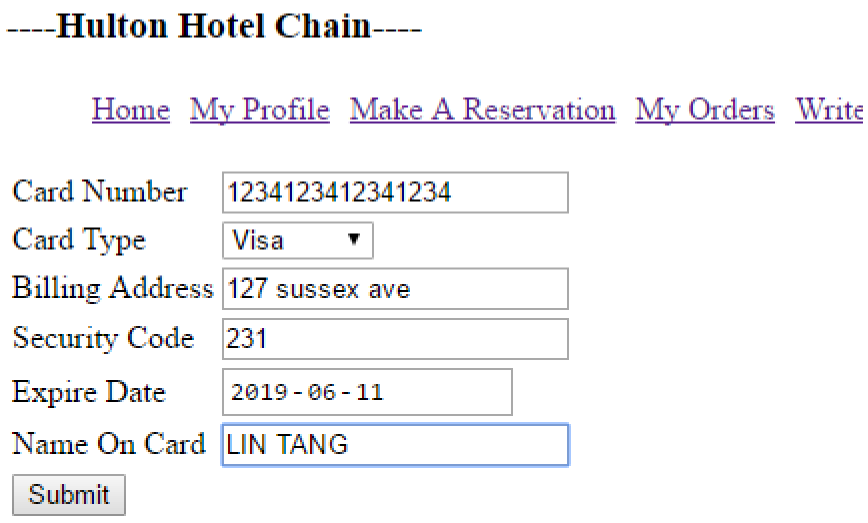

rresv_breakfast @hr (root) - Table
File Edit View Window Help
Import Wizard Export Wizard Filter Wizard Grid View Form View Memo He

hotel_id	b_type	room_no	check_in_date	no_of_orders
0	American	2	2017-08-04 00:00:00	1
0	English	2	2017-08-04 00:00:00	1
1	American	1	2017-05-03 00:00:00	1


rresv_service @hr (root) - Table
File Edit View Window Help
Import Wizard Export Wizard Filter Wizard Grid View Form View Memo

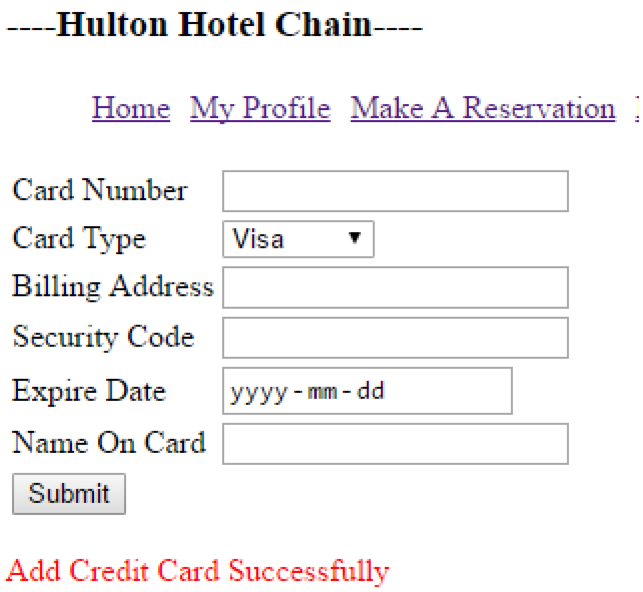
hotel_id	s_type	room_no	check_in_date
0	airport-pick-up	2	2017-08-04 00:00:00
0	laundry	2	2017-08-04 00:00:00
0	parking	2	2017-08-04 00:00:00
1	laundry	1	2017-05-03 00:00:00

17. By click Add Credit Card, customer can add any card for the future payment.



----Hulton Hotel Chain----

Card Number	1234123412341234
Card Type	Visa ▼
Billing Address	127 sussex ave
Security Code	231
Expire Date	2019-06-11
Name On Card	LIN TANG

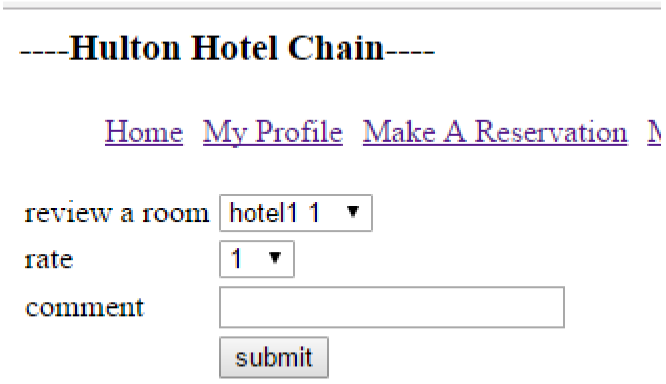


----Hulton Hotel Chain----

Card Number	
Card Type	Visa ▼
Billing Address	
Security Code	
Expire Date	yyyy-mm-dd
Name On Card	

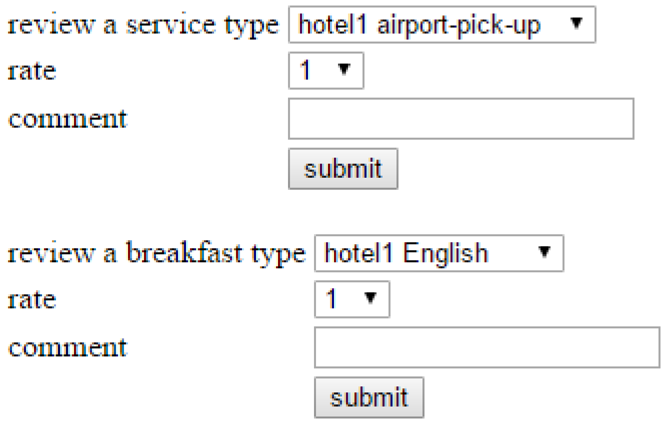
Add Credit Card Successfully

18. To write a review, just click Write A Review, after finish the review, click the submit button to submit your review



----Hulton Hotel Chain----

review a room	hotel1 1 ▼
rate	1 ▼
comment	



review a service type: hotel1 airport-pick-up

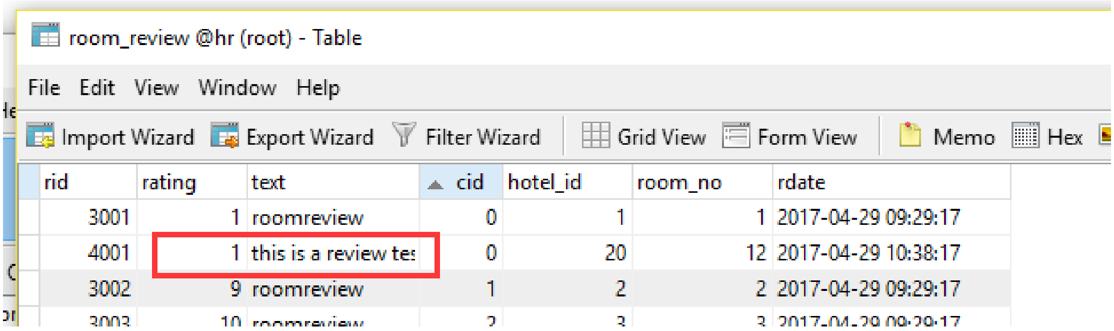
rate: 1

comment:

review a breakfast type: hotel1 English

rate: 1

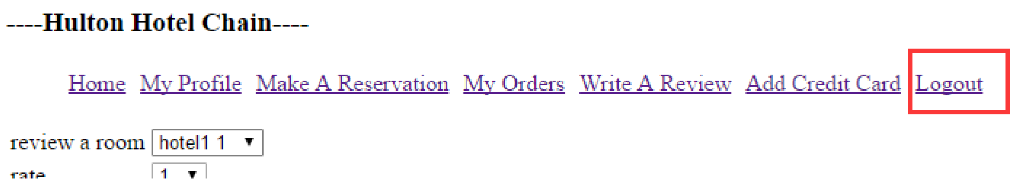
comment:



room_review @hr (root) - Table

rid	rating	text	cid	hotel_id	room_no	rdate
3001	1	roomreview	0	1	1	2017-04-29 09:29:17
4001	1	this is a review tes	0	20	12	2017-04-29 10:38:17
3002	9	roomreview	1	2	2	2017-04-29 09:29:17
3003	10	roomreview	2	3	3	2017-04-29 09:29:17

19. Click Logout to log out the website:

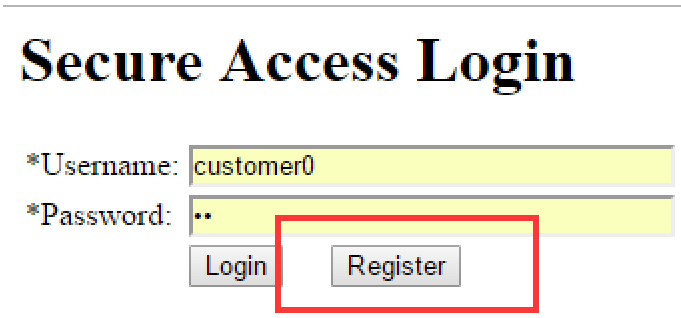


----Hulton Hotel Chain----

review a room: hotel1 1

rate: 1

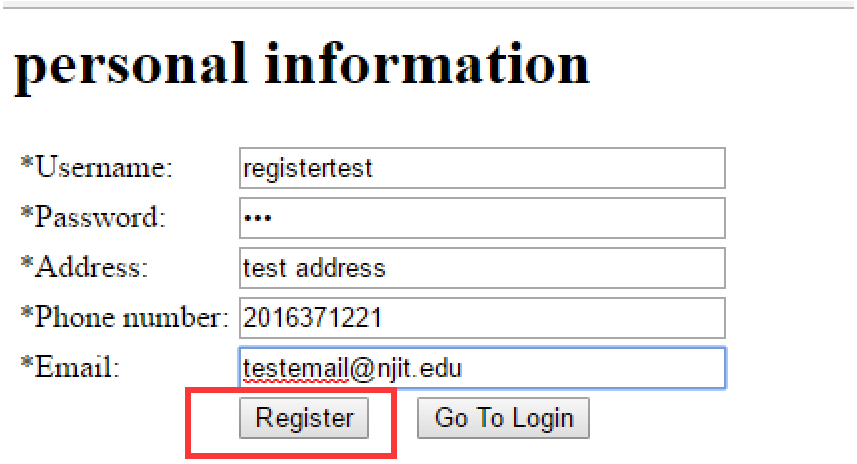
20. Click register to create a new account, click signup to submit your account creation



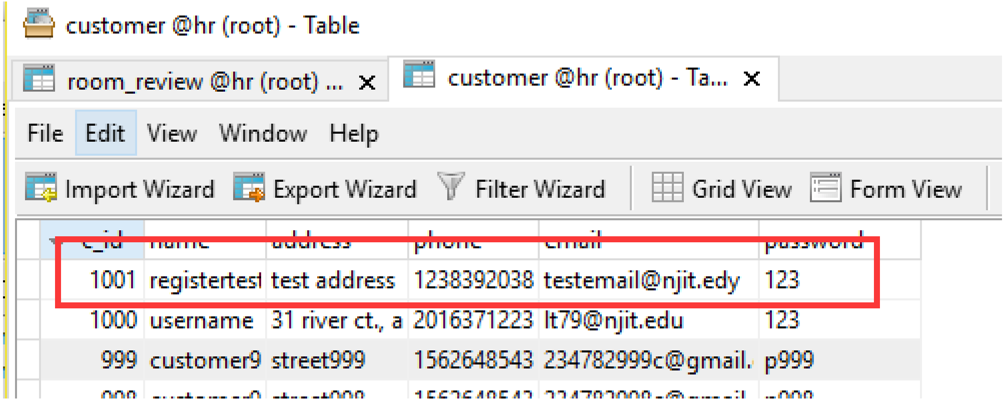
Secure Access Login

*Username: customer0

*Password: ••



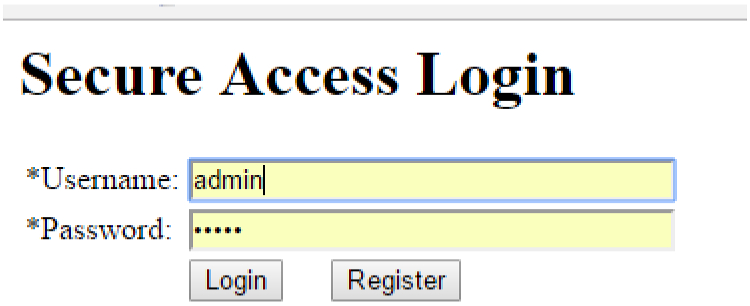
*Username:	registertest
*Password:	•••
*Address:	test address
*Phone number:	2016371221
*Email:	testemail@njit.edu



customer @hr (root) - Table
room_review @hr (root) ...
customer @hr (root) - Ta...
File
Edit
View
Window
Help
Import Wizard
Export Wizard
Filter Wizard
Grid View
Form View

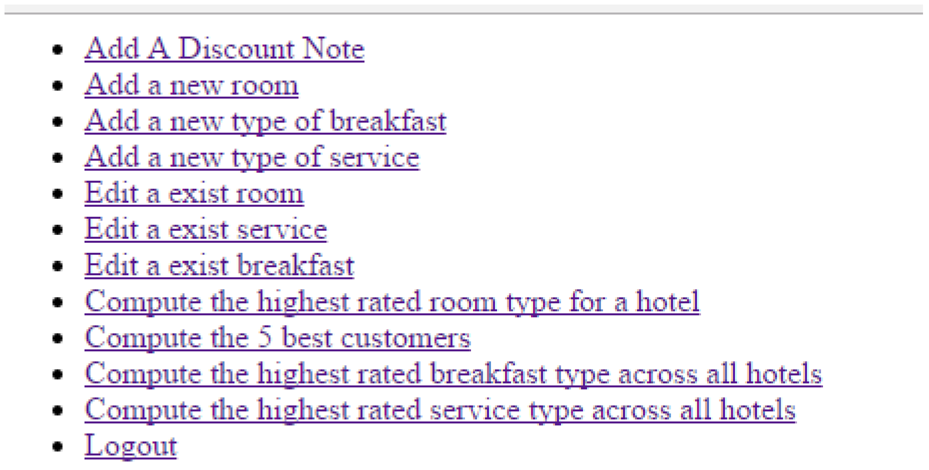
c_id	name	address	phone	email	password
1001	registertesl	test address	1238392038	testemail@njit.edy	123
1000	username	31 river ct., a	2016371223	lt79@njit.edu	123
999	customer9	street999	1562648543	234782999c@gmail.	p999
998	customer8	street998	1562648542	234782998c@gmail.	p998

21. For the managers page, run the manager_login.jsp. We create a default account: username: admin, password: admin for a manager account when doing the table population.



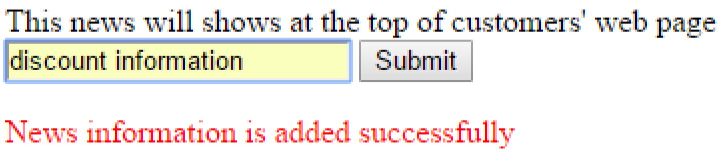
*Username:	admin
*Password:	•••••

22. User interface of manager:



- Add A Discount Note
- Add a new room
- Add a new type of breakfast
- Add a new type of service
- Edit a exist room
- Edit a exist service
- Edit a exist breakfast
- Compute the highest rated room type for a hotel
- Compute the 5 best customers
- Compute the highest rated breakfast type across all hotels
- Compute the highest rated service type across all hotels
- Logout

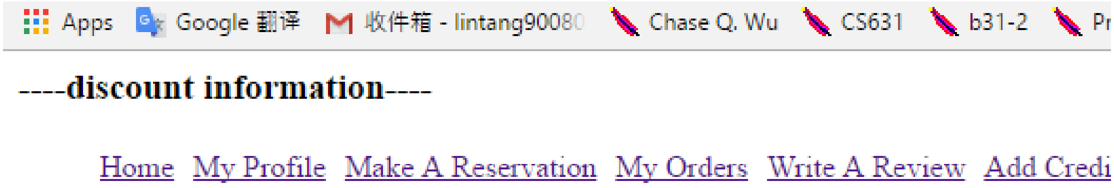
23. Add A Discount Note:



This news will shows at the top of customers' web page

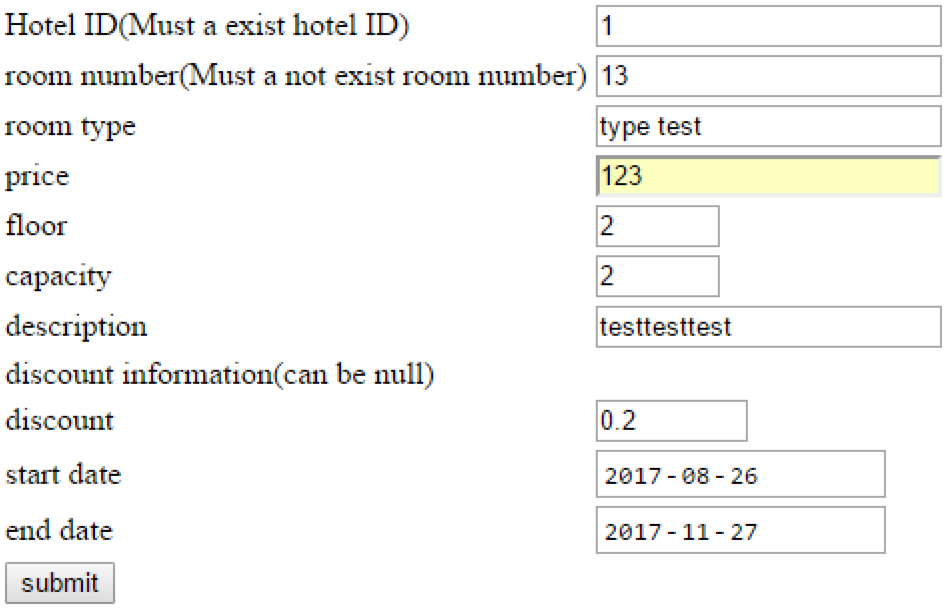
discount information

News information is added successfully

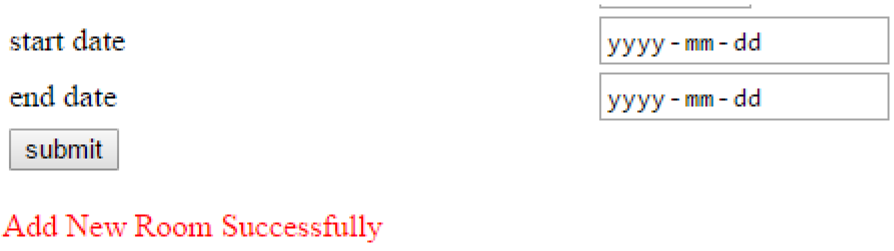


----discount information----

24. Add a new room:

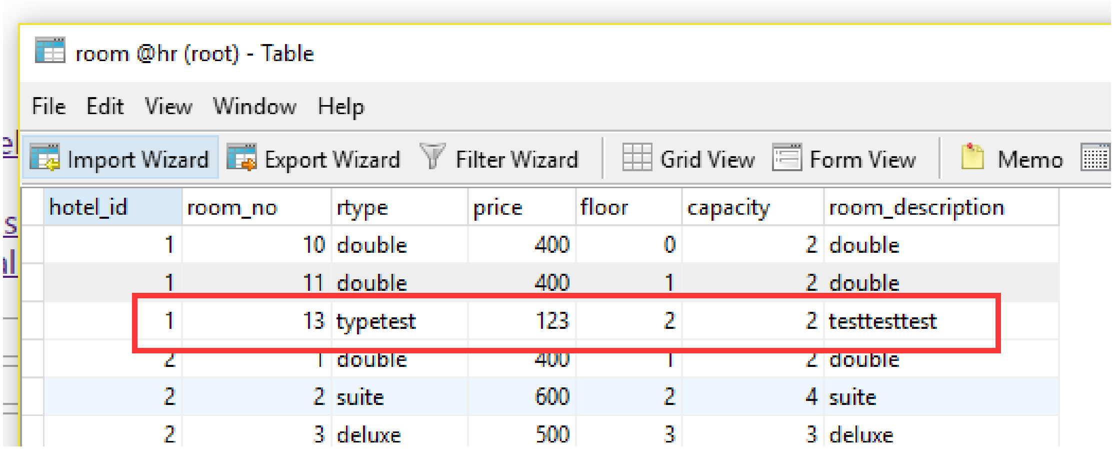


Hotel ID(Must a exist hotel ID)	1
room number(Must a not exist room number)	13
room type	type test
price	123
floor	2
capacity	2
description	testtesttest
discount information(can be null)	
discount	0.2
start date	2017-08-26
end date	2017-11-27



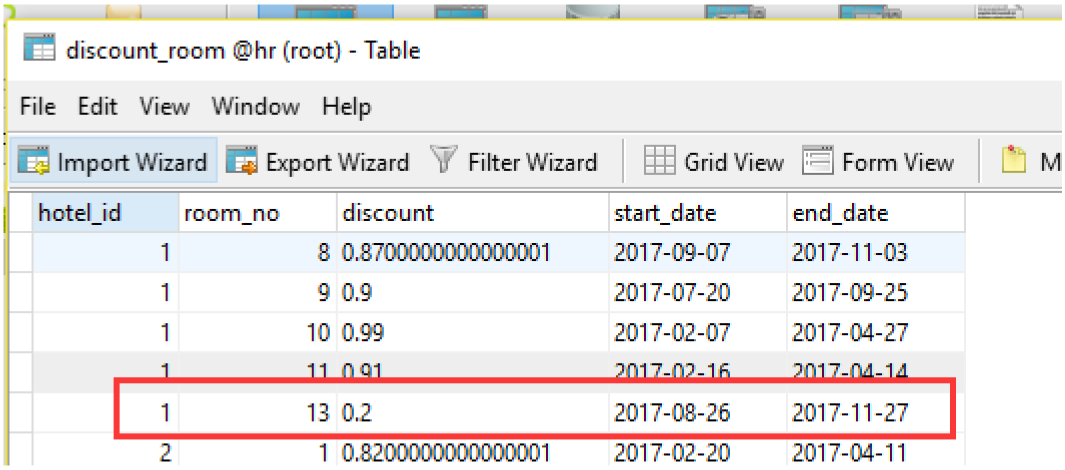
start date	yyyy - mm - dd
end date	yyyy - mm - dd

Add New Room Successfully



room @hr (root) - Table

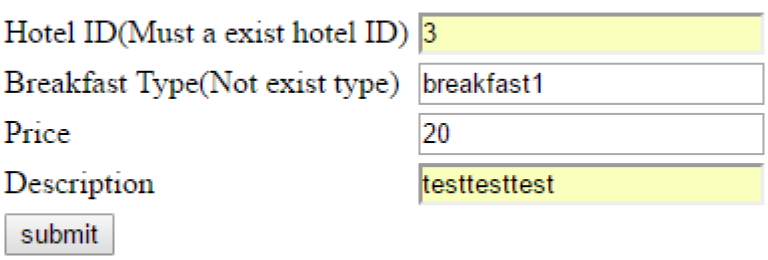
hotel_id	room_no	rtype	price	floor	capacity	room_description
1	10	double	400	0	2	double
1	11	double	400	1	2	double
1	13	typetest	123	2	2	testtesttest
2	1	double	400	1	2	double
2	2	suite	600	2	4	suite
2	3	deluxe	500	3	3	deluxe



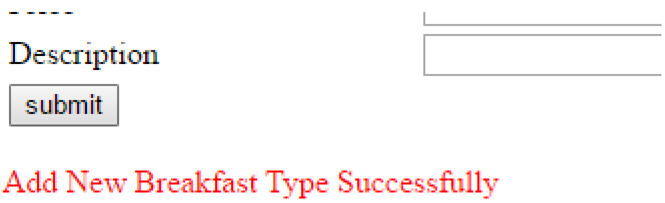
discount_room @hr (root) - Table

hotel_id	room_no	discount	start_date	end_date
1	8	0.8700000000000001	2017-09-07	2017-11-03
1	9	0.9	2017-07-20	2017-09-25
1	10	0.99	2017-02-07	2017-04-27
1	11	0.91	2017-02-16	2017-04-14
1	13	0.2	2017-08-26	2017-11-27
2	1	0.8200000000000001	2017-02-20	2017-04-11

25. Add a new type of breakfast:

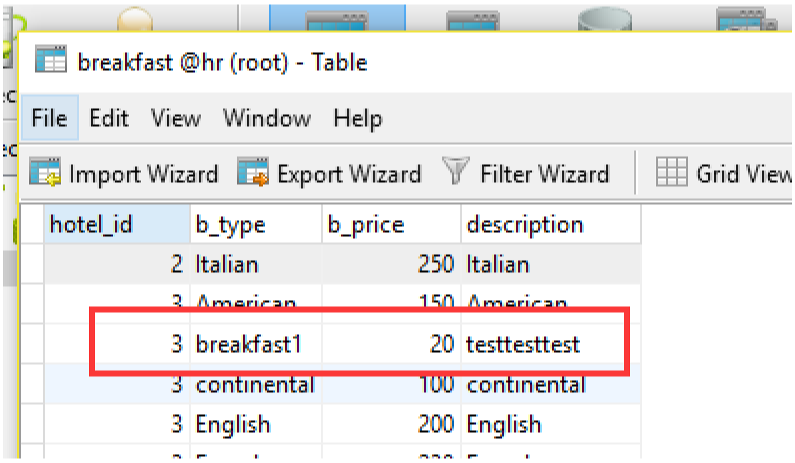


Hotel ID(Must a exist hotel ID)	3
Breakfast Type(Not exist type)	breakfast1
Price	20
Description	testtesttest



Description

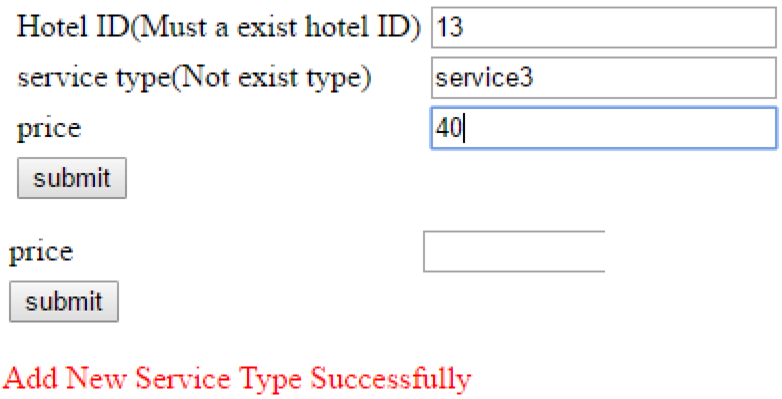
Add New Breakfast Type Successfully



breakfast @hr (root) - Table

hotel_id	b_type	b_price	description
2	Italian	250	Italian
3	American	150	American
3	breakfast1	20	testtesttest
3	continental	100	continental
3	English	200	English

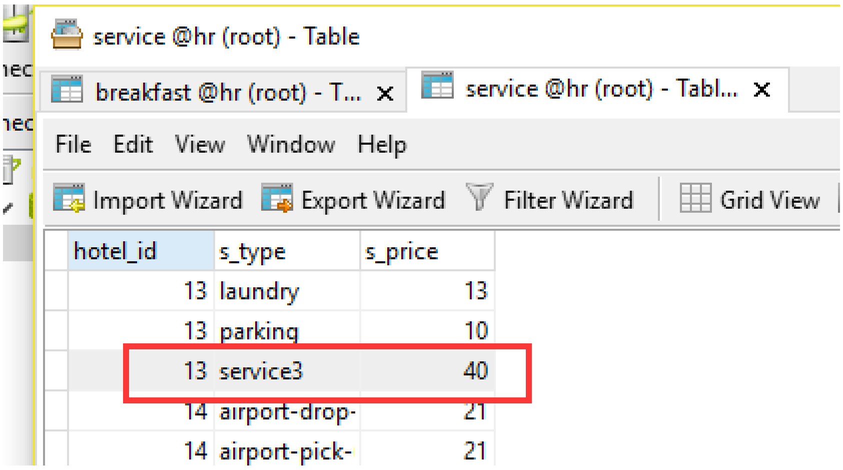
26. Add a new type of service:



Hotel ID(Must a exist hotel ID)	13
service type(Not exist type)	service3
price	40

price

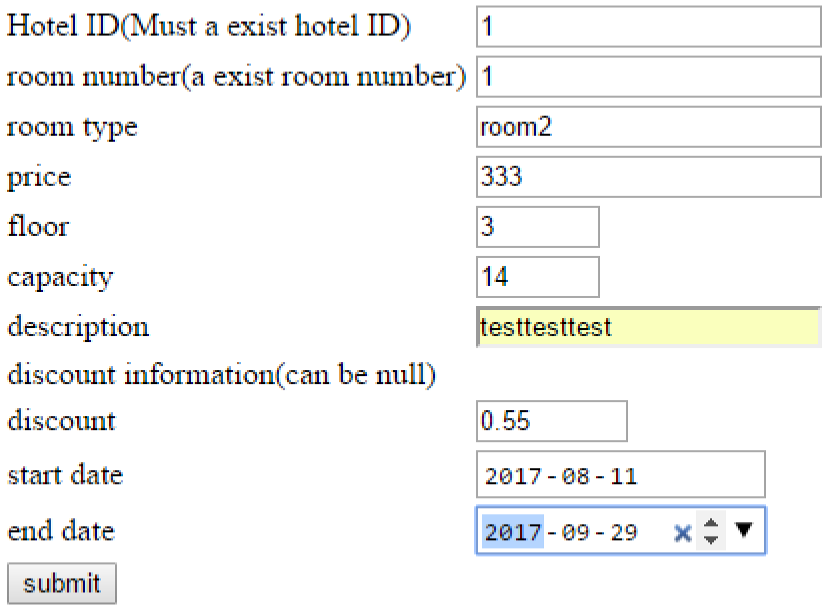
Add New Service Type Successfully



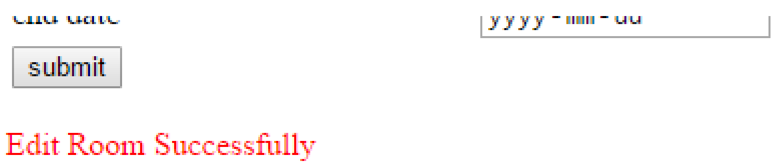
service @hr (root) - Table

hotel_id	s_type	s_price
13	laundry	13
13	parking	10
13	service3	40
14	airport-drop-	21
14	airport-pick-	21

27. Edit a exist room:

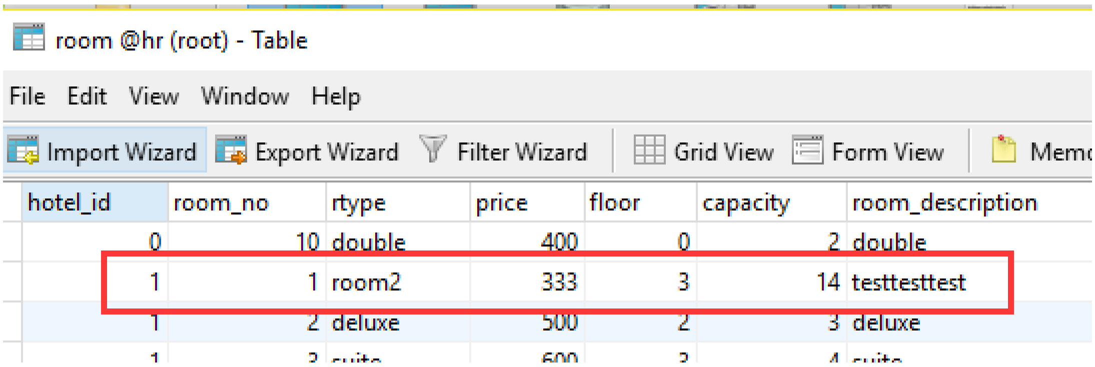


Hotel ID(Must a exist hotel ID)	1
room number(a exist room number)	1
room type	room2
price	333
floor	3
capacity	14
description	testtesttest
discount information(can be null)	
discount	0.55
start date	2017-08-11
end date	2017-09-29



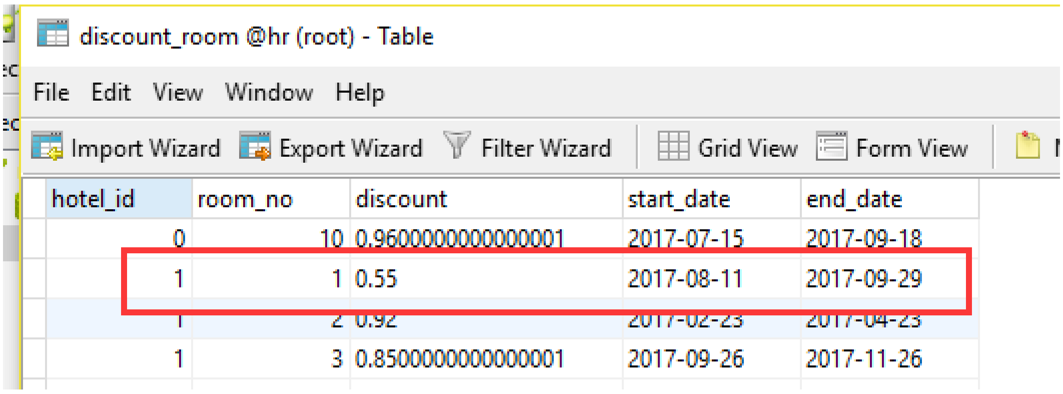
end date	yyyy-mm-dd

Edit Room Successfully



room @hr (root) - Table

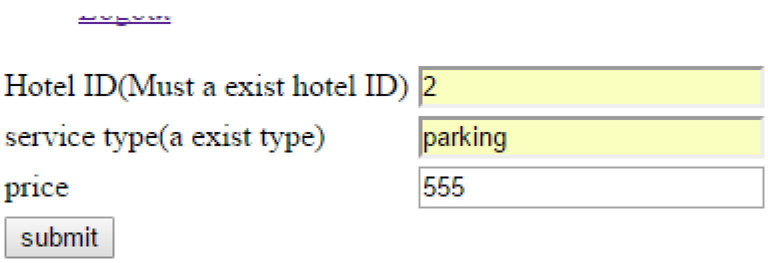
hotel_id	room_no	rtype	price	floor	capacity	room_description
0	10	double	400	0	2	double
1	1	room2	333	3	14	testtesttest
1	2	deluxe	500	2	3	deluxe
1	3	suite	600	3	4	suite



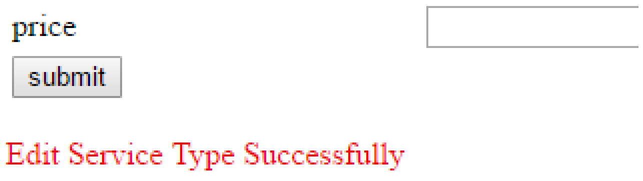
discount_room @hr (root) - Table

hotel_id	room_no	discount	start_date	end_date
0	10	0.9600000000000001	2017-07-15	2017-09-18
1	1	0.55	2017-08-11	2017-09-29
1	2	0.92	2017-02-23	2017-04-23
1	3	0.8500000000000001	2017-09-26	2017-11-26

28. Edit a exist service:

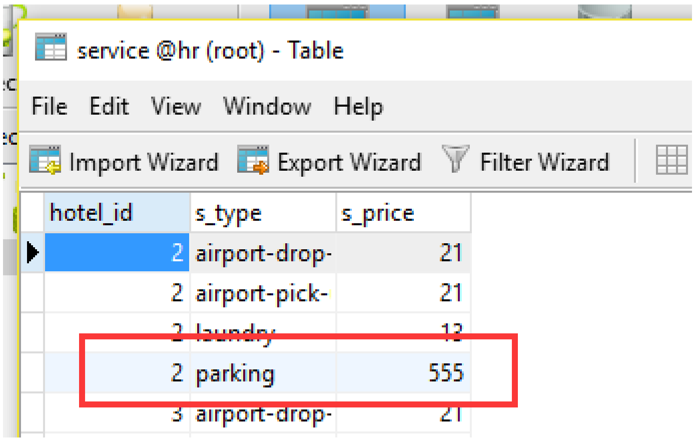


Hotel ID(Must a exist hotel ID)	2
service type(a exist type)	parking
price	555



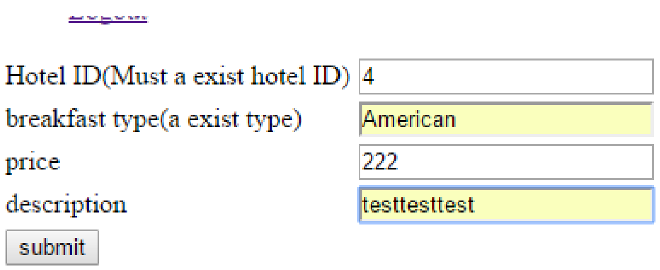
price	

Edit Service Type Successfully

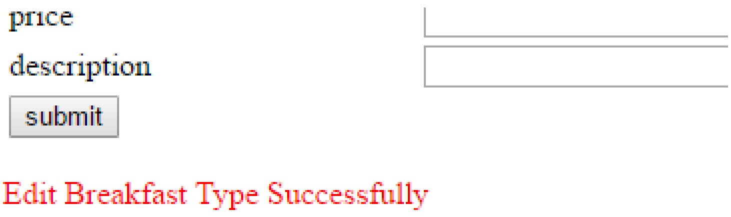


hotel_id	s_type	s_price
2	airport-drop-	21
2	airport-pick-	21
2	laundry	12
2	parking	555
3	airport-drop-	21

29. Edit a exist breakfast:

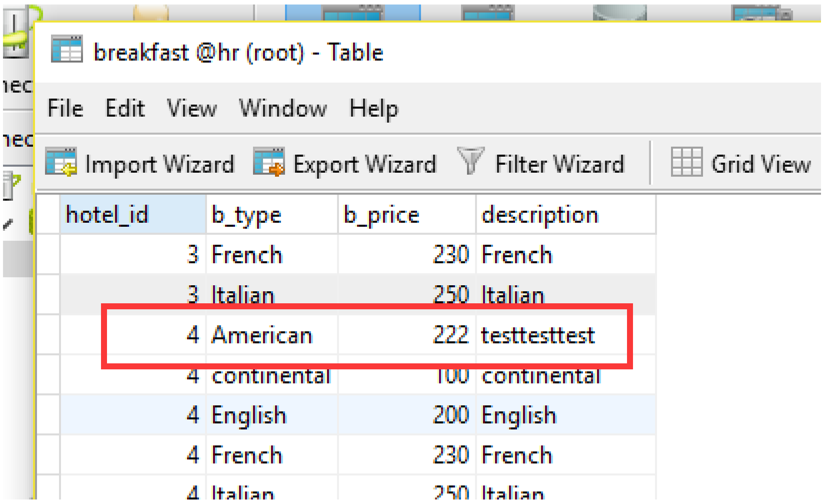


Hotel ID(Must a exist hotel ID)	4
breakfast type(a exist type)	American
price	222
description	testtesttest



price	
description	

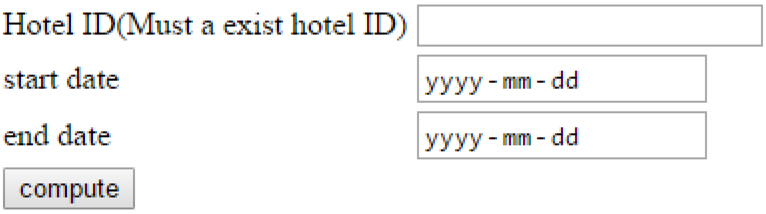
Edit Breakfast Type Successfully



breakfast @hr (root) - Table

hotel_id	b_type	b_price	description
3	French	230	French
3	Italian	250	Italian
4	American	222	testtesttest
4	continental	100	continental
4	English	200	English
4	French	230	French
4	Italian	250	Italian

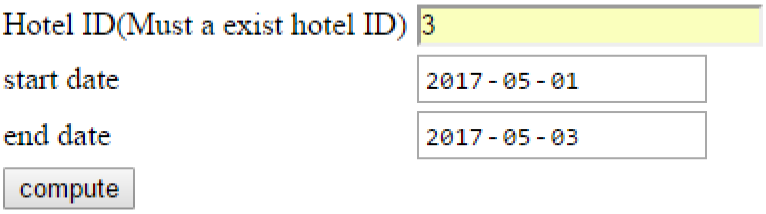
30. Compute the highest rated room type for a hotel:



Hotel ID(Must a exist hotel ID)

start date yyyy-mm-dd

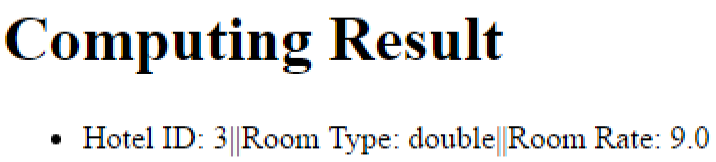
end date yyyy-mm-dd



Hotel ID(Must a exist hotel ID) 3

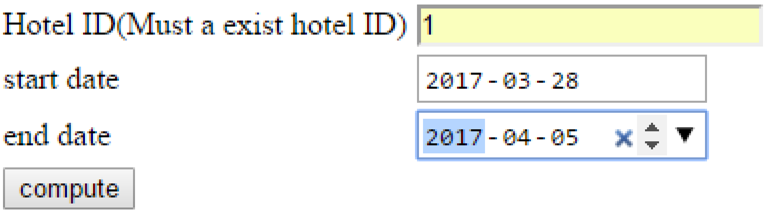
start date 2017-05-01

end date 2017-05-03



Computing Result

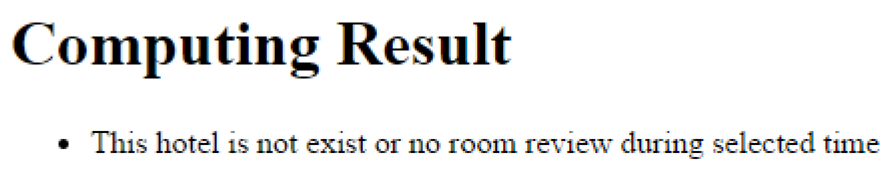
- Hotel ID: 3||Room Type: double||Room Rate: 9.0



Hotel ID(Must a exist hotel ID) 1

start date 2017-03-28

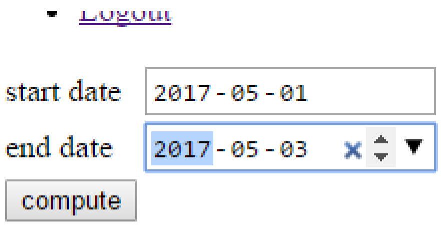
end date 2017-04-05



Computing Result

- This hotel is not exist or no room review during selected time

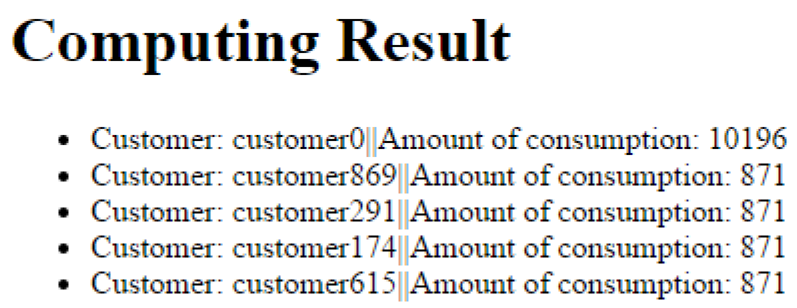
31. Compute the 5 best customers



- Logout

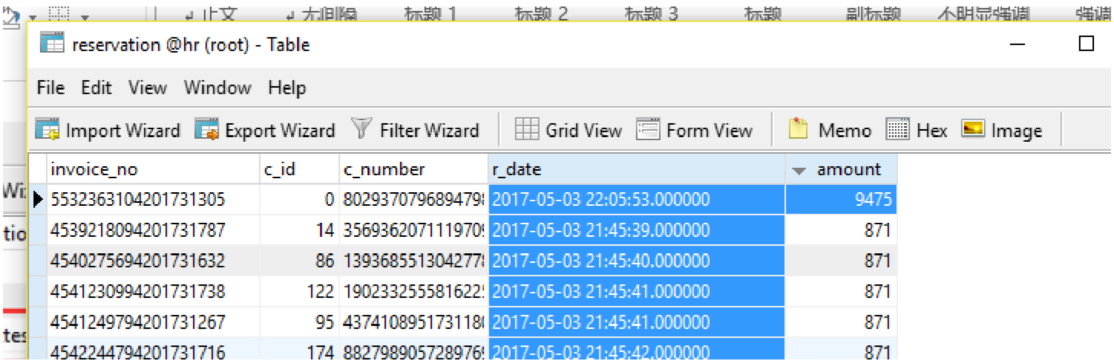
start date 2017-05-01

end date 2017-05-03



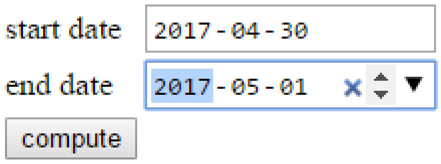
Computing Result

- Customer: customer0||Amount of consumption: 10196
- Customer: customer869||Amount of consumption: 871
- Customer: customer291||Amount of consumption: 871
- Customer: customer174||Amount of consumption: 871
- Customer: customer615||Amount of consumption: 871



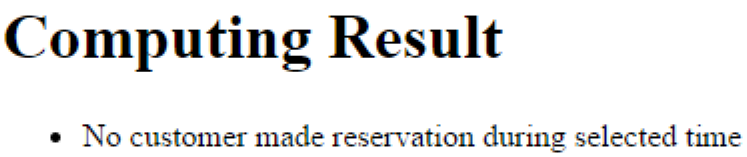
reservation @hr (root) - Table

invoice_no	c_id	c_number	r_date	amount
5532363104201731305	0	802937079689479	2017-05-03 22:05:53.000000	9475
4539218094201731787	14	356936207111970	2017-05-03 21:45:39.000000	871
4540275694201731632	86	139368551304277	2017-05-03 21:45:40.000000	871
4541230994201731738	122	190233255581622	2017-05-03 21:45:41.000000	871
4541249794201731267	95	437410895173118	2017-05-03 21:45:41.000000	871
4542244794201731716	174	882798905728976	2017-05-03 21:45:42.000000	871



start date 2017-04-30

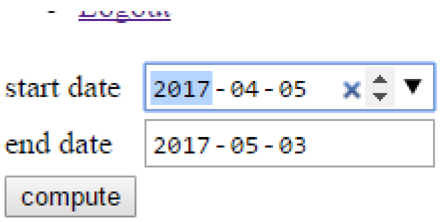
end date 2017-05-01



Computing Result

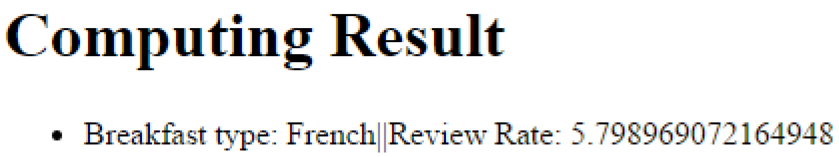
- No customer made reservation during selected time

32. Compute the highest rated breakfast type



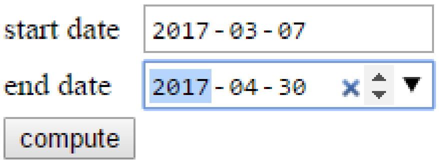
start date 2017-04-05

end date 2017-05-03



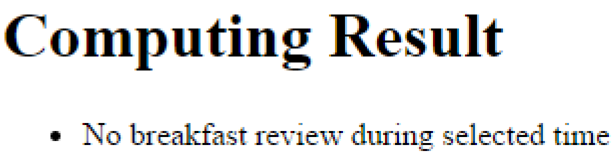
Computing Result

- Breakfast type: French||Review Rate: 5.798969072164948



start date 2017-03-07

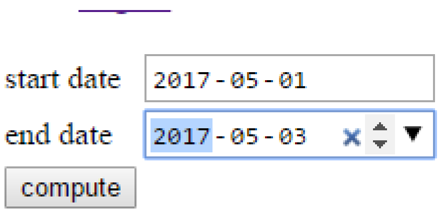
end date 2017-04-30



Computing Result

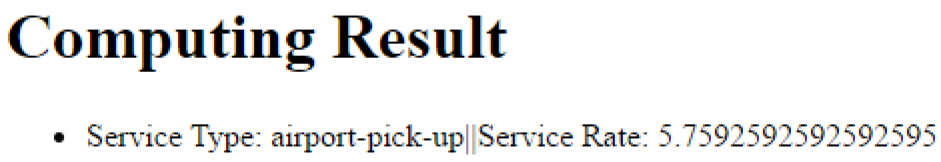
- No breakfast review during selected time

33. Compute the highest rated service type:



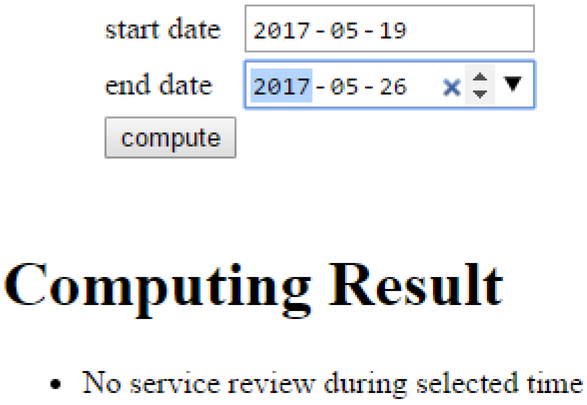
start date 2017-05-01

end date 2017-05-03



Computing Result

- Service Type: airport-pick-up||Service Rate: 5.7592592592592595



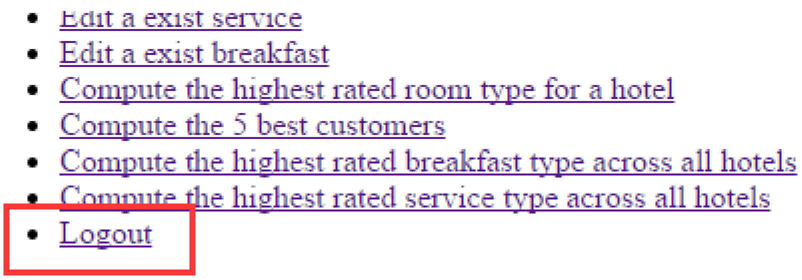
start date 2017-05-19

end date 2017-05-26

Computing Result

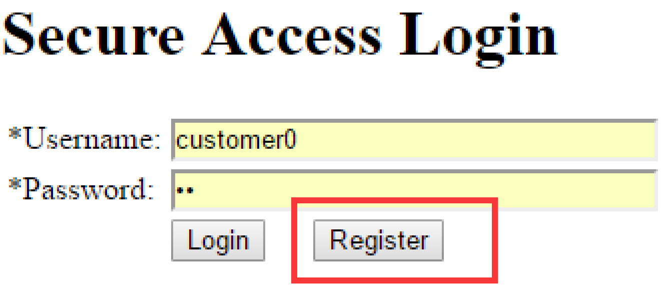
- No service review during selected time

34. Click Logout to logout:



- Edit a exist service
- Edit a exist breakfast
- Compute the highest rated room type for a hotel
- Compute the 5 best customers
- Compute the highest rated breakfast type across all hotels
- Compute the highest rated service type across all hotels
- Logout

35. Click Register to add a new manager account:

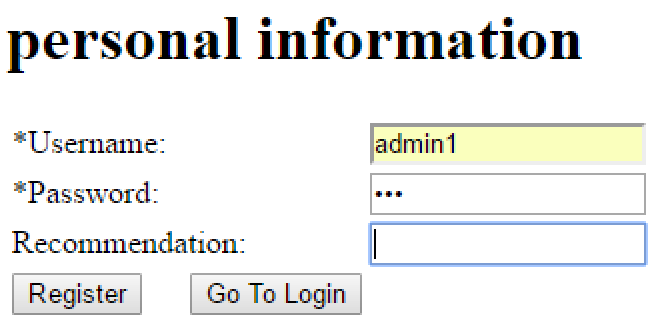


Secure Access Login

*Username: customer0

*Password: ••

36. Type the username and password, the recommendation can be null, the click Register:

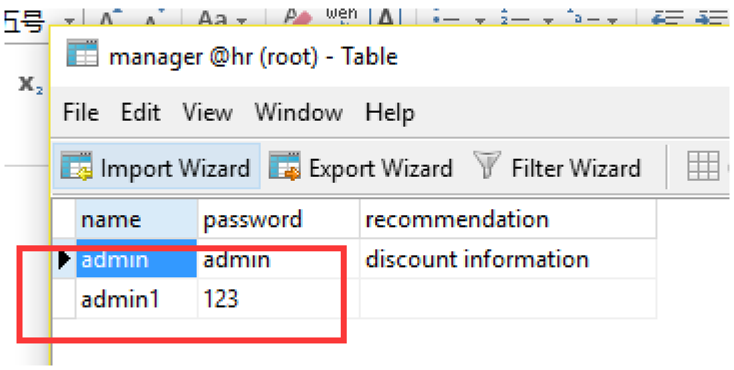


personal information

*Username: admin1

*Password: •••

Recommendation:



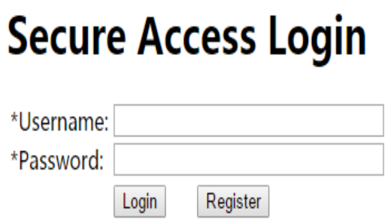
manager @hr (root) - Table

name	password	recommendation
admin	admin	discount information
admin1	123	

Users Guide

Customer Guide

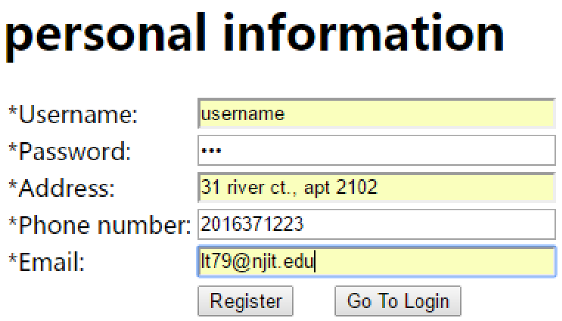
1. Sign up: click the signup button, type your information, click Register. Or click Go To Login to cancel the registration



Secure Access Login

*Username:

*Password:



personal information

*Username: username

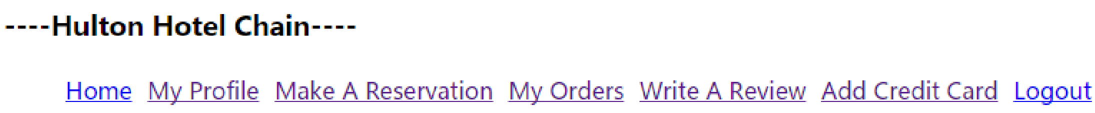
*Password: •••

*Address: 31 river ct., apt 2102

*Phone number: 2016371223

*Email: lt79@njit.edu

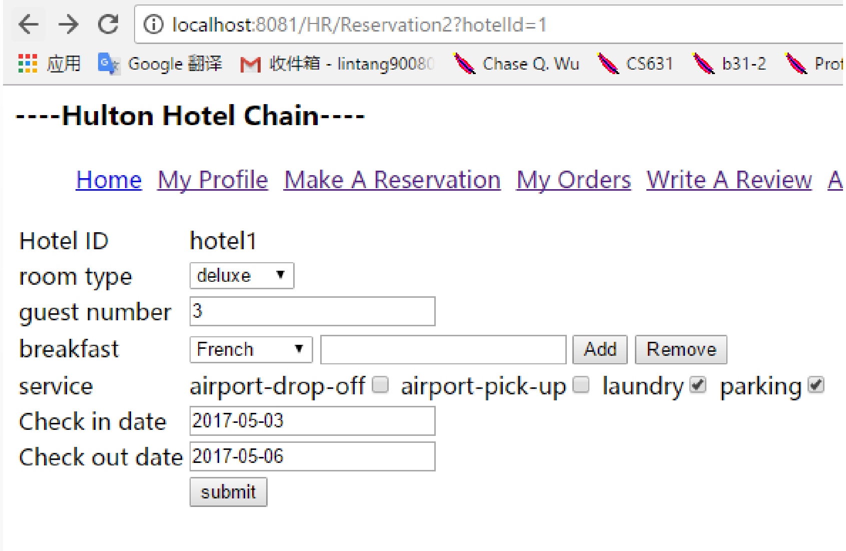
2. Log in: type your username and password, click login button



----Hulton Hotel Chain----

3. Edit your information: click My profile

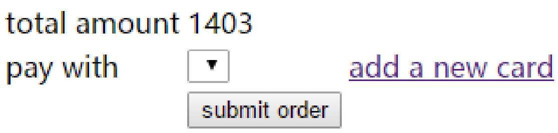
4. Make a room reservation: click Make A Reservation -> Room reservation -> type country and state -> click Search -> click the hotel -> fill up the reservation information -> click submit –> click make a payment -> add a new card -> click submit -> Make A Reservation -> make a payment -> submit order



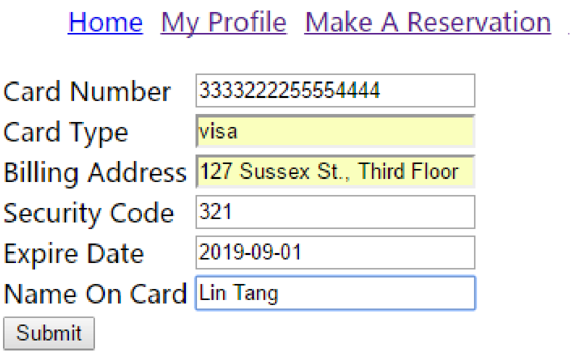
----Hulton Hotel Chain----

Home My Profile Make A Reservation My Orders Write A Review A

Hotel ID hotel1
room type deluxe ▾
guest number 3
breakfast French ▾ Add Remove
service airport-drop-off □ airport-pick-up □ laundry ☑ parking ☑
Check in date 2017-05-03
Check out date 2017-05-06
submit

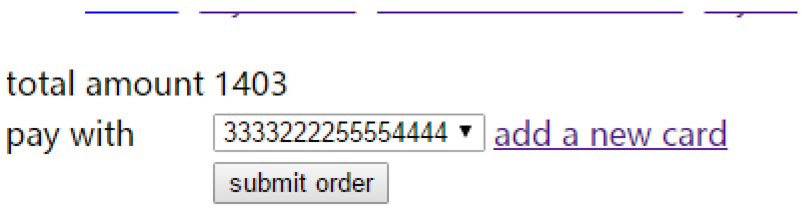


total amount 1403
pay with ▾ add a new card
submit order



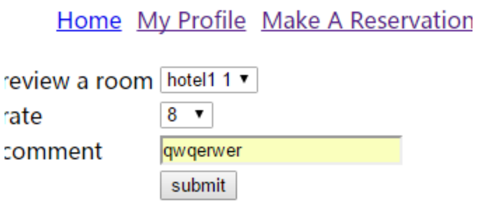
Home My Profile Make A Reservation

Card Number 3333222255554444
Card Type visa
Billing Address 127 Sussex St., Third Floor
Security Code 321
Expire Date 2019-09-01
Name On Card Lin Tang
Submit



total amount 1403
pay with 3333222255554444 ▾ add a new card
submit order

5. See orders: click My Orders
6. Write a review: click Write A Review -> click submit

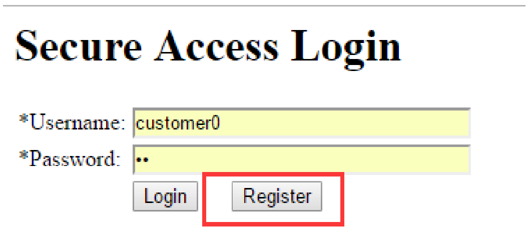


eview a room: hotel1 1 ▾
ate: 8 ▾
omment: qwqerwer

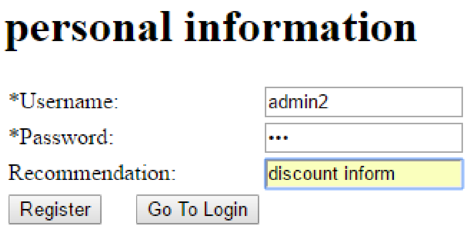
7. Add new credit card: click Add Credit Card and type your card information
8. Log out: click Logout

Administrator Guide

1. Sign up: click Register, type your information, recommendation can be null, then click Register to submit, or click Go To Login to cancel your registration.

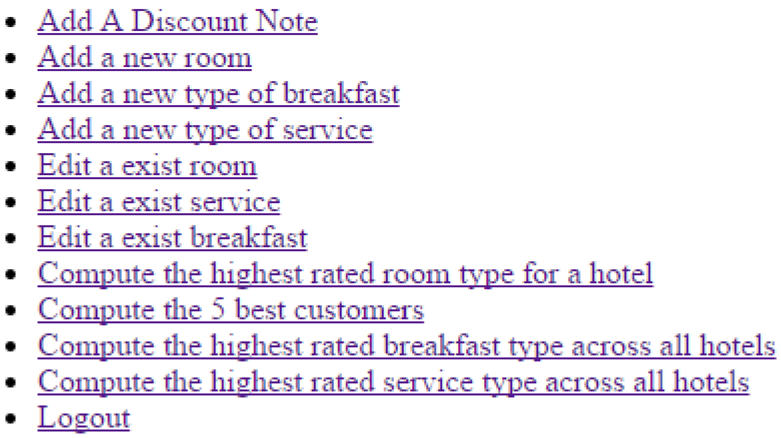


*Username: customer0
*Password: ••



*Username: admin2
*Password: •••
Recommendation: discount inform

2. All the options you can choose to do:

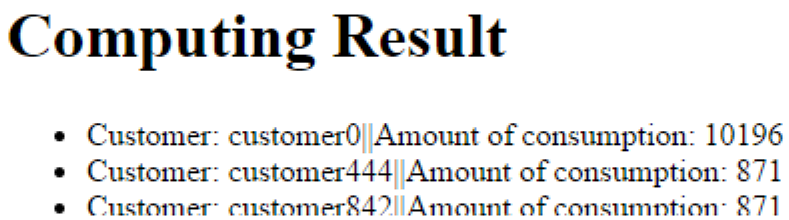


- Add A Discount Note
- Add a new room
- Add a new type of breakfast
- Add a new type of service
- Edit a exist room
- Edit a exist service
- Edit a exist breakfast
- Compute the highest rated room type for a hotel
- Compute the 5 best customers
- Compute the highest rated breakfast type across all hotels
- Compute the highest rated service type across all hotels
- Logout

3. Using the Add A Discount Note option to add a welcome message or a discount information, this message will shows at the top of customer's web page.
4. Using the Add a new room option to add a new room to any of hotels in the hulton hotel chain, and you also can add the room discount information in this option
5. Using the Add a new type of breakfast option to add a new type of breakfast to any of hotels in the hulton hotel chain
6. Using the Add a new type of service option to add a new type of service to any of hotels in the hulton hotel chain
7. Using the Edit a exit room option to edit a room of any of the hotels in the hulton hotel chain and you also can edit the room discount information in this option
8. Using the Edit a exist service option to edit a exist service of any of hotels in the hulton hotel

chain.

9. Using the Edit a exist breakfast option to edit a exist breakfast to any of hotels in the hulton hotel chain.
10. The four compute options can let you know the highest rated room type for a hotel, the 5 best customers. Highest rated breakfast type across all hotels and the highest rated service type across all hotels in a certain period of time.



Computing Result

- Customer: customer0||Amount of consumption: 10196
- Customer: customer444||Amount of consumption: 871
- Customer: customer842||Amount of consumption: 871

11. You can click the Logout option to log out.

SQL Commands

SQL Commands of Table Creation

```
/*
Navicat MySQL Data Transfer

Source Server         : root
Source Server Version : 50718
Source Host           : localhost:3306
Source Database       : hr

Target Server Type    : MYSQL
Target Server Version : 50718
File Encoding         : 65001

Date: 2017-04-29 09:24:59
*/

SET FOREIGN_KEY_CHECKS=0;

-- ----------------------------
-- Table structure for `breakfast`
-- ----------------------------
DROP TABLE IF EXISTS `breakfast`;
CREATE TABLE `breakfast` (
  `hotel_id` int(11) NOT NULL,
    `b_type` varchar(255) NOT NULL,
    `b_price` int(11) NOT NULL,
    `description` varchar(255) NOT NULL,
    PRIMARY KEY (`hotel_id`,`b_type`),
    CONSTRAINT `breakfast_ibfk_1` FOREIGN KEY (`hotel_id`) REFERENCES `hotel` (`hotel_id`) ON
DELETE CASCADE ON UPDATE CASCADE
) ENGINE=InnoDB DEFAULT CHARSET=utf8;

-- ----------------------------
-- Records of breakfast
-- ----------------------------

-- ----------------------------
-- Table structure for `breakfast_review`
-- ----------------------------
DROP TABLE IF EXISTS `breakfast_review`;
CREATE TABLE `breakfast_review` (
    `r_id` int(11) NOT NULL AUTO_INCREMENT,
    `hotel_id` int(11) NOT NULL,
    `b_type` varchar(255) NOT NULL,
    `c_id` int(11) NOT NULL,
    `text` varchar(255) NOT NULL,
    `rating` double NOT NULL,
    `rdate` datetime NOT NULL,
    PRIMARY KEY (`r_id`),
    KEY `breakfase_review_ibfk_2` (`hotel_id`,`b_type`),
    KEY `c_id` (`c_id`),
    CONSTRAINT `breakfast_review_ibfk_2` FOREIGN KEY (`hotel_id`, `b_type`) REFERENCES
`breakfast` (`hotel_id`, `b_type`) ON DELETE CASCADE ON UPDATE CASCADE,
    CONSTRAINT `breakfast_review_ibfk_3` FOREIGN KEY (`c_id`) REFERENCES `customer` (`c_id`)
ON DELETE CASCADE ON UPDATE CASCADE
) ENGINE=InnoDB AUTO_INCREMENT=3001 DEFAULT CHARSET=utf8;

-- ----------------------------
-- Records of breakfast_review
-- ----------------------------

-- ----------------------------
-- Table structure for `credit_card`
-- ----------------------------
DROP TABLE IF EXISTS `credit_card`;
CREATE TABLE `credit_card` (
    `c_number` varchar(255) NOT NULL,
    `c_type` varchar(255) NOT NULL,
  `baddress` varchar(255) NOT NULL,
  `code` int(11) NOT NULL,
  `exp_date` datetime NOT NULL,
  `name` varchar(255) NOT NULL,
  `c_id` int(11) NOT NULL,
  PRIMARY KEY (`c_number`),
  KEY `c_id` (`c_id`),
  CONSTRAINT `credit_card_ibfk_1` FOREIGN KEY (`c_id`) REFERENCES `customer` (`c_id`) ON DELETE CASCADE ON UPDATE CASCADE
) ENGINE=InnoDB DEFAULT CHARSET=utf8;

-- ----------------------------
-- Records of credit_card
-- ----------------------------

-- ----------------------------
-- Table structure for `customer`
-- ----------------------------
DROP TABLE IF EXISTS `customer`;
CREATE TABLE `customer` (
  `c_id` int(11) NOT NULL,
  `name` varchar(255) NOT NULL,
  `address` varchar(255) NOT NULL,
  `phone` varchar(255) NOT NULL,
  `email` varchar(255) NOT NULL,
  `password` varchar(255) NOT NULL,
  PRIMARY KEY (`c_id`)
) ENGINE=InnoDB DEFAULT CHARSET=utf8;

-- ----------------------------
-- Records of customer
-- ----------------------------

-- ----------------------------
-- Table structure for `discount_room`
-- ----------------------------
DROP TABLE IF EXISTS `discount_room`;
CREATE TABLE `discount_room` (
  `hotel_id` int(11) NOT NULL,
  `room_no` int(11) NOT NULL,
  `discount` double NOT NULL,
  `start_date` varchar(255) NOT NULL,
  `end_date` varchar(255) NOT NULL,
  PRIMARY KEY (`hotel_id`,`room_no`),
    CONSTRAINT `discount_room_ibfk_1` FOREIGN KEY (`hotel_id`, `room_no`) REFERENCES `room`
(`hotel_id`, `room_no`) ON DELETE CASCADE ON UPDATE CASCADE
) ENGINE=InnoDB DEFAULT CHARSET=utf8;

-- ----------------------------
-- Records of discount_room
-- ----------------------------

-- ----------------------------
-- Table structure for `hotel`
-- ----------------------------
DROP TABLE IF EXISTS `hotel`;
CREATE TABLE `hotel` (
    `hotel_id` int(11) NOT NULL,
    `street` varchar(255) NOT NULL,
    `country` varchar(255) NOT NULL,
    `state` varchar(255) NOT NULL,
    `zip` varchar(255) NOT NULL,
    `name` varchar(255) NOT NULL,
    PRIMARY KEY (`hotel_id`)
) ENGINE=InnoDB DEFAULT CHARSET=utf8;

-- ----------------------------
-- Records of hotel
-- ----------------------------

-- ----------------------------
-- Table structure for `manager`
-- ----------------------------
DROP TABLE IF EXISTS `manager`;
CREATE TABLE `manager` (
    `name` varchar(255) NOT NULL,
    `password` varchar(255) NOT NULL,
    `recommendation` varchar(1000) DEFAULT NULL,
    PRIMARY KEY (`name`)
) ENGINE=InnoDB DEFAULT CHARSET=utf8;

-- ----------------------------
-- Records of manager
-- ----------------------------

-- ----------------------------
-- Table structure for `phone`
-- ----------------------------
```

```
DROP TABLE IF EXISTS `phone`;
CREATE TABLE `phone` (
  `hotel_id` int(11) NOT NULL,
  `phone` varchar(255) NOT NULL,
  PRIMARY KEY (`hotel_id`,`phone`),
  KEY `phone_ibfk_1` (`hotel_id`),
  CONSTRAINT `phone_ibfk_1` FOREIGN KEY (`hotel_id`) REFERENCES `hotel` (`hotel_id`) ON DELETE CASCADE ON UPDATE CASCADE
) ENGINE=InnoDB DEFAULT CHARSET=utf8;

-- ----------------------------
-- Records of phone
-- ----------------------------

-- ----------------------------
-- Table structure for `reservation`
-- ----------------------------
DROP TABLE IF EXISTS `reservation`;
CREATE TABLE `reservation` (
  `invoice_no` varchar(255) NOT NULL,
  `c_id` int(11) NOT NULL,
  `c_number` varchar(255) NOT NULL,
  `r_date` datetime(6) NOT NULL,
  `amount` int(11) NOT NULL,
  PRIMARY KEY (`invoice_no`),
  KEY `c_id` (`c_id`),
  KEY `c_number` (`c_number`),
  CONSTRAINT `reservation_ibfk_2` FOREIGN KEY (`c_id`) REFERENCES `customer` (`c_id`) ON DELETE CASCADE ON UPDATE CASCADE,
  CONSTRAINT `reservation_ibfk_3` FOREIGN KEY (`c_number`) REFERENCES `credit_card` (`c_number`) ON DELETE CASCADE ON UPDATE CASCADE
) ENGINE=InnoDB DEFAULT CHARSET=utf8;

-- ----------------------------
-- Records of reservation
-- ----------------------------

-- ----------------------------
-- Table structure for `room`
-- ----------------------------
DROP TABLE IF EXISTS `room`;
CREATE TABLE `room` (
  `hotel_id` int(11) NOT NULL,
  `room_no` int(11) NOT NULL,
```

```sql
  `rtype` varchar(255) NOT NULL,
  `price` int(11) NOT NULL,
  `floor` int(11) NOT NULL,
  `capacity` int(11) NOT NULL,
  `room_description` varchar(255) NOT NULL,
  PRIMARY KEY (`hotel_id`,`room_no`),
  CONSTRAINT `room_ibfk_1` FOREIGN KEY (`hotel_id`) REFERENCES `hotel` (`hotel_id`) ON DELETE CASCADE ON UPDATE CASCADE
) ENGINE=InnoDB DEFAULT CHARSET=utf8;

-- ----------------------------
-- Records of room
-- ----------------------------

-- ----------------------------
-- Table structure for `room_reservation`
-- ----------------------------
DROP TABLE IF EXISTS `room_reservation`;
CREATE TABLE `room_reservation` (
  `invoice_no` varchar(255) NOT NULL,
  `hotel_id` int(11) NOT NULL,
  `room_no` int(11) NOT NULL,
  `check_in_date` datetime NOT NULL,
  `check_out_date` datetime NOT NULL,
  PRIMARY KEY (`hotel_id`,`room_no`,`check_in_date`),
  KEY `invoice_no` (`invoice_no`),
  CONSTRAINT `room_reservation_ibfk_1` FOREIGN KEY (`hotel_id`, `room_no`) REFERENCES `room` (`hotel_id`, `room_no`) ON DELETE CASCADE ON UPDATE CASCADE,
  CONSTRAINT `room_reservation_ibfk_2` FOREIGN KEY (`invoice_no`) REFERENCES `reservation` (`invoice_no`) ON DELETE CASCADE ON UPDATE CASCADE
) ENGINE=InnoDB DEFAULT CHARSET=utf8;

-- ----------------------------
-- Records of room_reservation
-- ----------------------------

-- ----------------------------
-- Table structure for `room_review`
-- ----------------------------
DROP TABLE IF EXISTS `room_review`;
CREATE TABLE `room_review` (
  `rid` int(11) NOT NULL AUTO_INCREMENT,
  `rating` int(11) NOT NULL,
  `text` varchar(255) NOT NULL,
  `cid` int(11) NOT NULL,
  `hotel_id` int(11) NOT NULL,
  `room_no` int(11) NOT NULL,
  `rdate` datetime DEFAULT NULL,
  PRIMARY KEY (`rid`),
  KEY `room_review_ibfk_1` (`hotel_id`,`room_no`),
  KEY `cid` (`cid`),
  CONSTRAINT `room_review_ibfk_1` FOREIGN KEY (`hotel_id`, `room_no`) REFERENCES `room`
(`hotel_id`, `room_no`) ON DELETE CASCADE ON UPDATE CASCADE,
  CONSTRAINT `room_review_ibfk_2` FOREIGN KEY (`cid`) REFERENCES `customer` (`c_id`) ON
DELETE CASCADE ON UPDATE CASCADE
) ENGINE=InnoDB AUTO_INCREMENT=3001 DEFAULT CHARSET=utf8;

-- ----------------------------
-- Records of room_review
-- ----------------------------

-- ----------------------------
-- Table structure for `rresv_breakfast`
-- ----------------------------
DROP TABLE IF EXISTS `rresv_breakfast`;
CREATE TABLE `rresv_breakfast` (
  `hotel_id` int(11) NOT NULL,
  `b_type` varchar(255) NOT NULL,
  `room_no` int(11) NOT NULL,
  `check_in_date` datetime NOT NULL,
  `no_of_orders` int(255) NOT NULL,
  PRIMARY KEY (`b_type`,`hotel_id`,`room_no`,`check_in_date`),
  KEY `rresv_breakfast_ibfk_1` (`hotel_id`,`b_type`),
  KEY `hotel_id` (`hotel_id`,`room_no`,`check_in_date`),
  CONSTRAINT `rresv_breakfast_ibfk_1` FOREIGN KEY (`hotel_id`, `b_type`) REFERENCES
`breakfast` (`hotel_id`, `b_type`) ON DELETE CASCADE ON UPDATE CASCADE,
  CONSTRAINT `rresv_breakfast_ibfk_2` FOREIGN KEY (`hotel_id`, `room_no`, `check_in_date`)
REFERENCES `room_reservation` (`hotel_id`, `room_no`, `check_in_date`) ON DELETE CASCADE
ON UPDATE CASCADE
) ENGINE=InnoDB DEFAULT CHARSET=utf8;

-- ----------------------------
-- Records of rresv_breakfast
-- ----------------------------

-- ----------------------------
-- Table structure for `rresv_service`
-- ----------------------------
```

```
DROP TABLE IF EXISTS `rresv_service`;
CREATE TABLE `rresv_service` (
  `hotel_id` int(11) NOT NULL,
  `s_type` varchar(255) NOT NULL,
  `room_no` int(11) NOT NULL,
  `check_in_date` datetime NOT NULL,
  PRIMARY KEY (`s_type`,`hotel_id`,`room_no`,`check_in_date`),
  KEY `rresv_service_ibfk_1` (`hotel_id`,`s_type`),
  KEY `hotel_id` (`hotel_id`,`room_no`,`check_in_date`),
  CONSTRAINT `rresv_service_ibfk_1` FOREIGN KEY (`hotel_id`, `s_type`) REFERENCES `service`
(`hotel_id`, `s_type`) ON DELETE CASCADE ON UPDATE CASCADE,
  CONSTRAINT `rresv_service_ibfk_2` FOREIGN KEY (`hotel_id`, `room_no`, `check_in_date`)
REFERENCES `room_reservation` (`hotel_id`, `room_no`, `check_in_date`) ON DELETE CASCADE
ON UPDATE CASCADE
) ENGINE=InnoDB DEFAULT CHARSET=utf8;

-- ----------------------------
-- Records of rresv_service
-- ----------------------------

-- ----------------------------
-- Table structure for `service`
-- ----------------------------
DROP TABLE IF EXISTS `service`;
CREATE TABLE `service` (
  `hotel_id` int(11) NOT NULL,
  `s_type` varchar(255) NOT NULL,
  `s_price` int(11) NOT NULL,
  PRIMARY KEY (`hotel_id`,`s_type`),
  CONSTRAINT `service_ibfk_1` FOREIGN KEY (`hotel_id`) REFERENCES `hotel` (`hotel_id`) ON
DELETE CASCADE ON UPDATE CASCADE
) ENGINE=InnoDB DEFAULT CHARSET=utf8;

-- ----------------------------
-- Records of service
-- ----------------------------

-- ----------------------------
-- Table structure for `service_review`
-- ----------------------------
DROP TABLE IF EXISTS `service_review`;
CREATE TABLE `service_review` (
  `r_id` int(11) NOT NULL AUTO_INCREMENT,
  `hotel_id` int(11) NOT NULL,
  `s_type` varchar(255) NOT NULL,
  `c_id` int(11) NOT NULL,
  `rating` double NOT NULL,
  `text` varchar(255) NOT NULL,
  `rdate` datetime NOT NULL,
  PRIMARY KEY (`r_id`),
  KEY `service_review_ibfk_2` (`hotel_id`,`s_type`),
  KEY `c_id` (`c_id`),
  CONSTRAINT `service_review_ibfk_2` FOREIGN KEY (`hotel_id`, `s_type`) REFERENCES `service` (`hotel_id`, `s_type`) ON DELETE CASCADE ON UPDATE CASCADE,
  CONSTRAINT `service_review_ibfk_3` FOREIGN KEY (`c_id`) REFERENCES `customer` (`c_id`) ON DELETE CASCADE ON UPDATE CASCADE
) ENGINE=InnoDB AUTO_INCREMENT=3001 DEFAULT CHARSET=utf8;

-- ----------------------------
-- Records of service_review
-- ----------------------------
```

SQL Commands of Table Population (Partial)

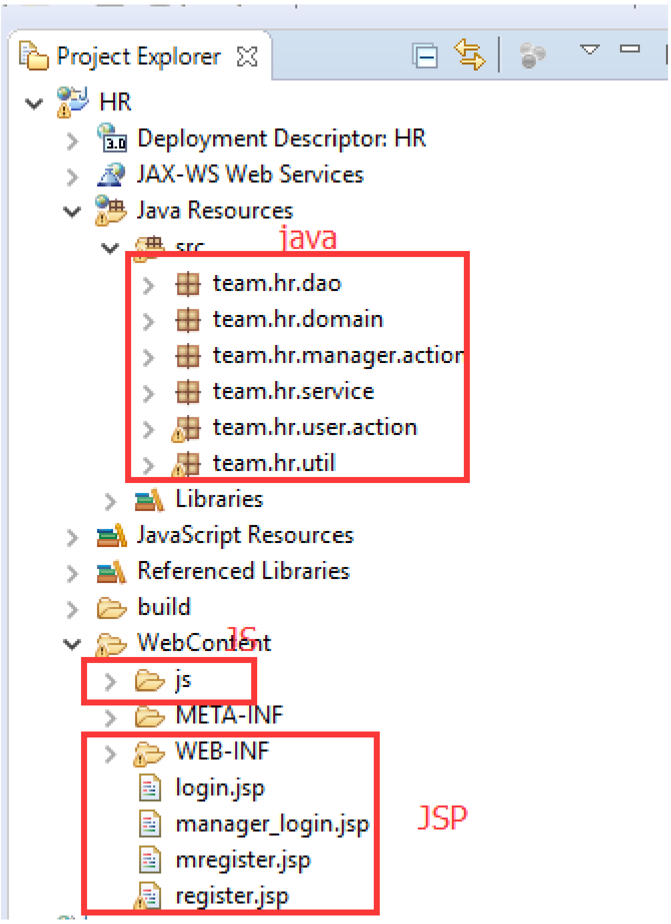
```
-- ----------------------------
-- Records of breakfast
-- ----------------------------
INSERT INTO `breakfast` VALUES ('0', 'American', '150', 'American');
INSERT INTO `breakfast` VALUES ('0', 'continental', '100', 'continental');
INSERT INTO `breakfast` VALUES ('0', 'English', '200', 'English');
INSERT INTO `breakfast` VALUES ('0', 'French', '230', 'French');
INSERT INTO `breakfast` VALUES ('0', 'Italian', '250', 'Italian');
INSERT INTO `breakfast` VALUES ('1', 'American', '150', 'American');

-- ----------------------------
-- Records of breakfast_review
-- ----------------------------
INSERT INTO `breakfast_review` VALUES ('3001', '1', 'American', '0', 'breakfast review', '7', '2017-04-29 09:29:17');
INSERT INTO `breakfast_review` VALUES ('3002', '2', 'continental', '1', 'breakfast review', '10', '2017-04-29 09:29:17');
INSERT INTO `breakfast_review` VALUES ('3003', '3', 'continental', '2', 'breakfast review', '7', '2017-04-29 09:29:17');
INSERT INTO `breakfast_review` VALUES ('3004', '4', 'continental', '3', 'breakfast review', '9', '2017-04-29 09:29:17');

-- ----------------------------
-- Records of credit_card
-- ----------------------------
INSERT INTO `credit_card` VALUES ('1017258780368178', 'Amex', 'address410', '535', '2030-04-03 00:00:00', 'c410', '410');
INSERT INTO `credit_card` VALUES ('1043210066621620', 'Discover', 'address450', '539', '2030-04-03 00:00:00', 'c450', '450');
INSERT INTO `credit_card` VALUES ('1052634640785415', 'Discover', 'address389', '692', '2030-04-03 00:00:00', 'c389', '389');
INSERT INTO `credit_card` VALUES ('1066565136632600', 'Master', 'address129', '456', '2030-04-03 00:00:00', 'c129', '129');

-- ----------------------------
-- Records of customer
-- ----------------------------
INSERT INTO `customer` VALUES ('0', 'customer0', 'street0', '15626485430', '2347820c@gmail.com', 'p0');
INSERT INTO `customer` VALUES ('1', 'customer1', 'street1', '15626485431', '2347821c@gmail.com', 'p1');
INSERT INTO `customer` VALUES ('2', 'customer2', 'street2', '15626485432', '2347822c@gmail.com', 'p2');
INSERT INTO `customer` VALUES ('3', 'customer3', 'street3', '15626485433', '2347823c@gmail.com', 'p3');

-- ----------------------------
-- Records of discount_room
-- ----------------------------
INSERT INTO `discount_room` VALUES ('0', '1', '0.98', '2017-04-20', '2017-06-13');
INSERT INTO `discount_room` VALUES ('0', '2', '0.99', '2017-08-08', '2017-10-16');
INSERT INTO `discount_room` VALUES ('0', '3', '0.9', '2017-02-05', '2017-04-10');
INSERT INTO `discount_room` VALUES ('0', '4', '0.8', '2017-10-24', '2017-12-06');
INSERT INTO `discount_room` VALUES ('0', '5', '0.8400000000000001', '2017-04-14', '2017-06-23');

-- ----------------------------
-- Records of hotel
-- ----------------------------
INSERT INTO `hotel` VALUES ('0', 'street0', 'country0', 'state0', '07029', 'hotel0');
INSERT INTO `hotel` VALUES ('1', 'street1', 'country1', 'state1', '07030', 'hotel1');
INSERT INTO `hotel` VALUES ('2', 'street2', 'country2', 'state2', '07031', 'hotel2');
INSERT INTO `hotel` VALUES ('3', 'street3', 'country3', 'state3', '07032', 'hotel3');

-- ----------------------------
-- Records of manager
-- ----------------------------
INSERT INTO `manager` VALUES ('admin', 'admin', 'Hulton Hotel Chain');
```

```
-- ----------------------------
-- Records of phone
-- ----------------------------
INSERT INTO `phone` VALUES ('0', '13911124555');
INSERT INTO `phone` VALUES ('0', '13917793333');
INSERT INTO `phone` VALUES ('0', '13927698906');
INSERT INTO `phone` VALUES ('1', '13956801052');
INSERT INTO `phone` VALUES ('1', '13957156215');
INSERT INTO `phone` VALUES ('1', '13959076272');

-- ----------------------------
-- Records of reservation
-- ----------------------------
INSERT INTO `reservation` VALUES ('29172056932017291190', '12', '1400655115772624', '2017-04-29 09:29:17.000000', '510');
INSERT INTO `reservation` VALUES ('29172124932017291466', '9', '7389753215178781', '2017-04-29 09:29:17.000000', '560');
INSERT INTO `reservation` VALUES ('29172211932017291486', '0', '2534397781775065', '2017-04-29 09:29:17.000000', '560');
INSERT INTO `reservation` VALUES ('29172234932017291127', '2', '8668629889848350', '2017-04-29 09:29:17.000000', '621');

-- ----------------------------
-- Records of room
-- ----------------------------
INSERT INTO `room` VALUES ('0', '1', 'standard', '300', '1', '1', 'standard');
INSERT INTO `room` VALUES ('0', '2', 'suite', '600', '2', '4', 'suite');
INSERT INTO `room` VALUES ('0', '3', 'deluxe', '500', '3', '3', 'deluxe');
INSERT INTO `room` VALUES ('0', '4', 'standard', '300', '4', '1', 'standard');
INSERT INTO `room` VALUES ('0', '5', 'standard', '300', '0', '1', 'standard');
INSERT INTO `room` VALUES ('1', '1', 'deluxe', '500', '1', '3', 'deluxe');
INSERT INTO `room` VALUES ('1', '2', 'deluxe', '500', '2', '3', 'deluxe');
INSERT INTO `room` VALUES ('1', '3', 'suite', '600', '3', '4', 'suite');

-- ----------------------------
-- Records of room_reservation
-- ----------------------------
INSERT INTO `room_reservation` VALUES ('29172211932017291486', '1', '1', '2017-06-02 00:00:00', '2017-06-03 00:00:00');
INSERT INTO `room_reservation` VALUES ('29202655932017291009', '1', '2', '2017-06-06 00:00:00', '2017-06-07 00:00:00');
INSERT INTO `room_reservation` VALUES ('29232302932017291670', '1', '3', '2017-06-10 00:00:00', '2017-06-11 00:00:00');
INSERT INTO `room_reservation` VALUES ('29262213932017291044', '1', '4', '2017-06-14 00:00:00',
'2017-06-15 00:00:00');
INSERT INTO `room_reservation` VALUES ('29282068932017291612', '1', '5', '2017-06-18 00:00:00',
'2017-06-19 00:00:00');

-- ----------------------------
-- Records of room_review
-- ----------------------------
INSERT INTO `room_review` VALUES ('3001', '1', 'roomreview', '0', '1', '1', '2017-04-29 09:29:17');
INSERT INTO `room_review` VALUES ('3002', '9', 'roomreview', '1', '2', '2', '2017-04-29 09:29:17');
INSERT INTO `room_review` VALUES ('3003', '10', 'roomreview', '2', '3', '3', '2017-04-29 09:29:17');
INSERT INTO `room_review` VALUES ('3004', '10', 'roomreview', '3', '4', '4', '2017-04-29 09:29:17');
INSERT INTO `room_review` VALUES ('3005', '7', 'roomreview', '4', '5', '5', '2017-04-29 09:29:17');

-- ----------------------------
-- Records of rresv_breakfast
-- ----------------------------
INSERT INTO `rresv_breakfast` VALUES ('2', 'American', '6', '2017-06-18 00:00:00', '1');
INSERT INTO `rresv_breakfast` VALUES ('3', 'American', '6', '2017-06-14 00:00:00', '1');
INSERT INTO `rresv_breakfast` VALUES ('4', 'American', '4', '2017-06-02 00:00:00', '1');
INSERT INTO `rresv_breakfast` VALUES ('5', 'American', '1', '2017-06-22 00:00:00', '1');
INSERT INTO `rresv_breakfast` VALUES ('5', 'American', '9', '2017-06-18 00:00:00', '1');

-- ----------------------------
-- Records of rresv_service
-- ----------------------------
INSERT INTO `rresv_service` VALUES ('1', 'parking', '1', '2017-06-02 00:00:00');
INSERT INTO `rresv_service` VALUES ('1', 'airport-drop-off', '2', '2017-06-06 00:00:00');
INSERT INTO `rresv_service` VALUES ('1', 'airport-drop-off', '3', '2017-06-10 00:00:00');
INSERT INTO `rresv_service` VALUES ('1', 'airport-drop-off', '4', '2017-06-14 00:00:00');
INSERT INTO `rresv_service` VALUES ('1', 'airport-pick-up', '5', '2017-06-18 00:00:00');
INSERT INTO `rresv_service` VALUES ('1', 'airport-pick-up', '6', '2017-06-22 00:00:00');

-- ----------------------------
-- Records of service
-- ----------------------------
INSERT INTO `service` VALUES ('0', 'airport-drop-off', '21');
INSERT INTO `service` VALUES ('0', 'airport-pick-up', '21');
INSERT INTO `service` VALUES ('0', 'laundry', '13');
INSERT INTO `service` VALUES ('0', 'parking', '10');
INSERT INTO `service` VALUES ('1', 'airport-drop-off', '21');
INSERT INTO `service` VALUES ('1', 'airport-pick-up', '21');
INSERT INTO `service` VALUES ('1', 'laundry', '13');
INSERT INTO `service` VALUES ('1', 'parking', '10');
```

```
-- ----------------------------
-- Records of service_review
-- ----------------------------
INSERT INTO `service_review` VALUES ('3001', '1', 'airport-pick-up', '0', '4', 'service review', '2017-04-29 09:29:17');
INSERT INTO `service_review` VALUES ('3002', '2', 'parking', '1', '5', 'service review', '2017-04-29 09:29:17');
INSERT INTO `service_review` VALUES ('3003', '3', 'airport-drop-off', '2', '3', 'service review', '2017-04-29 09:29:17');
INSERT INTO `service_review` VALUES ('3004', '4', 'laundry', '3', '5', 'service review', '2017-04-29 09:29:17');
INSERT INTO `service_review` VALUES ('3005', '5', 'airport-pick-up', '4', '2', 'service review', '2017-04-29 09:29:17');
INSERT INTO `service_review` VALUES ('3006', '6', 'parking', '5', '9', 'service review', '2017-04-29 09:29:17');
```

Source Code

There are three part of our source code: java part, jsp and js part.



Project Explorer

- HR
  - Deployment Descriptor: HR
  - JAX-WS Web Services
  - Java Resources
    - src
      - team.hr.dao
      - team.hr.domain
      - team.hr.manager.action
      - team.hr.service
      - team.hr.user.action
      - team.hr.util
    - Libraries
  - JavaScript Resources
  - Referenced Libraries
  - build
  - WebContent
    - js
    - META-INF
    - WEB-INF
    - login.jsp
    - manager_login.jsp
    - mregister.jsp
    - register.jsp

Java (Partial)

Part of MyDao.java

```java
private void execSQLNoRs(String sql, Object... args) {
        PreparedStatement ps = null;
        ResultSet rs = null;
        try {
            ps = getCon().prepareStatement(sql);
            for (int i = 0; i < args.length; i++) {
                if(args.getClass().getName().equals("java.sql.Date")){
                    ps.setTimestamp(i+1, (Timestamp) args[i]);
                }
                else{
                    ps.setObject(i + 1, args[i]);
                }
            }
            ps.execute();
            rs = ps.getResultSet();

        } catch (SQLException e) {

            e.printStackTrace();
        }finally{
            try {
                if(rs!=null)
                rs.close();
                if(ps!=null)
                ps.close();
            } catch (SQLException e) {
                // TODO Auto-generated catch block
            }
        }
    }

public int queryCount(String tableName){
        int result = 0;
        String sql = " select count(*) from "+tableName;
        ResultSet rs = execSQL(sql);
        try {
            if(rs.next()){
                result = rs.getInt(1);
            }
        } catch (SQLException e) {
            // TODO Auto-generated catch block
            e.printStackTrace();
        }finally{
            try {
                if(rs!=null)rs.close();
            } catch (SQLException e) {
                // TODO Auto-generated catch block
                e.printStackTrace();
            }
        }

        return result;
    }
```

Hotel.java

```
package team.hr.domain;

import java.util.List;

public class Hotel {
    private Integer hotelId;
    private String stree;
    private String country;
    private String state;
    private String zip;
    private String name;
    private List<String> rtype;
    private List<String> btype;
    private List<String> stype;
    public List<String> getRtype() {
        return rtype;
    }
    public void setRtype(List<String> rtype) {
        this.rtype = rtype;
    }
    public List<String> getBtype() {
        return btype;
    }
    public void setBtype(List<String> btype) {
        this.btype = btype;
    }
    public List<String> getStype() {
        return stype;
    }
    public void setStype(List<String> stype) {
        this.stype = stype;
    }
    public Hotel(){

    }
    public Hotel(Integer hotelId, String stree, String country, String
state, String zip, String name) {
        super();
        this.hotelId = hotelId;
        this.stree = stree;
        this.country = country;
        this.state = state;
        this.zip = zip;
        this.name = name;
    }
    public Integer getHotelId() {
        return hotelId;
    }
    public void setHotelId(Integer hotelId) {
        this.hotelId = hotelId;
    }
    public String getStree() {
        return stree;
    }
    public void setStree(String stree) {
        this.stree = stree;
    }
    public String getCountry() {
        return country;
    }
    public void setCountry(String country) {
        this.country = country;
    }
    public String getState() {
        return state;
    }
    public void setState(String state) {
        this.state = state;
    }
    public String getZip() {
        return zip;
```

```java
    }
    public void setZip(String zip) {
        this.zip = zip;
    }
    public String getName() {
        return name;
    }
    public void setName(String name) {
        this.name = name;
    }
}
```

Part of MyService.java

```java
public Integer findARoom(RoomReservation roomReservation) {
        Integer hotelId = roomReservation.getHotelId();
        ResultSet rs1 = myDao.queryRoomByHotelId(hotelId);
        Integer result = 0;
        try {
            while (rs1.next()) {
                Integer capacity = rs1.getInt(6);
                if(capacity<roomReservation.getCapacity()){
                    continue;
                }
                if(!roomReservation.getRtype().equals(rs1.getString(3))){
                    continue;
                }
                Integer roomNo = rs1.getInt(2);
                ResultSet rs2 = myDao.queryRoomReservation(roomNo,hotelId);
                boolean flag = true;
                while (rs2.next()) {
                    Date inDate = rs2.getDate(4);
                    Date outDate = rs2.getDate(5);

    if(!(roomReservation.getInDate().getTime()>=outDate.getTime()||roomRese
rvation.getOutDate().getTime()<=inDate.getTime())){
                        flag = false;
                        break;
                    }
                }
                if (flag) {
                    result = roomNo;
                    break;
                }
            }
        } catch (SQLException e) {
            e.printStackTrace();
        }finally{
            try {
                if(rs1!=null)rs1.close();
            } catch (SQLException e) {
                e.printStackTrace();
            }
        }
        return result;
    }

    public void pay(List<RoomReservation> list, Integer amount, String
cName, String cNumber) {
        Customer customer = myDao.queryCustomer(cName);
        Reservation reservation = new Reservation();
        String invoiceNo = MyUtil.getInvoiceNo();
        reservation.setInvoiceNo(invoiceNo);
        reservation.setcId(customer.getcId());
        reservation.setAmount(amount);
        reservation.setcNumber(cNumber);
        myDao.saveReservation(reservation);
        for (int i = 0; i < list.size(); i++) {
            RoomReservation r = list.get(i);
            myDao.saveRoomReservation(r, invoiceNo);
            for(int j=0;j<r.getBtype().size();j++){
                String b = r.getBtype().get(j);
                myDao.saveRresvBreakfast(r.getHotelId(), b, r.getRoomNo(),
r.getInDate(), r.getBamount().get(j));
            }
            for(int j=0;j<r.getStype().size();j++){
                String s = r.getStype().get(j);
                myDao.saveRresvService(r.getHotelId(), s, r.getRoomNo(),
r.getInDate());
            }
        }
    }
```

Reservation2.java

```
    protected void service(HttpServletRequest request, HttpServletResponse
response) throws ServletException, IOException {
        request.setCharacterEncoding("UTF-8");
        response.setContentType("text/html;charset=UTF-8");
        Integer hotelId =
Integer.parseInt(request.getParameter("hotelId"));
        MyService myService = new MyService();
        Hotel hotel = myService.getHotelDetail(hotelId);
        request.getSession().setAttribute("hotel", hotel);
        request.setAttribute("recommendation",
myService.getRecommendation());
        request.getRequestDispatcher("/WEB-
INF/MyPage/reservation2.jsp").forward(request, response);
    }

}
```

JSP and JS (Partial)

Part of reservation2.jsp

```
<script type="text/javascript">
        var btypeArray = new Array();
        function checkReservation() {


            var capacity = $("#capacity").val();
            var sum =0;
            var bamount = "";
            for(var i=0;i<btypeArray.length;i++){
                var number = btypeArray[i].split(":")[1] ;
                bamount +=number+"  ";
                sum =sum+parseInt(number);
            }

            $("#bamount").val(bamount);
            if (sum==0) {
                alert("Breakfast cannot be null");
                return;
            }
            if(sum>capacity){
                alert("No. of breakfast cannot more than guest number");
                return;
            }
            var inDate = document.getElementById("inDate");
            if(inDate.value!=""){
                if(!checkDate(inDate.value)){
                    return;
                }
            }
            else{
                alert("Check in date cannot be null");
                return;

            }
            var outDate = document.getElementById("outDate");
            if(outDate.value!=""){
                if(!checkDate(outDate.value)){
                    return;
                }
            }
            else{
                alert("Check out date cannot be null");
                return;
            }
            $("#btype").val(btypeArray.join());
            reservation_form.submit();
        }
        function checkDate(DateStr) {

            return true;
        }

        function add(){
             var btype = $('#typeSelected option:selected').val();
             var value = $('#bbamount').val();
             if(value == undefined ||value==""){
                 value=1;
             }
             btypeArray.push(btype+":"+value);
            $("#addB").after("<tr 
id='addTr'><td>"+btype+"</td><td>"+value+"</td></tr>");
        }
        function del(){
            if(btypeArray.length>0){
                btypeArray.splice(-1,1);
            }
            $("#addTr").remove();
        }
    </script>
```

Part of manager_login.jsp

```
<body>
    <h1>Secure Access Login</h1>
    <form name="login_form" action="./ManagerLogin" method="post">
        <table>
            <tr>
                <td><span class="require">*</span>Username: </td>
                <td><input type="text" id="name" name="name" size="30"
value="${requestScope.name}"
                    maxLength="30" /></td>
                <td><font color="red">${requestScope.userError}</font></td>
            </tr>
            <tr>
                <td><span class="require">*</span>Password: </td>
                <td><input type="password" id="password" name="password"
                    size="30" maxlength="30" /></td>
                <td><font
color="red">${requestScope.passwordError}</font></td>
            </tr>
            <tr>
                <td></td>
                <td><input type="button" value="Login"
onclick="checkLogin()" />&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;
                    <input type="submit" value="Register" name="mregister"
/></td>
            </tr>
        </table>
    </form>
    <script type="text/javascript">
        function checkLogin() {
            var name = document.getElementById("name");
            if (name.value == "") {
                alert("Please enter a valid username");
                name.focus();
                return;
            }
            var password = document.getElementById("password");
            if(password.value == ""){
                alert("Please enter your password");
                password.focus();
                return;
            }
```

```
           login_form.submit();
       }
   </script>
</body>
```