CSI 2132 – Database I

Project – e-Hotels

Part2

Presented

by

Saliha Sennad

And

Riley de Domenico

To

Verena Kantere

Introduction

Hotel database management system that consist of designing 5 hotel chains and 8 hotels for each of them. The hotel needs to maintain the history of the bookings and the reservations. The costumer should be able to reserve the available rooms according to their needs.

In part two of the project, we implemented the tables in PostgreSQL and inserted data into different tables. Also, a web application is created to simplify front desk office tasks, improve the experience of customers and guest reservations. With the software, a hotel can improve efficiency and effectively manage operations such as check-ins, check out and confirmation of reservations etc.

1. Report

Project requirements

The hotel reservation system should be able to satisfy the following requirements:

- Costumers should be able to search for available rooms through the application and make a reservation online
- The hotel employee should be able to transform the booking to renting at check_in
- A customer may present physically at a hotel without a booking and directly ask for a room where the employee of the hotel can do the renting of the room right away without prior booking
- Archive the bookings and the rentings
- We should be able to delete from our database hotel chains, hotels and rooms. We cannot have in the database information about a room without having in the database the information about the corresponding hotel (i.e. the hotel in which the room belongs too). In the same way, we cannot have in the database information about a hotel without having in the database the information about the corresponding hotel chain (i.e. the hotel chain in which the hotel belongs too).

a) DBMS

To develop application, we used:

1- PostgreSQL to implement the database
2- Java
3- Eclipse IDE
4- JAVA Script
5- Tomcat
6- HTML for the client side of the application

b) Steps to install the application

1. Download and Install Apache Tomcat Version 9
2. Perform basic setup of Apache Tomcat, however, set the HTTP port to 80 and set the AJP port to 89
3. Open up the Apache Software Foundation Folder within Program Files
4. Open the Tomcat 9.0 Folder then proceed to open the webapps folder
5. Move the Project.war file into the webapps folder
6. Turn on the Apache Tomcat servlet
7. Open a browser and type in http://localhost:80/Project
8. The web application should successfully open from here, if not, ensure that all ports are correctly set and that the servlet is on

c) SQL Queries

The following SQL statements used to create and modify schema objects. Insert, update delete data.

1- Hotel chains

```
-- Create Hotel_Chains Table
CREATE TABLE "Project_Hotel".Hotel_Chains(
   Hotel_Chain_Id INT PRIMARY KEY NOT NULL,
   Hotel_Chain_Name VARCHAR(30),
   Main_Office_Address VARCHAR(30),
   City VARCHAR(10),
   Province_State VARCHAR(10),
   Zip_Code VARCHAR(10),
   Country VARCHAR(10),
   Number_hotels INT
);
```

2- Hotels

```
-- Create Hotels tables
CREATE TABLE "Project_Hotel".Hotels (
  Hotel_Id INT PRIMARY KEY NOT NULL,
  Chain_Hotel_Id INT,
  Hotel_Name  VARCHAR(30),
  Hotel_Address  VARCHAR(30),
  City  VARCHAR(10),
  Province_State  VARCHAR(10),
  Zip_Code VARCHAR(10),
  Country  VARCHAR(10),
  Hotel_URL  VARCHAR(30),
  Number_Rooms INT,
  FOREIGN KEY (Chain_Hotel_Id )
  REFERENCES "Project_Hotel".Hotel_Chains
);
```

3- Hotel contacts

```
-- Create Hotel contacts
CREATE TABLE "Project_Hotel"."Hotel_Contacts" (
  "Contact_Id" INT PRIMARY KEY NOT NULL,
  "Phone_Number" VARCHAR(15),
  "Email" VARCHAR(20),
  "  Description" VARCHAR(255)
);
```

4- Chain contacts

```
-- Create Chain contacts
CREATE TABLE "Project_Hotel"."Chain_Contact" (
  "Contact_Id" INT PRIMARY KEY NOT NULL,
  "Phone_Number" VARCHAR(15),
  "Email" VARCHAR(20),
  "Description" VARCHAR(255)
);
```

5- Hotel ratings (Star rating)

```
-- Create hotel ratings
CREATE TABLE "Project_Hotel".Hotel_Ratings (
  Rating_Code INT PRIMARY KEY NOT NULL,
  Hotel_id INT,
  Star_Rating INT,
  FOREIGN KEY (hotel_id)
  REFERENCES "Project_Hotel".Hotels
);
```

6- Hotel rooms

```
--Create Table Hotel_Rooms
CREATE TABLE "Project_Hotel".Hotel_Rooms (
  Room_id INT PRIMARY KEY NOT NULL,
  Hotel_id INT,
  Size_Id INT,
  Mountain_View Boolean,
  Sea_View Boolean,
  TV Boolean,
  AC Boolean,
  Fridge Boolean,
  Room_Floor INT,
  Is_Smoking_Room Boolean,
  Availabiltty_Status Boolean,
  Nightly_Price Money,
  Remarks VARCHAR (255),
  Maximum_Capacity INT,
```

```
  Other VARCHAR (50),
  FOREIGN KEY (hotel_id)
  REFERENCES "Project_Hotel".Hotels,
  FOREIGN KEY (Size_id)
  REFERENCES "Project_Hotel".Room_Size
);
```

7- Room size

```
-- Create table Room_Size
CREATE TABLE "Project_Hotel".Room_Size (
  "Size_Id" INT PRIMARY KEY NOT NULL,
  "size_desc" VARCHAR (25)
);
```

8- Hotel Employees

```
--Create Table Employees
CREATE TABLE "Project_Hotel".Employees (
  Employee_Id INT PRIMARY KEY NOT NULL,
  Hotel_Id INT,
  Manager_Id INT,
  Employee_FName VARCHAR(15),
  Employee_LName VARCHAR(15),
  Phone_Numer VARCHAR(15),
  Joined_Date Date,
  FOREIGN KEY (hotel_id)
  REFERENCES "Project_Hotel".Hotels,
  FOREIGN KEY (Manager_Id)
  REFERENCES "Project_Hotel".Manager
);
```

9- Hotel manager

```
--Create Table Manager
CREATE TABLE "Project_Hotel".Manager (
   Manager_Id INT PRIMARY KEY NOT NULL,
   Hotel_Id INT,
   FOREIGN KEY (hotel_id)
   REFERENCES "Project_Hotel".Hotels
);
```

10- Employee role

```
--11-Create table employee Role
CREATE TABLE "Project_Hotel".Role (
  Role_Id INT PRIMARY KEY NOT NULL,
  Employee_Id INT,
  "Role" VARCHAR(30),
  FOREIGN KEY (Employee_Id)
  REFERENCES "Project_Hotel".Employees
);
```

11- Hotel customers

```
-- Create table Customers
CREATE TABLE "Project_Hotel".Customers (
  SSN VARCHAR(9) PRIMARY KEY NOT NULL,
  Customer_Name VARCHAR(15) ,
  Email1 VARCHAR(20),
  email2 VARCHAR(20),
  Customer_Address VARCHAR(30),
  Customer_City VARCHAR(20),
  Customer_Coutry VARCHAR(20),
  Customer_Contact_No VARCHAR(15),
  Cust_Registration_Date Date
);
```

12- Room renting's

```
--Create Table Rentings
CREATE TABLE "Project_Hotel".Rentings(
  Renting_Number INT PRIMARY KEY NOT NULL,
  Booking_Number INT,
  Employee_Id INT,
  SSN VARCHAR(9),
  Room_Id INT,
  CheckIn DATE,
  CheckOut DATE,
  Total_Price MONEY,
  FOREIGN KEY (Booking_Number)
  REFERENCES "Project_Hotel".Bookings,
  FOREIGN KEY (Room_id)
  REFERENCES "Project_Hotel".Hotel_Rooms,
  FOREIGN KEY (Employee_id)
  REFERENCES "Project_Hotel".Employees,

  FOREIGN KEY (SSN)
  REFERENCES "Project_Hotel".Customers
);
```

13- Room bookings

```
--Create Tabe Bookings
CREATE TABLE "Project_Hotel".Bookings (
  Booking_Number INT PRIMARY KEY NOT NULL,
  SSN VARCHAR(9),
  Room_Id INT,
  Bookind_Date DATE,
  Start_Date DATE,
  End_Date DATE,
  Total_Price MONEY,
  FOREIGN KEY (Room_id)
  REFERENCES "Project_Hotel".Hotel_Rooms,
  FOREIGN KEY (SSN)
  REFERENCES "Project_Hotel".Customers
);
```

14- Booking history

```
--Create table for rentings history
CREATE TABLE "Project_Hotel".Rentings_History (
  Renting_Number INT,
  Booking_Number INT,
  Room_Id INT,
  Check_in DATE,
  Check_out DATE,
  FOREIGN KEY (Renting_Number)
  REFERENCES "Project_Hotel".Rentings,
  FOREIGN KEY (Booking_Number)
  REFERENCES "Project_Hotel".Bookings,
  FOREIGN KEY (Room_Id)
  REFERENCES "Project_Hotel".Hotel_Rooms
);
```

15- Renting history

```
--Create table for rentings history
CREATE TABLE "Project_Hotel".Rentings_History (
  Renting_Number INT,
  Booking_Number INT,
  Room_Id INT,
  Check_in DATE,
  Check_out DATE,
  FOREIGN KEY (Renting_Number)
  REFERENCES "Project_Hotel".Rentings,
  FOREIGN KEY (Booking_Number)
  REFERENCES "Project_Hotel".Bookings,
  FOREIGN KEY (Room_Id)
  REFERENCES "Project_Hotel".Hotel_Rooms
);
```

16- Example of Inserting Data to tables

```
INSERT INTO "Project_Hotel".Customers
(
  SSN,Customer_Name,Email1,email2,Customer_Address,Customer_City
    , Customer_Country,customer_contact_no
    ,customer_registration_date
)
Values

(105760643,'Karen Carlisle','Karen.Carlisle@gmail.com','Carlisle_Karen@aol.com','834 Polo RD'
 ,'San Francisco','United States','504-845-1427','2015-06-16')
,(105767644,'Andrew Roberts','Andrew.Roberts@gmail.com','Roberts_Andrew@aol.com','230 saint_Michel'
  ,'Paris','France','810-374-9840','2015-06-16')
,(105781645,'Jill Fjeld','Jill.Fjeld@gmail.com','Fjeld_Jill@aol.com','12345 Albert St'
  ,'New York City','United States','856-264-4130','2015-06-16')
```

17- Example of updating a table

```
ALTER TABLE "Project_Hotel".Employees
    ADD CONSTRAINT fk_Emplyees_Role FOREIGN KEY (role_id)
    REFERENCES "Project_Hotel".role (role_id);
```

18- Example of adding the ON DELETE CASCADE

```
alter table "project_hotel".hotel_rooms
drop constraint hotel_rooms_hotel_id_fkey;

alter table "project_hotel".hotel_rooms
ADD CONSTRAINT hotel_rooms_hotel_id_fkey
foreign key (hotel_id)
references "project_hotel".hotels (hotel_id)
ON DELETE CASCADE;
```

2. Your SQL code that supports all the functionalities in your application

Two triggers are implemented to support the functionalities:

- To automatically insert the data from the renting tables after Insert or update in the renting table. That occurs when an employee changes the booking to renting at the time of check-in. Or when a customer walks in the hotel and rent directly on site.

```
CREATE TRIGGER TRIGGER_Rentings_History
AFTER INSERT ON "project_hotel".rentings
FOR EACH ROW
EXECUTE PROCEDURE logRentings();
```

```
CREATE OR REPLACE FUNCTION logRentings()
RETURNS trigger AS

$BODY$ BEGIN
IF(EXISTS(SELECT renting_number FROM "project_hotel".rentings  WHERE renting_number IS NOT NULL)) THEN
    INSERT INTO "project_hotel".rentings_history(renting_number, booking_number, room_id,checkin,checkout)
    SELECT renting_number,booking_number,room_id,checkin,checkout FROM project_hotel.rentings;
RETURN NULL;
END IF;
END;$BODY$
Language 'plpgsql'
```

- To automatically shows the available rooms based on a certain Province_State or city.

```
CREATE TRIGGER TRIGGER_Available_rooms
AFTER UPDATE ON "project_hotel".hotel_rooms

EXECUTE PROCEDURE Available_Rooms();
```

```
CREATE OR REPLACE FUNCTION Available_Rooms()
RETURNS trigger AS

$BODY$ BEGIN

    SELECT h.Province_state, h.city, count(r.room_id) as Number_of_Available_Rooms
    FROM "project_hotel".hotel_rooms r
    JOIN "project_hotel".hotels h ON h.hotel_id = r.hotel_id
    WHERE availabiltty_status is true
    GROUP BY Province_state, h.city;

END;$BODY$
Language 'plpgsql'
```

3. **All the code that is necessary for running your application**
   See Zip file