Hotel Booking System Database

**Computer Systems Structure**

# **Task 1 – Entity Relationship Model**

# **SCENARIO**

# The structure and implementation of a Database Management System for the purpose of hotel booking. The database will store customer data, hotel manager data, booking information and room information.

# Description automatically generated

Figure 1. Entity Relationship Model (ERM) for the Hotel Booking System

# **Task 2 – Normalisation**

|  |  |  |  |
| --- | --- | --- | --- |
| **UNF** | **1NF** | **2NF** | **3NF** |
| **CustomerID**  CustomerName  Address  PhoneNumber  CustomerEmail  CustomerPassword  CCNumber  CCExpiry  CCCVV  CCType  BookingID  StartDate  EndDate  RoomType  BookingStatus  BookingDate  StaffID  HotelManName  HotelManEmail  HotelManPassword  RoomNo  RoomType | **CustomerID**  CustomerName  Address  PhoneNumber  CustomerEmail  CustomerPassword  CCNumber  CCExpiry  CCCVV  CCType  **BookingID**  StartDate  EndDate  RoomType  BookingStatus  BookingDate  StaffID  HotelManName  HotelManEmail  HotelManPassword  RoomNo  RoomType  CustomerID\* | **CustomerID**  CustomerName  Address  PhoneNumber  CustomerEmail  CustomerPassword  CCNumber  CCExpiry  CCCVV  CCType  **BookingID**  StartDate  EndDate  RoomType  BookingStatus  BookingDate  StaffID  HotelManName  HotelManEmail  HotelManPassword  RoomNo  RoomType  CustomerID\* | **CustomerID**  CustomerName  Address  PhoneNumber  CustomerEmail  CustomerPassword  CCNumber  CCExpiry  CCCVV  CCType  **BookingID**  StartDate  EndDate  RequestedRoomType  BookingStatus  BookingDate  CustomerID\*  StaffID\*  RoomNo\*  **StaffID**  HotelManName  HotelManEmail  HotelManPassword  **RoomNo**  RoomType |

**Unnormalised form(UNF) to first normal form(1NF)** involves splitting the single and repeating groups. A single customer may make many bookings which allows data to repeat.

A table is in **second normal form(2NF)** if it is in 1NF and all non-key attributes are dependent on part of the key. Therefore, it is automatically in 2NF due to the tables having a single functionally dependent attribute.

A table is in **third normal form(3NF)** when the non-key attributes that depend on other non-key(s) are removed. For instance, Hotel Manager Name (HotelManName) depends on StaffID therefore it must be separated.

# **Task 3 – Physical Table Design (including Data Dictionary)**

**Part A – Skeleton Tables**

**Customer** (CustomerID, CustomerName, Address, PhoneNumber, CustomerEmail, CustomerPassword, CCNumber, CCExpiry, CCCVV, CCType)

**Booking** (BookingID, CustomerID\*, StaffID\*, RoomNumber\*, StartDate, EndDate, RequestedRoomType, BookingStatus, BookingDate)

**Hotel Manager** (StaffID, HotelManName, HotelManEmail, HotelManPass)

**Room** (RoomNumber, RoomType)

**Part B – Data Dictionary**

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Customer |  |  |  |  |  |  |  |
| Description:  Customer details |  |  |  |  |  |  |  |
| Field Name | Data Type | Length | Index | Null | Default Value | Validation rule | Description |
| CustomerID (Primary) | varchar(5) | 5 | PK | No |  |  | Unique number ID for each customer |
| Name | varchar(30) | 30 |  | No |  |  | Name for customer |
| Address | varchar(99) | 99 |  | No |  |  | Address for customer |
| PhoneNumber | varchar(15) | 15 |  |  |  |  | Contact number for customer |
| Email | varchar(30) | 30 |  | No |  | Must be email format containing an @ and a ‘.’ | Customer’s registered email address |
| Password | varchar(30) | 30 |  | No |  |  | Customer’s password |
| CCNumber | integer(16) | 16 |  | Yes |  |  | Customer’s credit card number |
| CCExpiry | varchar(5) | 5 |  | Yes |  |  | Expiry date for customer’s credit card |
| CCCVV | varchar(3) | 3 |  | Yes |  |  | Card verification value for customer’s credit card |
| CCType | varchar(15) | 15 |  | Yes |  |  | Type of credit card for customer |

Indexes

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Keyname** | **Type** | **Unique** | **Column** | **Null** |
| PRIMARY | BTREE | Yes | CustomerID | No |

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Booking |  |  |  |  |  |  |  |
| Description:  Booking details |  |  |  |  |  |  |  |
| Field Name | Data Type | Length | Index | Null | Default Value | Validation rule | Description |
| BookingID (Primary) | integer |  | PK | No |  |  | Autoincremented unique number ID for every booking |
| StartDate | date |  |  | No |  | Cannot be a past date | Date the room’s booking begins |
| EndDate | date |  |  | No |  | Must be a date after the start date | Date the room’s booking ends |
| RoomType | varchar(20) | 20 |  | No |  | Single, Double | The type of room to be requested |
| Booking\_Status | varchar(20) | 20 |  | Yes |  | Pending, Active, Completed, Cancelled | The status of the booking |
| BookingDate | date |  |  | Yes | sysdate |  | Date the room is booked |
| StaffID | varchar(8) | 8 | FK | Yes |  |  | Hotel manager who assigns a room to the booking |
| RoomNo | integer | 3 | FK | Yes |  |  | The room assigned to the booking |
| CustomerID | integer | 8 | FK | Yes |  |  | Customer who creates the booking |

Indexes

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Keyname** | **Type** | **Unique** | **Column** | **Null** |
| PRIMARY | BTREE | Yes | BookingID | No |

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Hotel Manager |  |  |  |  |  |  |  |
| Description:  Hotel manager details |  |  |  |  |  |  |  |
| Field Name | Data Type | Length | Index | Null | Default Value | Validation rule | Description |
| StaffID (Primary) | varchar(5) | 5 | PK | No |  |  | Unique number ID for each hotel manager |
| Name | varchar(30) | 30 |  | No |  |  | Name for hotel manager |
| Email | varchar(30) | 30 |  | No |  | Must be email format containing an @ and a ‘.’ | Hotel manager’s registered email address |
| Password | varchar(15) | 15 |  | No |  |  | Hotel manager’s password |

Indexes

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Keyname** | **Type** | **Unique** | **Column** | **Null** |
| PRIMARY | BTREE | Yes | StaffID | No |

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Room |  |  |  |  |  |  |  |
| Description:  Room details |  |  |  |  |  |  |  |
| Field Name | Data Type | Length | Index | Null | Default Value | Validation rule | Description |
| RoomNumber (Primary) | Int(5) | 5 | PK | No |  |  | Unique number for each room |
| RoomType | varchar(20) | 20 |  | No |  | Single, Double | The type of room |

Indexes

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Keyname** | **Type** | **Unique** | **Column** | **Null** |
| PRIMARY | BTREE | Yes | RoomNumber | No |

# **Task 4 – Query Design**

|  |  |  |
| --- | --- | --- |
| Query | Purpose | SQL Code |
| 1.List of all TT (Trinidad & Tobago) Customers | The hotel may require a list of customers from a specific country for the purposes of marketing. | **SELECT** c.Customer\_ID, c.Name, c.Address, c.Phone\_Number, c.Email  **FROM** Customer c  **WHERE** c.Address **LIKE** '%TT'  **ORDER BY** c.Name asc; |
| 2.Highest booked room type | The hotel may be interested in its most successfully booked room types. | **SELECT** b.Room\_Type, COUNT(\*) AS TypeCount  **FROM** Booking b  **WHERE** Booking\_Status <> 'Active' OR 'Completed'  **GROUP BY** b.Room\_Type  **ORDER BY** TypeCount desc; |
| 3.List of bookings not assigned to a room | The hotel manager may need to filter and view pending bookings to assign an appropriate room. This will be sorted by booking date. | **SELECT** b.Booking\_ID, b.Booking\_Status, b.Booking\_Date, b.Start\_Date, b.End\_Date, b.Room\_Type,  c.Customer\_ID, c.Name, c.Email, c.Phone\_Number  **FROM** Booking b, Customer c  **WHERE** c.Customer\_ID = b.Customer\_ID  **AND** b.Room\_Number is NULL  **GROUP BY** b.Booking\_Date |
| 4.Available rooms on a certain day | The hotel manager may need to filter and view available rooms on a certain date for booking assignment and approval purposes. | **SELECT** r.Room\_Number, r.Room\_Type  **FROM** Room r  **WHERE** r.Room\_Number NOT IN  (SELECT b.Room\_Number  **FROM** Booking b  **WHERE** '2020-11-27' BETWEEN b.Start\_Date AND b.End\_Date  **AND** b.Booking\_Status <> 'Cancelled') |
| 5.The customer with the most bookings and corresponding Hotel Manager that assigned the booking | The hotel may want to know their top customer as well as the hotel manager who assigned the booking for the purposes of record keeping. | **SELECT** c.Customer\_ID, c.Name, COUNT(\*) AS BookingCount, h.Staff\_ID, h.Name AS HotelManName  **FROM** Customer c, Booking b, HotelManager h  **WHERE** c.Customer\_ID = b.Customer\_ID  **AND** h.Staff\_ID = b.Staff\_ID  **AND** Booking\_Status <> 'Cancelled'  **GROUP BY** c.Customer\_ID, c.Name  **ORDER BY** COUNT(\*) desc |

# **Task 5 – Implementation of the Database**

**Part A – Creating Tables**

**CREATE TABLE** Customer (Customer\_ID VARCHAR(5) **PRIMARY KEY** NOT NULL, Name VARCHAR(30) NOT NULL, Address VARCHAR(99) NOT NULL, Phone\_Number VARCHAR(15) NOT NULL UNIQUE, Email VARCHAR(30) NOT NULL UNIQUE, Password VARCHAR(30) NOT NULL, Creditcard\_Number INT(16), Creditcard\_Expiry VARCHAR(5), Creditcard\_CVV VARCHAR(3), Creditcard\_Type VARCHAR(15));

**Graphical user interface

Description automatically generated**

**CREATE TABLE** Hotel Manager (Staff\_ID VARCHAR(5) **PRIMARY KEY**, Name VARCHAR(30) NOT NULL, Email VARCHAR(30) NOT NULL UNIQUE, Password VARCHAR(30) NOT NULL);

**Graphical user interface, application

Description automatically generated**

**CREATE TABLE** Room (Room\_Number INT(5) **PRIMARY KEY**, Room\_Type VARCHAR(20) NOT NULL);

Graphical user interface, text, application, email

Description automatically generated

**CREATE TABLE** Booking (Booking\_ID INTEGER **PRIMARY KEY** AUTOINCREMENT, Start\_Date DATE NOT NULL, End\_Date DATE NOT NULL, Room\_Type VARCHAR(20) NOT NULL, Booking\_Status VARCHAR(20), Booking\_Date DATE DEFAULT sysdate, Customer\_ID REFERENCES Customer(Customer\_ID), Staff\_ID REFERENCES HotelManager(Staff\_ID), Room\_Number REFERENCES Room(Room\_Number));

A picture containing text

Description automatically generated

**Part B – Inserting Data**

**INSERT INTO** Customer (Customer\_ID, Name, Address, Phone\_Number, Email, Password, Creditcard\_Number, Creditcard\_Expiry, Creditcard\_CVV, Creditcard\_Type) VALUES ('C1111', 'Varoun Sankar', '77 Valentinos St, Night City', '1(868)612-2077', 'vpunk@email.com', 'cyberpunk2077', '2077-2077-2077-2077', '12/77', '277', 'VISA');

**INSERT INTO** Customer (Customer\_ID, Name, Address, Phone\_Number, Email, Password, Creditcard\_Number, Creditcard\_Expiry, Creditcard\_CVV, Creditcard\_Type) VALUES ('C2222', 'Kim Kali', '22 Bush St, Fyzabad', '1(868)610-2000', 'kimki@email.com', 'celestine', '2000-2000-2000-2000', '04/22', '200', 'MASTERCARD');

Graphical user interface, text, application, email

Description automatically generated

**INSERT INTO** HotelManager (Staff\_ID, Name, Email, Password) VALUES (‘H1087’, ‘Leo Messiah’, ‘lmessiah@hoteldb.com’, ‘thegoat10’);

**INSERT INTO** HotelManager (Staff\_ID, Name, Email, Password) VALUES (‘H1986’, ‘Drake Graham’, ‘dgraham@hoteldb.com’, ‘TakeCare11’);Graphical user interface, text, application, email

Description automatically generated

**INSERT INTO** Room (Room\_Number, Room\_Type) VALUES ('411', ‘Single');

**INSERT INTO** Room (Room\_Number, Room\_Type) VALUES ('320', ‘Double');

Graphical user interface, text, application

Description automatically generated

**INSERT INTO** Booking (Booking\_ID, Customer\_ID, Staff\_ID, Room\_Number, Start\_Date, End\_Date, Room\_Type, Booking\_Status, Booking\_Date) VALUES (1, 'C1111', 'H1087', 411, '2020-11-25', '2020-11-27', 'Single', 'Active', '2020-10-27');

**INSERT INTO** Booking

(Booking\_ID, Customer\_ID, Staff\_ID, Room\_Number, Start\_Date, End\_Date, Room\_Type, Booking\_Status, Booking\_Date) VALUES (2, 'C2222', 'H1986', 320, '2020-10-20', '2020-10-22', 'Double', 'Cancelled', '2020-10-01');

Text

Description automatically generated

**Part C – SQL Queries**

In reference to Task 4 – Query Design

**1.List of all TT (Trinidad & Tobago) Customers**

Graphical user interface, text, application, email

Description automatically generated

**2.Highest booked room type**

**Graphical user interface, text, application

Description automatically generated**

**3.List of bookings not assigned to a room**

**A picture containing table

Description automatically generated**

**4.Available rooms on a certain day**

**Graphical user interface, text, application, email

Description automatically generated**

**5.The customer with the most bookings and corresponding Hotel Manager that assigned the booking**

A picture containing table

Description automatically generated

**Discussion and Reflection**

Finishing this assignment truly familiarised me with understanding and utilising a database system.

It involved drawing and understanding an Entity Relationship Model, normalising and going into further detail with its entities and attributes by physical table design. Understanding those tasks were key in utilising Structured Query Language (SQLite Studio) to create the database.

The task at hand ultimately involved a hotel seeking to provide a database system for customers to book either single or double rooms and checking their upcoming trips. After a booking is made, hotel managers would assign an appropriate room to each booking.

For the first task, an Entity Relationship Model was drawn up involving the four main entities which were Customer, Booking, Hotel Manager and Room. Relevant attributes and tables were assigned and normalised to remove anomalies and bring the database to a consistent state. The physical table design was finalised through skeleton tables and data dictionaries which provided detailed information about the dataset.

The meat of the assignment then dealt with query design and implementation of the database, bringing the system to life.

There were several hiccups involving errors. One, in relation to inserting foreign keys, made me lose a good bit of time in deducing but it was a price that needed to be paid for the sake of learning. The data also seemed prone to human error and might be the main cause of faults within data structures.

Nevertheless, this assignment was a necessary steppingstone to the world of Computer Systems Structure.