ASSIGNMENT 6

1. **Write queries for the following:**

   a) Show all the customers who have no invoices.

   Ans:

```
select c.customerId, c.Name from customer c
left join invoice i
on c.customerId = i.customerId
where i.invoiceId is null;
```

   b) Show all the transactions which have no invoices generated.

   Ans:

```
select t.transactionId, transactionDate from dbo.[transaction] as t
left join invoice i
on t.invoiceId = i.invoiceId
where i.invoiceId is null;
```

   c) Show all the invoices with customer detail which has more than 5 quantity bought.

   Ans:

```
select i.invoiceId, c.customerId, c.Name from invoice as i
join [transaction] as t
on i.invoiceId = t.invoiceId
join customer as c
on t.customerId = c.customerId
where t.Quantity<5;
```

d) Show all the customers who only booked room but didn't take any other services.

Ans:

```
select c.customerId, c.Name from customer c
join booking b
on c.customerId = b.customerId
left join [transaction] t
on b.bookingId = t.bookingId
where t.serviceId is null;
```

2. Create the following functions:

a) Create a function which provides customer name (fullname in same column) based on customerId passed

Ans:

```
CREATE FUNCTION getCustomerFullName(@customerId int)

RETURNS VARCHAR(100)

AS
BEGIN
    DECLARE @fullName VARCHAR(100)

    SELECT @fullName = c.FirstName + ' ' +c.LastName
    FROM dbo.[Customer] as c
    WHERE CustomerId = @customerId
```

```
    RETURN @fullName

END;

SELECT Dbo.getCustomerFullName(1) as FullName;
```

b) Create a function which returns all the transaction details based on the Invoice Number passed.

Ans:

```
CREATE OR ALTER FUNCTION getTransactionDetails(@invoiceNumber INT)
RETURNS TABLE
AS
RETURN
(
   SELECT t.TransactionId, t.CustomerId, t.BookingId, t.ServiceId, t.TransactionDate,
t.UserId, t.Total, t.Quantity
   FROM dbo.[Transaction] t
   JOIN dbo.Invoice i ON t.InvoiceId = i.InvoiceId
   WHERE i.Invoiceid = @invoiceNumber
);

SELECT * FROM dbo.getTransactionDetails(1);
```

NOTE: In the context of a table-valued function in SQL Server, the declaration of the table variable is not required. The table variable is implicitly declared within the RETURNS TABLE statement.

c) Create a function to return all the branches based on the hotelId passed.

Ans:

```
CREATE OR ALTER FUNCTION getAllBranches(@hotelId INT)
RETURNS VARCHAR(200)
AS
BEGIN

    DECLARE @branchName VARCHAR(200)
   SELECT @branchName = b.BranchName
   FROM dbo.[Branch] as b
   JOIN dbo.Hotel h ON h.HotelId = b.HotelId
   WHERE h.HotelId = @hotelId

    RETURN @branchName

END;

SELECT dbo.[getAllBranches](5) as BranchName;
```

d) Create a function to return CustomerName, Total number of Transaction made, Total services taken, total number of invoices and total invoice amount (only 1 row) based on the customerId passed.

Ans:

```
CREATE OR ALTER FUNCTION getCustomerSummary(@customerId INT)
RETURNS TABLE
```

```
AS
RETURN
(
    SELECT c.Name AS CustomerName,
           COUNT(t.TransactionId) AS TotalTransactions,
           COUNT(t.ServiceId) AS TotalServices,
           COUNT(i.InvoiceId) AS TotalInvoices,
           SUM(i.TotalInvoice) AS TotalInvoiceAmount
    FROM dbo.[Transaction] t
    JOIN dbo.Customer c ON t.TransactionId = c.CustomerId
    JOIN dbo.Invoice i ON t.InvoiceId = i.InvoiceId
    WHERE c.CustomerId = @customerId
    GROUP BY c.Name
);


SELECT * FROM dbo.getCustomerSummary(5);
```