SQL CASE STUDY

HOTEL MANAGEMENT

DONE BY

ARBAZ SHARIFF S

SQL CASE STUDY

HOTEL MANAGEMENT

INTRODUCTION:

In this case study we have been made to create a database for a hotel to efficiently manage its day-to-day operations. This database will store information about rooms, guests, reservations and payments.

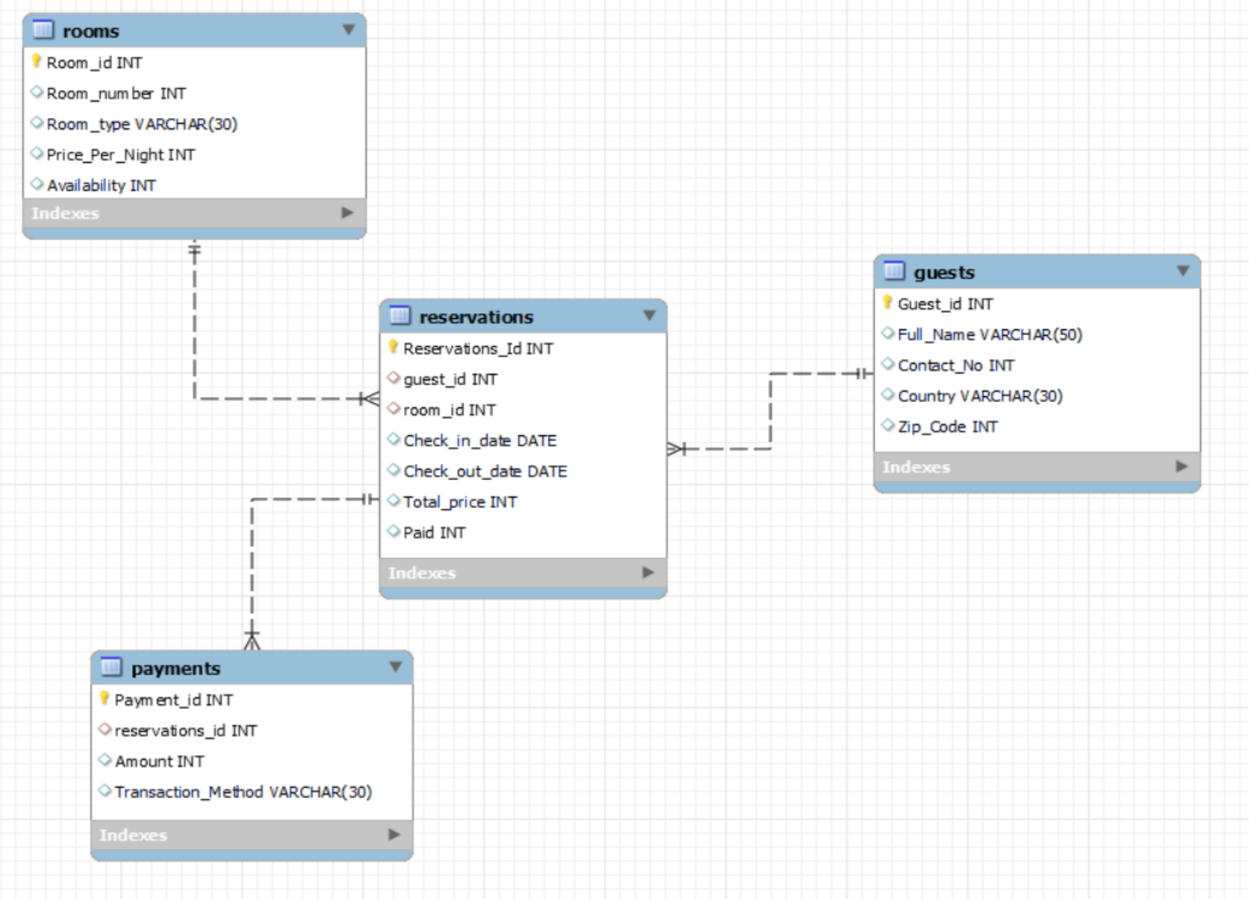
Key statements:

Hotel Management wants to use the data to answer a few questions about their guests, especially about their visiting patterns, how much money they've spent… Having this deeper connection with his guests will help them deliver a better and more personalized experience for his loyal customers.

Key Tables provided:

1. Rooms
2. Guests
3. Reservations
4. Payments

Entity_Relationship_Diagram:



Dataset:

create database Hotel_Management;

use Hotel_Management;

select * from Rooms;

select * from Guests;

select * from Reservations;

Select * from Payments;

---ROOM TABLE---

Create & Insert data:

```
create table Rooms(
Room_id int primary key,
Room_number int,
Room_type varchar(30),
Price_Per_Night int,
Availability int
);
insert into Rooms
values
(1,101,'Single',1500,1),
(2,201,'Double',3000,1),
(3,301,'Suite',5000,1);
```

---GUESTS TABLE---

Create & Insert data:

```
create table Guests(
Guest_id int primary key,
Full_Name varchar(50),
Contact_No int ,
Country varchar(30),
Zip_Code int
);
insert into guests
```

```
values
(1,'Levi Ackermann',4516258,'JPN',12345),
(2,'William Butcher',5452154,'USA',86164),
(3,'MuthuRaj',3665141,'IND',94514),
(4,'Prathana',5221472,'ENG',16543),
(5,'Abdul Hameed',7412591,'UAE',45842),
(6,'John Smith',1244141,'USA',86164),
(7,'Michael Bay',6441411,'FRA',26434),
(8,'Arjun Kumar',3614525,'IND',94514),
(9,'Samuel Jackson',5788515,'ENG',16543),
(10,'Saravanavel',3661578,'IND',94514);
```

--RESERVATIONS TABLE--------------------------------

Create & Insert data:

```
create table Reservations (
Reservations_Id int primary key,
guest_id int,
room_id int,
Check_in_date date,
Check_out_date date,
Total_price int,
Paid int,
foreign key (guest_id) references Guests (Guest_id),
foreign key (room_id) references Rooms (Room_id)
);
```

```
insert into Reservations (Reservations_Id, guest_id, room_id, Check_in_date,
Check_out_date, Total_price, Paid)

values

(1, 1, 1, '2023-07-20', '2023-07-25',7500 , 1),

(2, 3, 2, '2023-08-05', '2023-08-08', 9000, 1),

(3, 5, 3, '2023-08-15', '2023-08-16', 5000, 0),

(4, 2, 1, '2023-09-01', '2023-09-03', 3000, 1),

(5, 6, 2, '2023-07-25', '2023-08-01', 15000, 1),

(6, 8, 3, '2023-08-10', '2023-08-12', 10000, 1),

(7, 10, 1, '2023-08-20', '2023-08-21', 1500, 0),

(8, 1, 2, '2023-09-05', '2023-09-09', 12000, 1),

(9, 7, 3, '2023-09-15', '2023-09-20', 25000, 1),

(10, 3, 1, '2023-10-01', '2023-10-02', 1500, 0),

(11, 9, 2, '2023-10-01', '2023-10-04', 12000, 1),

(12, 4, 3, '2023-10-01', '2023-10-03', 10000, 1),

(13, 10, 1, '2023-11-01', '2023-11-03', 3000, 1),

(14, 1, 2, '2023-11-11', '2023-11-13', 6000, 0),

(15, 2, 3, '2023-11-13', '2023-11-16', 15000, 1),

(16, 6, 1, '2023-11-17', '2023-11-18', 1500, 1),

(17, 10, 2, '2023-11-10', '2023-11-15', 15000, 0),

(18, 8, 3, '2023-12-20', '2023-12-23', 15000, 1),

(19, 4, 1, '2023-12-23', '2023-12-27', 6000, 1),

(20, 10, 2, '2024-01-09', '2024-01-13', 12000, 0),

(21, 3, 3, '2024-01-12', '2024-01-16', 20000, 1),

(22, 7, 1, '2024-01-12', '2024-01-15', 4500, 1),

(23, 8, 2, '2024-02-08', '2024-02-11', 9000, 0),
(24, 4, 3, '2024-02-09', '2024-02-11', 10000, 1),

(25, 6, 1, '2024-03-02', '2024-03-05', 4500, 1),

(26, 1, 2, '2024-03-10', '2024-03-13', 9000, 0),

(27, 7, 3, '2024-03-17', '2024-03-21', 20000, 1),

(28, 6, 1, '2024-04-03', '2024-04-05', 3000, 1),

(29, 9, 2, '2024-04-10', '2024-04-13', 9000, 0),

(30, 3, 3, '2024-04-15', '2024-04-18', 15000, 1),

(31, 5, 1, '2024-05-01', '2024-05-10', 15000, 1),

(32, 8, 2, '2024-05-03', '2025-05-06', 9000, 0),

(33, 7, 3, '2024-05-21', '2024-05-25', 20000, 1),

(34, 4, 1, '2024-06-02', '2024-06-05', 4500, 1),

(35, 8, 2, '2024-06-10', '2024-06-15', 15000, 1),

(36, 6, 3, '2024-06-23', '2024-06-25', 10000, 0),

(37, 10, 1, '2024-07-03', '2024-07-05', 3000, 1),

(38, 1, 2, '2024-07-11', '2024-07-15', 12000, 1),

(39, 9, 3, '2024-07-22', '2024-07-25', 15000, 0),

(40, 3, 1, '2024-08-02', '2024-08-05', 4500, 1),

(41, 4, 2, '2024-08-13', '2024-08-15', 9000, 1),

(42, 6, 3, '2024-08-20', '2024-08-22', 10000, 0),

(43, 5, 1, '2024-09-01', '2024-09-04', 6000, 1),

(44, 3, 2, '2024-09-13', '2024-09-15', 6000, 1),

(45, 1, 3, '2024-09-17', '2024-09-20', 15000, 0),

(46, 2, 1, '2024-10-02', '2024-10-05', 4500, 1),

(47, 7, 2, '2024-10-12', '2024-10-15', 9000, 0),

(48, 10, 3, '2024-10-21', '2024-10-25', 20000, 1),

(49, 1, 1, '2024-10-29', '2024-11-02', 4500, 1),
(50, 3, 2, '2024-11-08', '2024-11-11', 9000, 0);
```

--PAYMENTS TABLE--

Create & Insert data:

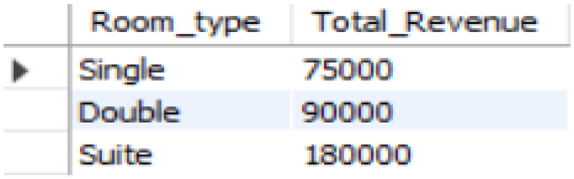
```
create table Payments(
Payment_id int primary key,
reservations_id int,
Amount int ,
Transaction_Method varchar(30),
foreign key (reservations_id) references Reservations (Reservations_Id)
);

INSERT INTO Payments (Payment_id, reservations_id, Amount, Transaction_Method)
VALUES
(101, 1, 7500, 'Credit Card'),
(102, 2, 9000, 'Debit Card'),
(103, 3, NULL, NULL),
(104, 4, 3000, 'Cash'),
(105, 5, 15000, 'Credit Card'),
(106, 6, 10000, 'Debit Card'),
(107, 7, NULL, NULL),
(108, 8, 12000, 'Credit Card'),
(109, 9, 25000, 'Debit Card'),
(110, 10, NULL, NULL),
(111, 11, 12000, 'Credit Card'),
(112, 12, 10000, 'Debit Card'),
(113, 13, 3000, 'Cash'),
(114, 14, NULL, NULL),
(115, 15, 15000, 'Credit Card'),
(116, 16, 1500, 'Debit Card'),
(117, 17, NULL, NULL),
(118, 18, 15000, 'Credit Card'),
(119, 19, 6000, 'Debit Card'),
(120, 20, NULL, NULL),
(121, 21, 20000, 'Credit Card'),
(122, 22, 4500, 'Debit Card'),
(123, 23, NULL, NULL),
(124, 24, 10000, 'Credit Card'),
(125, 25, 4500, 'Debit Card'),
(126, 26, NULL, NULL),
(127, 27, 20000, 'Credit Card'),
(128, 28, 3000, 'Debit Card'),
(129, 29, NULL, NULL),
(130, 30, 15000, 'Credit Card'),
(131, 31, 9000, 'Debit Card'),
(132, 32, NULL, NULL),
(133, 33, 20000, 'Credit Card'),
(134, 34, 4500, 'Debit Card'),
(135, 35, 15000, 'Credit Card'),
(136, 36, NULL, NULL),
(137, 37, 3000, 'Debit Card'),
(138, 38, 12000, 'Credit Card'),
(139, 39, NULL, NULL),
(140, 40, 4500, 'Debit Card'),
(141, 41, 9000, 'Credit Card'),
(142, 42, NULL, NULL),
(143, 43, 6000, 'Debit Card'),
(144, 44, 6000, 'Credit Card'),
(145, 45, NULL, NULL),
(146, 46, 4500, 'Debit Card'),
(147, 47, 9000, 'Credit Card'),
(148, 48, NULL, NULL),
(149, 49, 4500, 'Debit Card'),
(150, 50, 9000, 'Credit Card');
```

Case Study Questions & Answers:

1. RETRIEVE THE TOTAL REVENUE ANALYSIS GENERATED FROM RESERVATION WHERE PAYMENT HAS RECEIVED (Paid = 1) GROUPED BY ROOM TYPE

```
select
r.Room_type,
sum(re.Total_price) as Total_Revenue
from Rooms r
join Reservations re
on r.Room_id = re.room_id
where Paid = 1
```

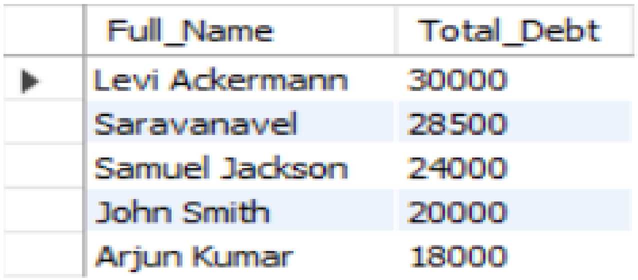
```
group by r.Room_type;
```



Room_type	Total_Revenue
Single	75000
Double	90000
Suite	180000

2. FIND OUT TOP 5 UNPAID CUSTOMERS WITH TOTAL DEBT

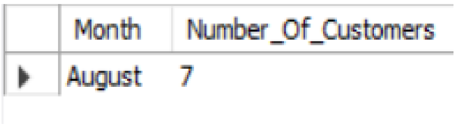
```
select g.Full_Name,
sum(r.Total_price) as Total_Debt
from Guests g
join Reservations r
on g.Guest_id = r.guest_id
where r.paid = 0
group by g.Full_Name
order by Total_Debt desc
limit 5 ;
```



Full_Name	Total_Debt
Levi Ackermann	30000
Saravanavel	28500
Samuel Jackson	24000
John Smith	20000
Arjun Kumar	18000

3. FIND OUT WHICH MONTH RECEIVED THE MOST CUSTOMERS BASED ON RESERVATIONS.

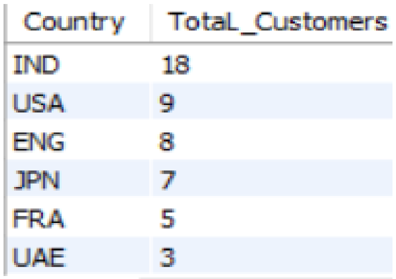
```
select monthname(Check_in_date) as Month,
count(*) as Number_Of_Customers
from Reservations
group by monthname(Check_in_date)
order by Number_Of_Customers desc
limit 1;
```



Month	Number_Of_Customers
August	7

4. RETRIEVE WHICH ZONE HAS MOST REPETETIVE CUSTOMER USING RESERVATIONS.

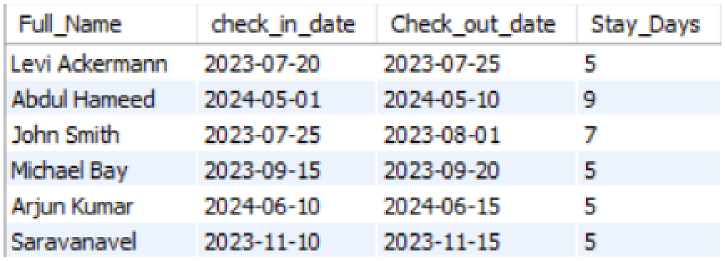
```
select Country,

count(*) as TotaL_Customers

from Reservations r

join Guests g

where r.guest_id = g.Guest_id

group by g.Country

order by Total_Customers desc;
```



Country	TotaL_Customers
IND	18
USA	9
ENG	8
JPN	7
FRA	5
UAE	3

5. FIND OUT CUSTOMERS WHO STAYED MORE THAN FOUR DAYS.

```
select g.Full_Name,
r.check_in_date,
r.Check_out_date,
datediff(r.Check_out_date,r.Check_in_date) as Stay_Days
from Guests g
join Reservations r
on g.Guest_id  = r.guest_id
where datediff(r.Check_out_date,r.Check_in_date) > 4;
```

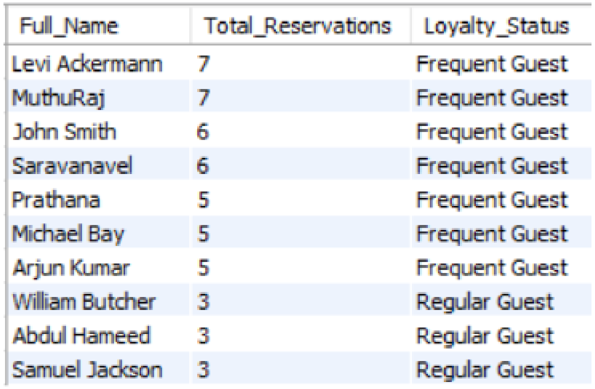


Full_Name	check_in_date	Check_out_date	Stay_Days
Levi Ackermann	2023-07-20	2023-07-25	5
Abdul Hameed	2024-05-01	2024-05-10	9
John Smith	2023-07-25	2023-08-01	7
Michael Bay	2023-09-15	2023-09-20	5
Arjun Kumar	2024-06-10	2024-06-15	5
Saravanavel	2023-11-10	2023-11-15	5

6. ASSIGN LOYALTY STATUS TO GUESTS BY CATEGORIZING GUESTS BY RESERVATIONS.

```
select g.Full_Name,
count(r.Reservations_id) as Total_Reservations,
case
when count(r.Reservations_id) = 1 then 'New Guest'
when count(r.Reservations_id) >=2 AND count(r.Reservations_id) < 5  then 'Regular Guest'
else 'Frequent Guest'
```

```
end as 'Loyalty_Status'

from Guests g

join Reservations r

on g.Guest_id  = r.guest_id

group by g.Full_Name

order by Total_Reservations desc;
```

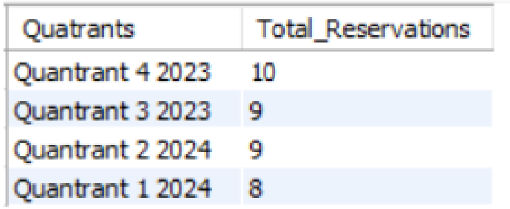


Full_Name	Total_Reservations	Loyalty_Status
Levi Ackermann	7	Frequent Guest
MuthuRaj	7	Frequent Guest
John Smith	6	Frequent Guest
Saravanavel	6	Frequent Guest
Prathana	5	Frequent Guest
Michael Bay	5	Frequent Guest
Arjun Kumar	5	Frequent Guest
William Butcher	3	Regular Guest
Abdul Hameed	3	Regular Guest
Samuel Jackson	3	Regular Guest

7. DIVIDE THE YEAR OF FROM JULY 2023 TO JUNE 2024 BY FOUR QUARANTS AND FIND OUT WHICH QUATRANT HAVE MOST TO LEAST RESERVATIONS.

With Built-in Function :

```
select

case

when date_format(Check_in_date,'%Y-%m') between '2023-07' and '2023-09' then 'Quantrant 3 2023'

when date_format(Check_in_date,'%Y-%m') between '2023-10' and '2023-12' then 'Quantrant 4 2023'
when date_format(Check_in_date,'%Y-%m') between '2024-01' and '2024-03' then 'Quantrant 1 2024'

when date_format(Check_in_date,'%Y-%m') between '2024-04' and '2024-06' then 'Quantrant 2 2024'

end as Quatrants,

count(*) as Total_Reservations

from Reservations

where Check_in_date between '2023-07-01' and '2024-06-30'

group by Quatrants

order by Total_Reservations desc;
```

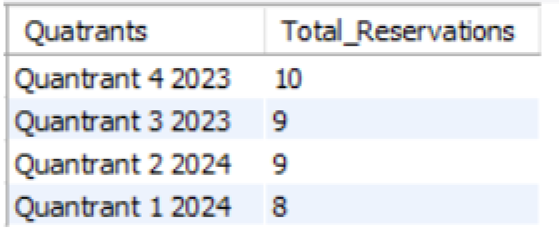


Quatrants	Total_Reservations
Quantrant 4 2023	10
Quantrant 3 2023	9
Quantrant 2 2024	9
Quantrant 1 2024	8

Without Built-in Functions:

```
select

case

when Check_in_date between '2023-07-01' and '2023-09-30' then 'Quantrant 3 2023'

when Check_in_date between '2023-10-01' and '2023-12-31' then 'Quantrant 4 2023'

when Check_in_date between '2024-01-01' and '2024-03-31' then 'Quantrant 1 2024'

when Check_in_date between '2024-04-01' and '2024-06-30' then 'Quantrant 4 2024'

end as Quatrants,

count(*) as Total_Reservations

from Reservations

where Check_in_date between '2023-07-01' and '2024-06-30'

group by Quatrants
```

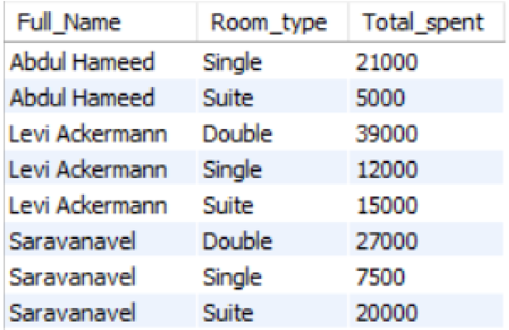
```
order by Total_Reservations desc;
```



Quatrants	Total_Reservations
Quantrant 4 2023	10
Quantrant 3 2023	9
Quantrant 2 2024	9
Quantrant 1 2024	8

8. FIND THE TOTAL AMOUNT SPENT IN EACH ROOM TYPE OF GUEST NAME LEVI ACKERMANN, ABDUL HAMEED, SARAVANAVEL

```
select g.Full_Name,

r.Room_type, sum(re.Total_price) as Total spent

from Guests g

join  Reservations re

on g.Guest_id = re.guest_id

join Rooms r

on r.Room_id = re.room_id

where g.Full_Name in ('Levi Ackermann','Abdul Hameed','Saravanavel')

group by g.Full_name,r.Room_type

order by g.Full_Name;
```



Full_Name	Room_type	Total_spent
Abdul Hameed	Single	21000
Abdul Hameed	Suite	5000
Levi Ackermann	Double	39000
Levi Ackermann	Single	12000
Levi Ackermann	Suite	15000
Saravanavel	Double	27000
Saravanavel	Single	7500
Saravanavel	Suite	20000

9. RANK GUESTS BASED ON THEIR TOTAL SPENDING ON RESERVATIONS.

Select g.Full_Name, sum(r.Total_price) as Total spent,

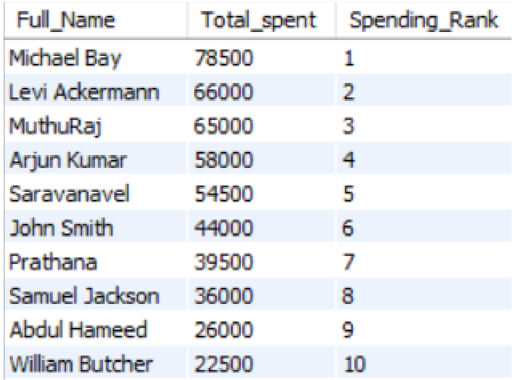
rank()over(order by sum(r.Total_price)desc) as Spending_Rank

from Guests g

join Reservations r

on g.Guest_id = r.guest_id

group by g.Full_Name;



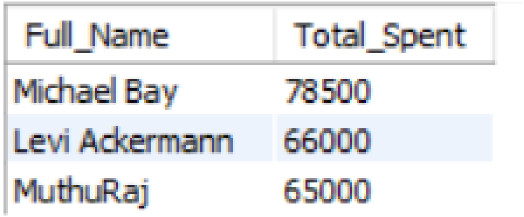
Full_Name	Total_spent	Spending_Rank
Michael Bay	78500	1
Levi Ackermann	66000	2
MuthuRaj	65000	3
Arjun Kumar	58000	4
Saravanavel	54500	5
John Smith	44000	6
Prathana	39500	7
Samuel Jackson	36000	8
Abdul Hameed	26000	9
William Butcher	22500	10

10. FIND THE TOP 3 GUESTS WHO HAVE SPENT THE MOST IN TOTAL ACROSS ALL THEIR RESERVATIONS. DISPLAY THEIR FULL NAMES AND TOTAL AMOUNT SPENT

select g.Full_Name,

sum(Total_price) as Total_Spent

```
from Guests g

join Reservations r

on g.Guest_id = r.guest_id

group by g.Full_Name

order by Total_spent desc

limit 3;
```



Full_Name	Total_Spent
Michael Bay	78500
Levi Ackermann	66000
MuthuRaj	65000

Conclusion :

The hotel management case study provides valuable insights into various aspects of operations, guests behavior, and financial performance through comprehensive SQL queries and analyses.

These queries provide valuable insight into various aspects of hotel management, including revenue analysis, customer behavior, loyalty categorization for making informed operational and strategic decisions in the hospitality industry.