Hotel Management System Case Study

An Overview of Database Schema,Data and Queries

Objective

- To Manage and analyze hotel operations using SQL.

Database Schema Overview

- Tables Created:
  - Guests
  - Rooms
  - Bookings
  - Services
  - Booking Services
  - Payments
- Relationships:Foreign Key linking related tables.

- **Guests Table**

➢ Purpose:Stores Guest Information.

❖ Columns:

1. GuestID(Primary Key)
2. Firstname
3. Lastname
4. Email
5. Phone

```
create database h_m_s;
use h_m_s;
Create Table Guests(
    GuestID int primary key,
    FirstName varchar(50),
    LastName varchar(50),
    Email varchar(100),
    Phone varchar(20));
```

```
insert into Guests(GuestID, FirstName, LastName, Email, Phone)
values(1, 'Daniel', 'Radcliffe', 'daniel.r@example.com', '555-1234'),
(2, 'Emma', 'Watson', 'emma.w@example.com', '222-5678'),
(3, 'Bonnie', 'Wright', 'bonnie.w@example.com', '555-8765'),
(4, 'Rupert', 'Grint', 'rupert.g@example.com', '555-4321'),
(5, 'Evanna', 'Lynch', 'evanna.l@example.com', '555-3456'),
(6, 'Tom', 'Felton', 'tom.f@example.com', '444-6543'),
(7, 'Maggie', 'Smith', 'maggie.s@example.com', '555-9876'),
(8, 'Matthew', 'Lewis', 'matthew.l@example.com', '555-5432'),
(9, 'Katie', 'Leung', 'katie.l@example.com', '555-2345'),
(10, 'Robert', 'Pattinson', 'robert.p@example.com', '555-6789');
```

Guest Table:

```
select * from Guests;
```

Result Grid | Filter Rows: | Edit: | Export/Import: | Wrap Cell Content:

GuestID	FirstName	LastName	Email	Phone
1	Daniel	Radcliffe	daniel.r@example.com	555-1234
2	Emma	Watson	emma.w@example.com	222-5678
3	Bonnie	Wright	bonnie.w@example.com	555-8765
4	Rupert	Grint	rupert.g@example.com	555-4321
5	Evanna	Lynch	evanna.l@example.com	555-3456
6	Tom	Felton	tom.f@example.com	444-6543
7	Maggie	Smith	maggie.s@example.com	555-9876
8	Matthew	Lewis	matthew.l@example.com	555-5432
9	Katie	Leung	katie.l@example.com	555-2345
10	Robert	Pattinson	robert.p@example.com	555-6789
NULL	NULL	NULL	NULL	NULL

- **Rooms Table**

➢ Purpose:Stores Room Details

❖ Columns:

1. RoomID(Primary Key)
2. RoomNumber
3. RoomType
4. PricePerNight

```
Create Table Rooms(
    RoomID int primary key,
    RoomNumber varchar(10),
    RoomType varchar(50),
    PricePerNight decimal(10, 2));
```

```
insert into Rooms(RoomID, RoomNumber, RoomType, PricePerNight)
values(1, '101', 'Single', 100),
(2, '102', 'Double', 150),
(3, '201', 'Suite', 250),
(4, '202', 'Suite', 275),
(5, '301', 'Single', 100),
(6, '302', 'Double', 160),
(7, '401', 'Suite', 300),
(8, '402', 'Single', 110),
(9, '501', 'Double', 140),
(10, '502', 'Suite', 320);
```

Rooms Table:

```
Select * from Rooms;
```

Result Grid | Filter Rows: | Edit: | Export/Import: | Wrap Cell Content:

RoomID	RoomNumber	RoomType	PricePerNight
1	101	Single	100.00
2	102	Double	150.00
3	201	Suite	250.00
4	202	Suite	275.00
5	301	Single	100.00
6	302	Double	160.00
7	401	Suite	300.00
8	402	Single	110.00
9	501	Double	140.00
10	502	Suite	320.00
NULL	NULL	NULL	NULL

- **Bookings Table:**

➢ Purpose:Records bookings made by guests.

❖ Columns

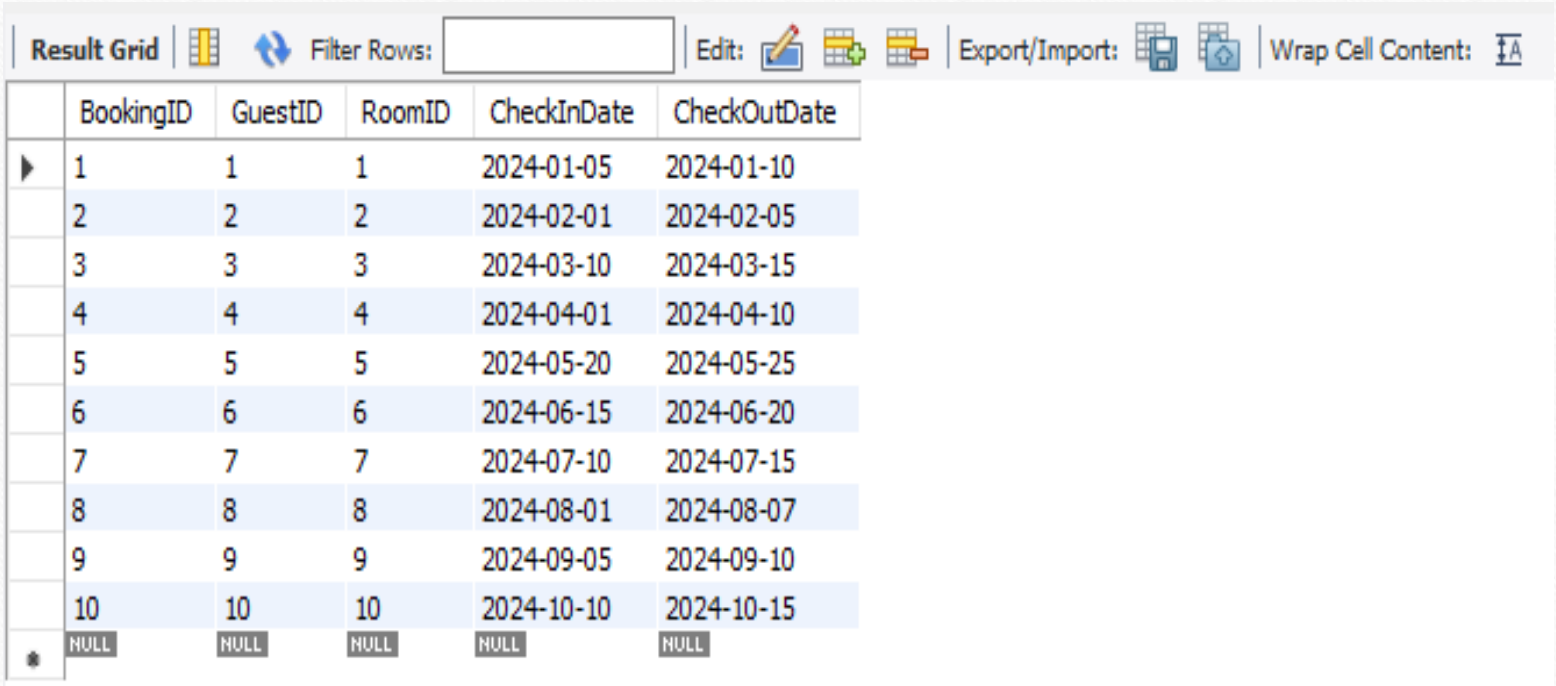
1. BookingID(Primary Key)
2. GuestID(Foreign Key)
3. RoomID(Foreign Key)
4. CheckInDate
5. CheckOutDate

```
Create Table Bookings(
    BookingID int primary key,
    GuestID int,
    RoomID int,
    CheckInDate date,
    CheckOutDate date,
    Foreign Key (GuestID) References Guests(GuestID),
    Foreign Key (RoomID) References Rooms(RoomID));
```

```
insert into Bookings(BookingID, GuestID, RoomID, CheckInDate, CheckOutDate)
values(1, 1, 1, '2024-01-05', '2024-01-10'),
(2, 2, 2, '2024-02-01', '2024-02-05'),
(3, 3, 3, '2024-03-10', '2024-03-15'),
(4, 4, 4, '2024-04-01', '2024-04-10'),
(5, 5, 5, '2024-05-20', '2024-05-25'),
(6, 6, 6, '2024-06-15', '2024-06-20'),
(7, 7, 7, '2024-07-10', '2024-07-15'),
(8, 8, 8, '2024-08-01', '2024-08-07'),
(9, 9, 9, '2024-09-05', '2024-09-10'),
(10, 10, 10, '2024-10-10', '2024-10-15');
```

Bookings Table:

```
select * from Bookings;
```



Result Grid | Filter Rows: | Edit: | Export/Import: | Wrap Cell Content:

BookingID	GuestID	RoomID	CheckInDate	CheckOutDate
1	1	1	2024-01-05	2024-01-10
2	2	2	2024-02-01	2024-02-05
3	3	3	2024-03-10	2024-03-15
4	4	4	2024-04-01	2024-04-10
5	5	5	2024-05-20	2024-05-25
6	6	6	2024-06-15	2024-06-20
7	7	7	2024-07-10	2024-07-15
8	8	8	2024-08-01	2024-08-07
9	9	9	2024-09-05	2024-09-10
10	10	10	2024-10-10	2024-10-15
NULL	NULL	NULL	NULL	NULL

- **Services Table:**

➢ Purpose:Tracks Additional services used by guests during their stay.

❖ Columns:

1. ServiceID(Primary Key)
2. Service Name
3. Price

```
Create Table Services(
    ServiceID int primary Key,
    ServiceName varchar(100),
    Price decimal(10, 2));
```

```
insert into Services(ServiceID, ServiceName, Price)
values(1, 'Room Service', 20),
(2, 'Spa', 100),
(3, 'Airport Pickup', 50),
(4, 'Laundry', 30),
(5, 'Breakfast', 15),
(6, 'Dinner', 40),
(7, 'Guided Tour', 75),
(8, 'Gym Access', 25),
(9, 'Massage', 60),
(10, 'Mini-Bar', 35);
```

Services Table:

```
Select * from Services;
```

Result Grid | Filter Rows: | Edit: | Export/Import: | Wrap Cell Content:

ServiceID	ServiceName	Price
1	Room Service	20.00
2	Spa	100.00
3	Airport Pickup	50.00
4	Laundry	30.00
5	Breakfast	15.00
6	Dinner	40.00
7	Guided Tour	75.00
8	Gym Access	25.00
9	Massage	60.00
10	Mini-Bar	35.00
NULL	NULL	NULL

- **Booking Services Table:**

➢ Purpose:Tracks additional services used by guests during their stay.

❖ Columns:

1. BookingServiceID(Primary Key)
2. BookingID(Foreign Key)
3. ServiceID(Foreign Key)
4. Quantity

```
Create Table BookingServices(
    BookingServiceID int primary key,
    BookingID int,
    ServiceID int,
    Quantity int,
    Foreign Key (BookingID) References Bookings(BookingID),
    Foreign Key (ServiceID) References Services(ServiceID));
```

```
insert into BookingServices(BookingServiceID, BookingID, ServiceID, Quantity)
values(1, 1, 1, 2),
(2, 1, 5, 1),
(3, 2, 3, 1),
(4, 3, 2, 1),
(5, 4, 6, 1),
(6, 5, 4, 2),
(7, 6, 8, 1),
(8, 7, 7, 1),
(9, 8, 9, 1),
(10, 9, 10, 2);
```

Booking Services Table:

```
select * from BookingServices;
```

Result Grid | Filter Rows: | Edit: | Export/Import: | Wrap Cell Content:

BookingServiceID	BookingID	ServiceID	Quantity
1	1	1	2
2	1	5	1
3	2	3	1
4	3	2	1
5	4	6	1
6	5	4	2
7	6	8	1
8	7	7	1
9	8	9	1
10	9	10	2
NULL	NULL	NULL	NULL

- **Payments Table:**

➢ Purpose:Records payment details for bookings.

❖ Columns:

1. PaymentID(Primary Key)
2. BookingID(Foreign Key)
3. PaymentDate
4. Amount

```
Create Table Payments(
    PaymentID int primary Key,
    BookingID int,
    PaymentDate date,
    Amount decimal(10, 2),
    Foreign Key(BookingID) References Bookings(BookingID));
```

```
insert into Payments(PaymentID, BookingID, PaymentDate, Amount)
values(1, 1, '2024-01-10', 600),
(2, 2, '2024-02-05', 650),
(3, 3, '2024-03-15', 550),
(4, 4, '2024-04-10', 900),
(5, 5, '2024-05-25', 500),
(6, 6, '2024-06-20', 600),
(7, 7, '2024-07-15', 750),
(8, 8, '2024-08-07', 690),
(9, 9, '2024-09-10', 600),
(10, 10, '2024-10-15', 800);
```

Payments Table:

```
select * from Payments;
```

Result Grid | Filter Rows: | Edit: | Export/Import: | Wrap Cell Content:

PaymentID	BookingID	PaymentDate	Amount
1	1	2024-01-10	600.00
2	2	2024-02-05	650.00
3	3	2024-03-15	550.00
4	4	2024-04-10	900.00
5	5	2024-05-25	500.00
6	6	2024-06-20	600.00
7	7	2024-07-15	750.00
8	8	2024-08-07	690.00
9	9	2024-09-10	600.00
10	10	2024-10-15	800.00
NULL	NULL	NULL	NULL

Questions:

1) Select Records:

- Write a query to select all bookings from the Bookings table where the CheckOutDate is after January 1, 2024.

```
select * from Bookings
where CheckOutDate > '2024-01-01';
```

Result Grid | Filter Rows: | Edit: | Export/Import: | Wrap Cell Content:

BookingID	GuestID	RoomID	CheckInDate	CheckOutDate
1	1	1	2024-01-05	2024-01-10
2	2	2	2024-02-01	2024-02-05
3	3	3	2024-03-10	2024-03-15
4	4	4	2024-04-01	2024-04-10
5	5	5	2024-05-20	2024-05-25
6	6	6	2024-06-15	2024-06-20
7	7	7	2024-07-10	2024-07-15
8	8	8	2024-08-01	2024-08-07
9	9	9	2024-09-05	2024-09-10
10	10	10	2024-10-10	2024-10-15
NULL	NULL	NULL	NULL	NULL

2.Where Clause (AND/OR):

- Write a query to select all guests who have a phone number starting with '555' and an email containing 'example.com'.

```
select * from Guests
where phone like '555%' and email like '%example.com%';
```

Result Grid | Filter Rows: | Edit: | Export/Import: | Wrap Cell Content:

GuestID	FirstName	LastName	Email	Phone
1	Daniel	Radcliffe	daniel.r@example.com	555-1234
3	Bonnie	Wright	bonnie.w@example.com	555-8765
4	Rupert	Grint	rupert.g@example.com	555-4321
5	Evanna	Lynch	evanna.l@example.com	555-3456
7	Maggie	Smith	maggie.s@example.com	555-9876
8	Matthew	Lewis	matthew.l@example.com	555-5432
9	Katie	Leung	katie.l@example.com	555-2345
10	Robert	Pattinson	robert.p@example.com	555-6789
NULL	NULL	NULL	NULL	NULL

3.LIKE Operator:

- Write a query to select all rooms where the RoomType contains 'Suite'.

```
select * from Rooms
where RoomType like '%Suite%';
```

Result Grid | Filter Rows: | Edit: | Export/Import: | Wrap Cell Content:

RoomID	RoomNumber	RoomType	PricePerNight
3	201	Suite	250.00
4	202	Suite	275.00
7	401	Suite	300.00
10	502	Suite	320.00
NULL	NULL	NULL	NULL

4.CASE Statement:

- Write a query to select RoomNumber and a new column DiscountedPrice from the Rooms table. If PricePerNight is greater than $200, set DiscountedPrice to PricePerNight * 0.8, otherwise PricePerNight.

```
select RoomNumber,
case
when PricePerNight > 200 then PricePerNight*0.8
else PricePerNight
end as DiscountedPrice
from Rooms;
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

RoomNumber	DiscountedPrice
101	100.00
102	150.00
201	200.000
202	220.000
301	100.00
302	160.00
401	240.000
402	110.00
501	140.00
502	256.000

5.Subquery:

- Write a query to find all guests who have made bookings with a total amount exceeding $500. Use a subquery to find these GuestIDs.

```
select * from Guests
where GuestID in (Select GuestID from bookings
                  join Payments on Bookings.BookingID=Payments.BookingID
                  group by GuestID
                  Having sum(Amount)>500);
```

Result Grid | Filter Rows: | Edit: | Export/Import: | Wrap Cell Content:

GuestID	FirstName	LastName	Email	Phone
1	Daniel	Radcliffe	daniel.r@example.com	555-1234
2	Emma	Watson	emma.w@example.com	222-5678
3	Bonnie	Wright	bonnie.w@example.com	555-8765
4	Rupert	Grint	rupert.g@example.com	555-4321
6	Tom	Felton	tom.f@example.com	444-6543
7	Maggie	Smith	maggie.s@example.com	555-9876
8	Matthew	Lewis	matthew.l@example.com	555-5432
9	Katie	Leung	katie.l@example.com	555-2345
10	Robert	Pattinson	robert.p@example.com	555-6789
NULL	NULL	NULL	NULL	NULL

6.Group By:

- Write a query to get the total number of rooms booked by each RoomType. Group the results by RoomType.

```
select RoomType, count(*) as TotalRoomsBooked
from Rooms
join Bookings on Rooms.RoomID = Bookings.RoomId
Group by RoomType;
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

RoomType	TotalRoomsBooked
Single	3
Double	3
Suite	4

7.Having Clause:

- Write a query to get the total amount paid for each booking, but only include bookings where the total amount is greater than $100. Use the HAVING clause.

```
select BookingID,sum(Amount) TotalAmount
from payments
group by BookingID
having sum(Amount) > 100;
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

BookingID	TotalAmount
1	600.00
2	650.00
3	550.00
4	900.00
5	500.00
6	600.00
7	750.00
8	690.00
9	600.00
10	800.00

8.Limit:

- Write a query to select the top 5 guests who have stayed the most number of nights.

```
select g.GuestID,sum(datediff(b.CheckOutDate,b.CheckInDate)) NumberOfNights
from Guests g
join Bookings b on g.GuestID = b.GuestID
group by g.GuestID
order by NumberOfNights desc
limit 5;
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content: | Fetch rows:

GuestID	NumberOfNights
4	9
8	6
1	5
3	5
5	5

9.Inner Join:

- Write a query to join Bookings with Guests to get a list of all bookings with FirstName, LastName, CheckInDate, and CheckOutDate.

```
select Firstname,Lastname,CheckInDate,CheckOutDate
from Guests g join Bookings b
on (g.GuestID = b.GuestID);
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

Firstname	Lastname	CheckInDate	CheckOutDate
Daniel	Radcliffe	2024-01-05	2024-01-10
Emma	Watson	2024-02-01	2024-02-05
Bonnie	Wright	2024-03-10	2024-03-15
Rupert	Grint	2024-04-01	2024-04-10
Evanna	Lynch	2024-05-20	2024-05-25
Tom	Felton	2024-06-15	2024-06-20
Maggie	Smith	2024-07-10	2024-07-15
Matthew	Lewis	2024-08-01	2024-08-07
Katie	Leung	2024-09-05	2024-09-10
Robert	Pattinson	2024-10-10	2024-10-15

10.Outer Join:

- Write a query to get a list of all rooms and any associated bookings. Include rooms that might not be booked.

```
select r.RoomID,r.RoomNumber,r.RoomType,r.PricePerNight,b.BookingID,b.GuestID,b.CheckInDate,b.CheckOutDate
from Rooms r left join Bookings b
on r.RoomID=b.RoomID
order by r.RoomID,b.BookingID;
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

RoomID	RoomNumber	RoomType	PricePerNight	BookingID	GuestID	CheckInDate	CheckOutDate
1	101	Single	100.00	1	1	2024-01-05	2024-01-10
2	102	Double	150.00	2	2	2024-02-01	2024-02-05
3	201	Suite	250.00	3	3	2024-03-10	2024-03-15
4	202	Suite	275.00	4	4	2024-04-01	2024-04-10
5	301	Single	100.00	5	5	2024-05-20	2024-05-25
6	302	Double	160.00	6	6	2024-06-15	2024-06-20
7	401	Suite	300.00	7	7	2024-07-10	2024-07-15
8	402	Single	110.00	8	8	2024-08-01	2024-08-07
9	501	Double	140.00	9	9	2024-09-05	2024-09-10
10	502	Suite	320.00	10	10	2024-10-10	2024-10-15

11.Join with Aggregation:

- Write a query to get the total revenue generated by each RoomType. Use an INNER JOIN between Rooms and Bookings, and group by RoomType.

```
select r.RoomType, SUM(r.PricePerNight * DATEDIFF(b.CheckOutDate, b.CheckInDate)) AS TotalRevenue
from Rooms r
inner join Bookings b on r.RoomID = b.RoomID
group by r.RoomType;
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

RoomType	TotalRevenue
Single	1660.00
Double	2100.00
Suite	6825.00

12.Subquery with Join:

- Write a query to find all bookings where the total services cost is higher than the average service cost. Use a subquery in the WHERE clause.

```
select b.BookingID,b.GuestID
from Bookings b
join BookingServices bs on b.BookingID=bs.BookingID
join Services s on bs.ServiceID=s.ServiceID
where (select sum(bs.Quantity*s.Price))>(select avg(price) from services);
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

BookingID	GuestID
2	2
3	3
5	5
7	7
8	8
9	9

13.Advanced Join:

- Write a query to list RoomNumber, GuestName, and ServiceName for all rooms that have been booked and where additional services were provided. Use INNER JOIN and LEFT JOIN as necessary to get all required details.

```
select r.RoomNumber,CONCAT(g.Firstname,' ',g.Lastname) GuestName,s.ServiceName
from Rooms r
join Bookings b on r.RoomID = b.RoomID
join Guests g on b.GuestID = g.GuestID
join BookingServices bs on b.BookingID = bs.BookingID
left join services s on bs.ServiceID = s.ServiceID
where bs.Quantity>0;
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

RoomNumber	GuestName	ServiceName
101	Daniel Radcliffe	Room Service
101	Daniel Radcliffe	Breakfast
102	Emma Watson	Airport Pickup
201	Bonnie Wright	Spa
202	Rupert Grint	Dinner
301	Evanna Lynch	Laundry
302	Tom Felton	Gym Access
401	Maggie Smith	Guided Tour
402	Matthew Lewis	Massage
501	Katie Leung	Mini-Bar

Conclusion

- **Summary:**Effective use of SQL for managing hotel data.
- **Key Insights:**How the queries help in understand bookings,revenue,guest behavior and service usage.

Thank You