Principle of Database Management – Final Project Report

**Topic: 23 – Restaurant Management System**

**Team Members:**

Le Do Huy Du – ITITIU18027 – team member

**About database system and application used:**

NetBeans IDE 8.0.2 for Java codes

SQL Server 2014 Management Studio for creating database

**About submission file:**

**INTRODUCTION:**

In this report, we will show all the step and progress of developing a Hotel Management System with SQL and Java language, from the planning to the program itself.

The Hotel Management System will help improve functionality and efficiency of works and transaction in any Hotel system.

**ANALYSIS:**

- Customers that want to make a reservation in the hotel can access to book via online website or at the hotel’s reception desk

- Customers must provide basic information for security reasons.

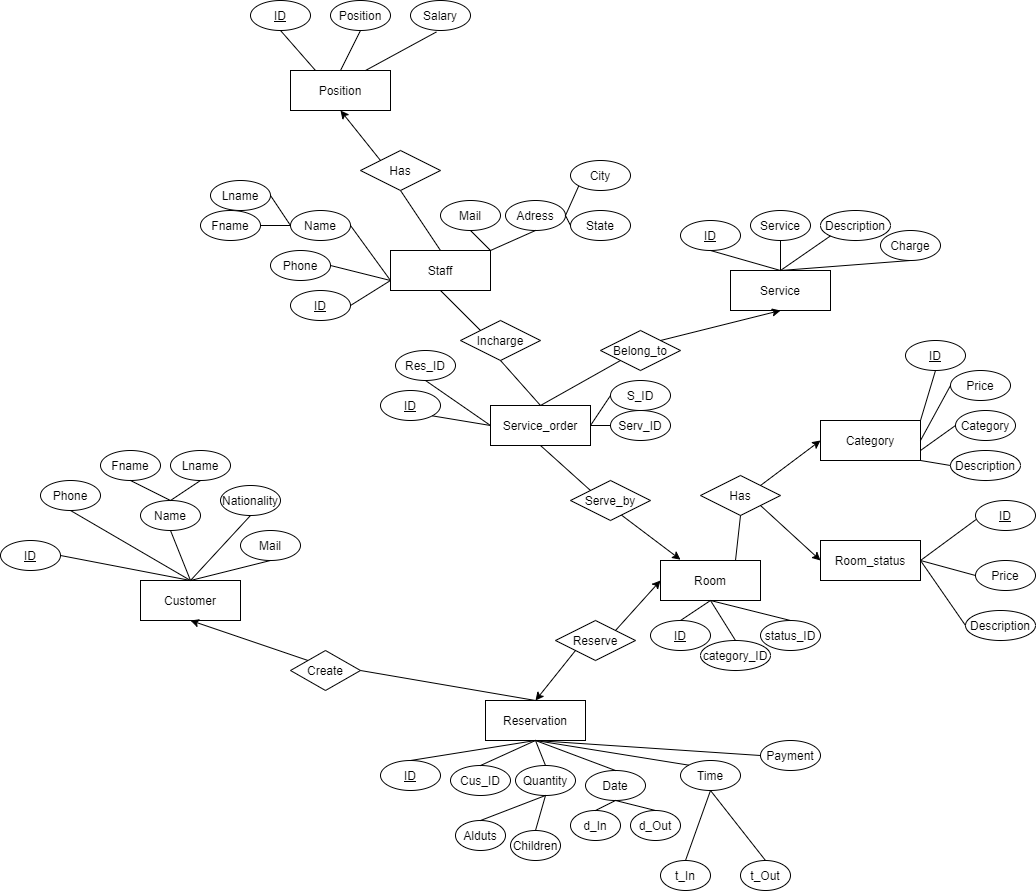
- a reservation must have the booking room, the temporary ID for the customer, date and time of stay, number of people and the total cost.

- Rooms of hotel consisted of many categories with various prices for the customers to choose.

- While staying at the hotel, customers can pay for extra services provided by the hotel facilities and staffs, the cost will be added into the total cost in the reservation form on computer.

- Each services will be recorded as a service order with the service type, the employee providing the service and the customer demanding the service

- Each staff personal information and position also stored in the system for security, service management and salary payment

ENTITY RELATIONSHIP DIAGRAM – CONVERT TO RELATIONAL MODEL

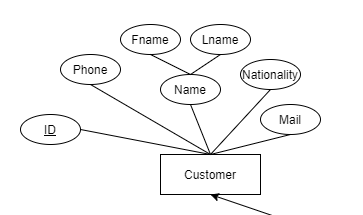
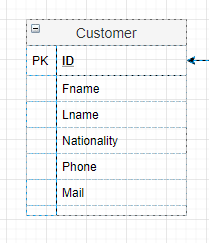
In this diagram, we can see there are total 9 strong entities:

Customer, Reservation, Room, Room\_status, Category, Service\_order, Service, Staff and Position

For these 9 entities we will have 9 tables with the same name

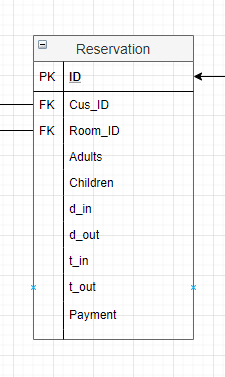
Let us convert all the entities into tables, one by one:

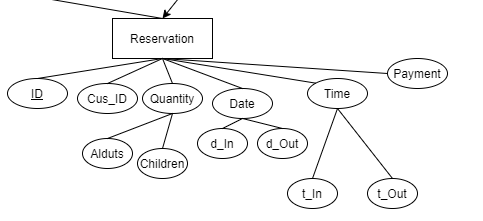
**1.Customer**



Schema: Customer (ID, Fname, Lname, Nationality, Phone, Mail)

This table contain the needed information of the customers



**2.Reservation**

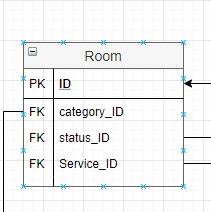
Schema: Reservatiom (ID, Cus\_ID, Room\_ID, Adults, Children, d\_in, d\_out, t\_in, t\_ out, Payment)

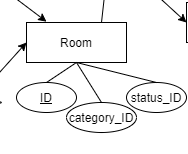
Cus\_ID is a foreign key with references to Customer (ID)

Room\_ID is a foreign key with references to Room (ID)

As a reservation must have the information of a customer and the room he/she has reserved

\*each customer can make many reservations

**3.Room**



Schema: Room (ID, category\_ID, Status\_ID, Service\_ID)

Category\_ID is a foreign key with references to Category (ID)

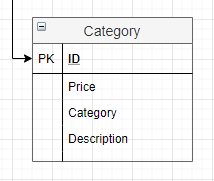
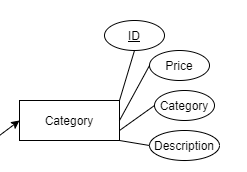
Status\_ID is a foreign key with references to Room\_status (ID)

As there are many rooms belong to different categories and each room has their own status

Service\_ID is a foreign key with references to Service\_order (ID)

As the customer from a specific room can demand different services

**4.Category**

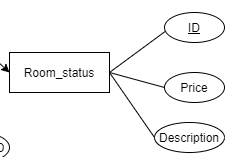
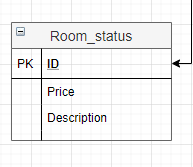


Schema: Category (ID, Price, Category, Description)

This table contain the price and description of room categories

\*each category can have many rooms and each room can only belong to one category

**5.Room\_status**

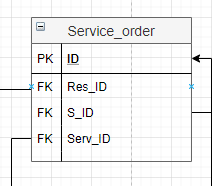


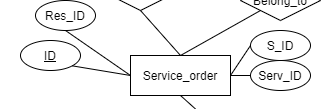
Schema: Room\_status(ID, Price, Description)

This table contain the description of room statuses and the prices that the room status can affect beside the room category

\*each room can only have one status at a time, the status will be frequently updated

**6.Service\_order**





Schema: Service\_order (ID, Res\_ID, S\_ID, Serv\_ID)

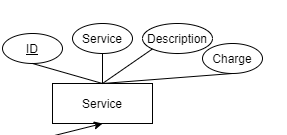
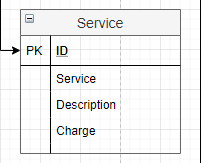
Res\_ID is a foreign key with references to Reservation (ID) which indicate the customer or room of the reservation that demanded the service

S\_ID is a foreign key with references to Staff (ID) that will specify the staff who will do the service

Serv\_ID is a foreign key with references to Service (ID) that indicate the type of service that need to be done

\*each service order can have one service each and one staff do the servicing

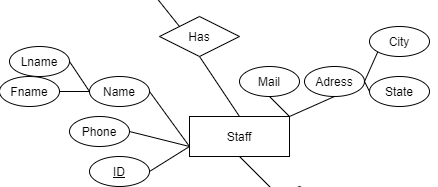
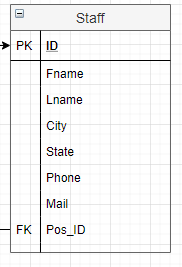
**7.Service**



Schema: Service (ID, Service, Description, Charge)

This table contain information about types of service, their description and the charge

**8.Staff**

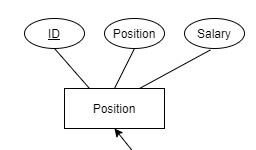
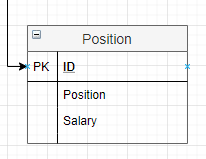


Schema: Staff (ID, Fname, Lname, City, State, Phone, Mail, Pos\_ID)

Pos\_ID is a foreign key with references to Position (ID) which indicate the job and position of the employee in the Hotel system

Each staff member can only have one position, this can be update

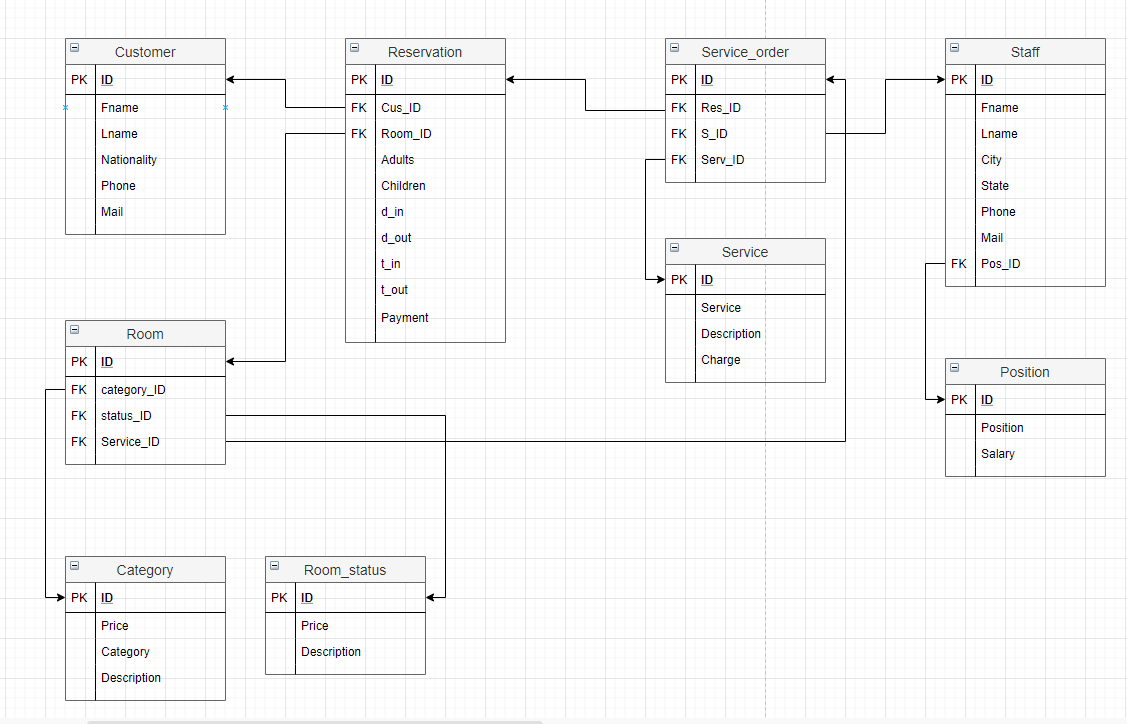
**9.Position**



Schema: Position (ID, Position, Salary)

This table contain the position and the Salary of each job

**The Relational model:**



FINAL RESULT:



We can redefine the Model above as tuples:

**Customer** (ID, Fname, Lname, Nationality, Phone, Mail)

**Reservation** (ID, Cus\_ID, Room\_ID, Adults, Children, d\_in, d\_out,t\_in, t\_out, Payment)

**Room** (ID, category\_ID, status\_ID, Service\_ID)

**Cateogry** (ID, Price, Category, Description)

**Room**\_status (ID, Price, Description)

**Service**\_order (ID, Res\_ID, S\_ID, Serv\_ID)

**Service** (ID, Description, Charge)

**Staff** (ID, Fname, Lname, City, State, Phone, Mail, Pos\_ID)

**Position** (ID, Position, Salary)

**MAKING THE PROGRAM**

Base on the Relational Model, we can create a SQL databse with SQL Server.

We will not include the “insert data” steps along with creating tables

**1.Create Customer Table**

CREATE TABLE IF NOT EXISTS "Customer" (

"ID" INTEGER NOT NULL PRIMARY KEY AUTOINCREMENT UNIQUE,

"First\_name" VARCHAR(50) NOT NULL,

"Last\_name" VARCHAR(50) NOT NULL,

"Gender" TEXT NOT NULL,

"Phone" INTEGER NOT NULL,

"Email" TEXT

);

**2.Create Reservation Table**

CREATE TABLE IF NOT EXISTS "Reservation" (

"ID" INTEGER PRIMARY KEY AUTOINCREMENT,

"Customer\_ID" INTEGER,

"Adults" INTEGER,

"Children" INTEGER,

"Room\_ID" INTEGER,

"Check\_in" TIME,

"Day\_in" DATE,

"Check\_out" TIME,

"Day\_out" DATE,

"Reservation\_dat" DATE,

"Payment" INTEGER,

CONSTRAINT "fk\_customer" FOREIGN KEY("Customer\_ID") REFERENCES "Customer"("ID"),

CONSTRAINT "fk\_room\_id\_reserve" FOREIGN KEY("Room\_ID") REFERENCES "room"("ID")

);

**3.Create Room Table**

CREATE TABLE IF NOT EXISTS "Room" (

"ID" INTEGER NOT NULL,

"category\_id" INTEGER NOT NULL,

"status\_id" INTEGER NOT NULL,

CONSTRAINT "fk\_category" FOREIGN KEY("category\_id") REFERENCES "category"("ID"),

CONSTRAINT "fk\_status" FOREIGN KEY("status\_id") REFERENCES "room\_status"("ID"),

PRIMARY KEY("ID");

**4.Create Room Status Table**

CREATE TABLE IF NOT EXISTS "category" (

"ID" INTEGER NOT NULL,

"category" TEXT NOT NULL,

"price" INTEGER NOT NULL,

"description" TEXT,

PRIMARY KEY("ID")

);

**5.Create Category Table**

CREATE TABLE IF NOT EXISTS "category" (

"ID" INTEGER NOT NULL,

"category" TEXT NOT NULL,

"price" INTEGER NOT NULL,

"description" TEXT,

PRIMARY KEY("ID")

);

**6.Create Service Table**

CREATE TABLE IF NOT EXISTS "Service" (

"ID" TEXT NOT NULL,

"Service" TEXT NOT NULL,

"Description" TEXT NOT NULL,

"Charge" INTEGER NOT NULL,

PRIMARY KEY("ID")

);

**7.Create Service order Table**

CREATE TABLE IF NOT EXISTS "Service\_order" (

"ID" INTEGER NOT NULL,

"Res\_ID" INT NOT NULL,

"Service\_ID" TEXT NOT NULL,

PRIMARY KEY("ID" AUTOINCREMENT),

CONSTRAINT "fk\_Services" FOREIGN KEY("Service\_ID") REFERENCES "Service"("ID"),

CONSTRAINT "fk\_reservation\_id\_order" FOREIGN KEY("Res\_ID") REFERENCES "Reservation"("ID")

);

**8.Create Staff Table**

CREATE TABLE IF NOT EXISTS "Staff" (

"ID" INTEGER NOT NULL,

"first\_name" TEXT,

"last\_name" TEXT,

"Gender" TEXT NOT NULL,

"Mail" TEXT,

"Phone\_number" TEXT,

"Address" TEXT,

"Position\_ID" INTEGER,

FOREIGN KEY("Position\_ID") REFERENCES "Position\_ID"("ID"),

PRIMARY KEY("ID");

**9,Create Position\_ID Table**

CREATE TABLE IF NOT EXISTS "Position\_ID" (

"ID" INTEGER NOT NULL,

"Position" TEXT NOT NULL,

"Salary" INTEGER NOT NULL,

PRIMARY KEY("ID")

);

**JAVA CODE**

The Java codes are long so we will only name the Classes and the functionality of each class

…

…

…

**PROGRAM DEMO**

**1.customer/staff login**

**2.checking/searching customer information**

**3.make a reservation**

**4.checking room status**

**5.provide/record a service**

**6.making bills**

**7.delete/update customer information**

**8.update room status**

**9.delete/update reservation info**

**10.delete/update staff info**

**CONCLUSION**