**CREATE**

CREATE TABLE HotelLocation(

ZipCode INT NOT NULL PRIMARY KEY,

City VARCHAR(25),

Area VARCHAR(25),

);

CREATE TABLE HotelType(

HotelTypeId VARCHAR(5) NOT NULL PRIMARY KEY,

HotelTypeDescriptions VARCHAR(100)

);

CREATE TABLE RoomType(

RoomTypeId VARCHAR(5) NOT NULL PRIMARY KEY,

RoomTypeDescriptions VARCHAR(100),

PriceRange INT

);

CREATE TABLE Hotel(

HotelId VARCHAR(5) NOT NULL PRIMARY KEY,

HotelName VARCHAR(100),

AboutInfo VARCHAR(100),

HotelTypeId VARCHAR(5) NOT NULL,

ZipCode INT NOT NULL,

CONSTRAINT HotelTypeId\_Fk FOREIGN KEY (HotelTypeId) REFERENCES HotelType(HotelTypeId),

CONSTRAINT ZipCode\_Fk FOREIGN KEY (ZipCode) REFERENCES HotelLocation(ZipCode)

);

CREATE TABLE Room(

RoomId VARCHAR(5) NOT NULL PRIMARY KEY,

NumberOfPersons INT,

Availablity VARCHAR(10)

CONSTRAINT Availibity\_Chk CHECK (Availablity IN ('Available', 'Unavilable')), --CHECK

Beds INT,

Tariff INT,

RoomTypeId VARCHAR(5) NOT NULL,

HotelId VARCHAR(5) NOT NULL,

CONSTRAINT HotelId\_Fk FOREIGN KEY (HotelId) REFERENCES Hotel(HotelId),

CONSTRAINT RoomTypeId\_Fk FOREIGN KEY (RoomTypeId) REFERENCES RoomType(RoomTypeId)

);

--ALTER TABLE Room ALTER COLUMN Availablity VARCHAR(10)

CREATE TABLE Customer(

CustomerId VARCHAR(5) NOT NULL PRIMARY KEY,

CustomerName VARCHAR(100),

Gender VARCHAR(10) CONSTRAINT Gender\_Chk CHECK (Gender IN ('Male', 'Female', 'Other')),

EmailId VARCHAR(25) CONSTRAINT email\_ck CHECK (Emailid LIKE '[a-z,0-9,\_,-]%@[a-z]%.[a-z][a-z]%'),

Phone BIGINT NOT NULL CONSTRAINT Phone\_ck CHECK (len([Phone])=10),

DOB DATE

);

CREATE TABLE Subscription(

SubscriptionId VARCHAR(5) NOT NULL PRIMARY KEY,

SubscriptionStartDate DATE,

SubscriptionEndDate DATE,

CustomerId VARCHAR(5) NOT NULL

CONSTRAINT CustomerId\_Fk FOREIGN KEY (CustomerId) REFERENCES Customer(CustomerId)

);

CREATE TABLE Offer(

OfferId VARCHAR(5) NOT NULL PRIMARY KEY,

OfferDescription VARCHAR(100),

OfferTermsandConditions VARCHAR(100),

OfferValidity DATE

);

CREATE TABLE Booking(

BookingId VARCHAR(5) NOT NULL PRIMARY KEY,

CustomerId VARCHAR(5) NOT NULL,

HotelId VARCHAR(5) NOT NULL,

RoomId VARCHAR(5) Not null,

BookingDate DATE,

BookingDetails VARCHAR(100),

CONSTRAINT BRoomId\_Fk FOREIGN KEY (RoomId) REFERENCES Room(RoomId),

CONSTRAINT BookingCustomerId\_Fk FOREIGN KEY (CustomerId) REFERENCES Customer(CustomerId),

CONSTRAINT BookingHotel\_Fk FOREIGN KEY (HotelId) REFERENCES Hotel(HotelId),

);

CREATE TABLE Payment(

PaymentId VARCHAR(5) NOT NULL,

Total BIGINT,

DateOfTransaction DATE Default GETDATE(),

TimeofTransaction TIME Default GETDATE(),

BookingId VARCHAR(5) NOT NULL,

Constraint PaymentBookingId\_pk PRIMARY KEY (PaymentId),

CONSTRAINT PaymentBookingId\_Fk FOREIGN KEY (BookingId) REFERENCES Booking(BookingId)

);

CREATE TABLE Discount(

DiscountPercentage INT,

CustomerId VARCHAR(5) NOT NULL,

OfferId VARCHAR(5) NOT NULL,

CONSTRAINT Discount\_Pk PRIMARY KEY (CustomerId, OfferId),

CONSTRAINT DiscountCustomerId\_Fk FOREIGN KEY (CustomerId) REFERENCES Customer(CustomerId),

CONSTRAINT DiscountOfferId\_Fk FOREIGN KEY (OfferId) REFERENCES Offer(OfferId)

);

CREATE TABLE BookingHistory(

BookingHistoryId VARCHAR(5) NOT NULL PRIMARY KEY,

BookingId VARCHAR(5) NOT NULL,

CONSTRAINT BHBookingId\_Fk FOREIGN KEY (BookingId) REFERENCES Booking(BookingId)

);

CREATE TABLE Review(

CustomerId VARCHAR(5) NOT NULL,

HotelId VARCHAR(5) NOT NULL,

ReviewDescription VARCHAR(100),

CONSTRAINT Review\_Pk PRIMARY KEY (CustomerId, HotelId),

CONSTRAINT ReviewCustomerId\_Fk FOREIGN KEY (CustomerId) REFERENCES Customer(CustomerId),

CONSTRAINT ReviewHotelId\_Fk FOREIGN KEY (HotelId) REFERENCES Hotel(HotelId)

);

CREATE TABLE RoomBookingConfirmation(

BookingId VARCHAR(5) NOT NULL,

RoomId VARCHAR(5) NOT NULL,

CheckInDate DATE,

CheckOutDate DATE,

CONSTRAINT Date\_check CHECK (CheckInDate<CheckOutDate),

CONSTRAINT RBCRoomBookingConfirmation\_Pk PRIMARY KEY (BookingId,RoomId),

CONSTRAINT RBCBookingId\_Fk FOREIGN KEY (BookingId) REFERENCES Booking(BookingId),

CONSTRAINT RBCRoomId\_Fk FOREIGN KEY (RoomId) REFERENCES Room(RoomId)

);

CREATE TABLE LocationType(

CustomerId VARCHAR(5) NOT NULL,

ZipCode INT NOT NULL,

LocationName VARCHAR(100),

CONSTRAINT LocationType\_Pk PRIMARY KEY (CustomerId, ZipCode),

CONSTRAINT LTCustomerId\_Fk FOREIGN KEY (CustomerId) REFERENCES Customer(CustomerId),

CONSTRAINT LTZipCode\_Fk FOREIGN KEY (ZipCode) REFERENCES HotelLocation(ZipCode)

);

**DROP**

drop table LocationType

drop table RoomBookingConfirmation

drop table Review

drop table Discount

drop table Payment

drop table BookingHistory

drop table Booking

drop table Offer

drop table Subscription

drop table Customer

drop table Room

drop table Hotel

drop table RoomType

drop table HotelType

drop table HotelLocation

**INSERT**

--inserting values into HoteLocation Table

Insert into HotelLocation(

ZipCode,

City,

Area

)

values

(02134,'Boston','Allston'),

(02215, 'Boston', 'Park Dr'),

(02119, 'Boston', 'Roxbury'),

(02135,'Boston','Brighton'),

(10001, 'New York', 'New York'),

(02188, 'Boston', 'Weymouth Town'),

(02090, 'Boston', 'Westwood'),

(02467, 'Boston', 'Brookline'),

(02474, 'Boston', 'Arlington'),

(02145, 'Boston', 'Someville');

--inserting values into HoteType Table

Insert into HotelType (

HotelTypeId,

HotelTypeDescriptions

)

values

('HT001','1 star'),

('HT002','2 star'),

('HT003','2 star'),

('HT004','3 star'),

('HT005','5 star'),

('HT006','1 star'),

('HT007','3 star'),

('HT008','5 star'),

('HT009','7 star'),

('HT010','4 star');

--inserting values into RoomType Table

Insert into RoomType(

RoomTypeId,

RoomTypeDescriptions,

PriceRange

)

values

('RT001','Single Bed Room',80),

('RT002','Double Bed Room', 100),

('RT003','Private Suite', 300),

('RT004', '3 Bed Room', 200),

('RT005','Single Bed with Balcony', 120),

('RT006','Double Bed with Balcony', 180),

('RT007','Single Bed Room', 80),

('RT008','Double Bed Room', 100),

('RT009','Private Suite', 300),

('RT010','3 Bed Room', 200);

--inserting values into Hotel Table

Insert into Hotel values

('H0001','The Bostonian Boston','Serviced Apartment','HT001',02134),

('H0002','The City Hotel','Motel','HT002',02215),

('H0003','Flamingo In','Resort','HT003',02119),

('H0004','The Alfa','All suites','HT004',02135),

('H0005','The Westin','Boutique hotel','HT005',10001),

('H0006','Hiton Hotel','Serviced Apartment','HT006',02188),

('H0007','Bosto5n View Hotel','Motel','HT007',02090),

('H0008','JW Mariott','Serviced Apartment','HT008',02467),

('H0009','Sheraton','Resort','HT009',02474),

('H0010','Boston Park Plaza','Boutique Hotel','HT010',02145);

--inserting values into Room Table

Insert into Room Values

('R0001',2,'Available',2,90,'RT001','H0001'),

('R0002',2,'Available',2,100,'RT002','H0002'),

('R0003',3,'Available',3,140,'RT003','H0003'),

('R0004',2,'Available',2,80,'RT004','H0004'),

('R0005',2,'Available',2,95,'RT005','H0005'),

('R0006',4,'Available',4,180,'RT006','H0006'),

('R0007',2,'Available',2,110,'RT007','H0007'),

('R0008',4,'Available',4,250,'RT008','H0008'),

('R0009',2,'Available',2,85,'RT009','H0009'),

('R0010',2,'Available',2,75,'RT010','H0010');

Insert into Customer(

CustomerId,

CustomerName,

Gender,

EmailId,

Phone,

DOB

)

Values

('C0011','Kratos','Male','THEspartan@gmail.com',8529639514,'06/28/1997'),

('C0001','Benjamin','Male','ben8@gmail.com',8529639514,'06/28/1992'),

('C0002','Ryan','Male','ryan1@gmail.com',8571247265,'02/07/1999'),

('C0003','Fatima','Female','fat.ima@gmail.com',9541926735,'12/01/2000'),

('C0004','Alex','Male','all@gmail.com',9025162253,'03/21/1998'),

('C0005','Monica','female','monica9@gmail.com',9083451172,'10/11/2001'),

('C0006','Rachel','Female','rachel.ra@gmail.com',6782049918,'09/09/1890'),

('C0007','Chandler','Male','chad.chandler@gmail.com',6679821190,'10/02/1997'),

('C0008','Joey','Male','joeyy@gmail.com',7895674453,'01/21/2002'),

('C0009','Ross','Male','raw\_ross@gmail.com',8769023916,'07/11/2000'),

('C0010','Emily','Female','emily@gmail.com',6667892098,'11/26/2001');

--inserting values into Suscription Table

Insert into Subscription values

('SB001','05/18/2022','05/17/2023','C0001'),

('SB002','12/10/2021','12/09/2022','C0002'),

('SB003','01/09/2023','01/08/2023','C0003'),

('SB004','10/01/2022','09/01/2023','C0004'),

('SB005','07/21/2023','07/20/2023','C0005'),

('SB006','04/05/2024','04/04/2024','C0006'),

('SB007','09/29/2024','09/28/2024','C0007'),

('SB008','08/14/2023','08/13/2023','C0008'),

('SB009','11/17/2024','11/16/2024','C0009'),

('SB010','02/02/2022','02/01/2022','C0010');

--inserting values into Offer Table

Insert into Offer Values

('OF001','Silver','15% Discount From Final Amount','11/25/2022'),

('OF002','Gold','20% Discount From Final Amount','01/12/2022'),

('OF003','Platinum','25% Discount From Final Amount','09/09/2023'),

('OF004','Silver','15% Discount From Final Amount','07/15/2023'),

('OF005','Gold','20% Discount From Final Amount','03/02/2024'),

('OF006','Platinum','25% Discount From Final Amount','04/20/2023'),

('OF007','Silver','15% Discount From Final Amount','07/11/2024'),

('OF008','Gold','20% Discount From Final Amount','02/02/2024'),

('OF009','Platinum','25% Discount From Final Amount','10/06/2024'),

('OF010','Silver','15% Discount From Final Amount','12/25/2024');

--inserting values into Booking Table -\*\*\*\*\*Booking Details\*\*\*\*\*-

Insert into Booking Values

('B0011','C0001','H0001','R0001','11-28-2022','Booked'),

('B0001','C0001','H0001','R0001','11-16-2022','Booked'),

('B0002','C0002','H0002','R0002','03-09-2022','Pay at Hotel'),

('B0003','C0003','H0003','R0003','08-21-2022','Confirmed'),

('B0004','C0004','H0004','R0004','12-01-2022','Waiting'),

('B0005','C0005','H0005','R0005','05-10-2022','Confirmed'),

('B0006','C0006','H0006','R0006','09-15-2023','Booked'),

('B0007','C0007','H0007','R0007','12-12-2022','Pay at Hotel'),

('B0008','C0008','H0008','R0008','02-20-2022','Booked'),

('B0009','C0009','H0009','R0009','01-31-2022','Confirmed'),

('B0010','C0010','H0010','R0010','10-27-2022','Confrimed');

--inserting values into Payment Table

Insert into Payment

(PaymentId,

Total,

BookingId)

Values

('P0011',3500,'B0011'),

('P0001',3000,'B0001'),

('P0002',4000,'B0002'),

('P0003',1500,'B0003'),

('P0004',1000,'B0004'),

('P0005',500,'B0005'),

('P0006',1500,'B0006'),

('P0007',600,'B0007'),

('P0008',1010,'B0008'),

('P0009',900,'B0009'),

('P0010',4000,'B0010');

--inserting values into Discount Table

Insert into Discount Values

(15,'C0001','OF001'),

(20,'C0002','OF002'),

(25,'C0003','OF003'),

(15,'C0004','OF004'),

(20,'C0005','OF005'),

(10,'C0006','OF006'),

(15,'C0007','OF007'),

(20,'C0008','OF008'),

(10,'C0009','OF009'),

(15,'C0010','OF010');

--inserting values into Booking History Table

Insert into BookingHistory Values

('BH001','B0001'),

('BH002','B0002'),

('BH003','B0003'),

('BH004','B0004'),

('BH005','B0005'),

('BH006','B0006'),

('BH007','B0007'),

('BH008','B0008'),

('BH009','B0009'),

('BH010','B0010'),

('BH011','B0001');

--inserting values into Review Table

Insert into Review Values

('C0001','H0001','good'),

('C0002','H0002','Excellent service'),

('C0003','H0003','Clean and neat'),

('C0004','H0004','Perfect Location'),

('C0005','H0005','good quality food'),

('C0006','H0006','Spcaious and good locality'),

('C0007','H0007','perfect for vacation'),

('C0008','H0008','good for family time'),

('C0009','H0009','excellent service'),

('C0010','H0010','housekeeping service is nice');

--inserting values into Room Booking Confirmation Table

Insert into RoomBookingConfirmation Values

('B0001','R0001','12/01/2022','12/03/2022'),

('B0002','R0002','02/21/2023','02/22/2023'),

('B0003','R0003','09/10/2022','09/12/2022'),

('B0004','R0004','07/29/2023','07/30/2023'),

('B0005','R0005','08/15/2023','08/18/2023'),

('B0006','R0006','10/17/2022','10/20/2022'),

('B0007','R0007','11/06/2024','11/07/2024'),

('B0008','R0008','12/31/2023','01/02/2024'),

('B0009','R0009','12/25/2022','01/02/2023'),

('B0010','R0010','03/27/2023','03/28/2023');

--inserting values into Location Type Table

Insert into LocationType Values

('C0001',02134,'Boston'),

('C0002',02215,'Boston'),

('C0003',02119,'Boston'),

('C0004',02135,'Boston'),

('C0005',10001,'New York'),

('C0006',02188,'Boston'),

('C0007',02090,'Boston'),

('C0008',02467,'Boston'),

('C0009',02474,'Boston'),

('C0010',02145,'Boston');

**SELECT**

select \* from HotelLocation

select \* from HotelType

select \* from RoomType

select \* from Hotel

select \* From Room

select \* from Customer

select \* From Subscription

select \* from offer

select \* from Booking

select \* from Payment

select \* from Discount

select \* from BookingHistory

select \* from Review

select \* from RoomBookingConfirmation

select \* from LocationType

**DELETE**

delete from LocationType

delete from RoomBookingConfirmation

delete from Review

delete from Discount

delete from Payment

delete from BookingHistory

delete from Booking

delete from Offer

delete from Subscription

delete from Customer

delete from Room

delete from Hotel

delete from RoomType

delete from HotelType

delete from HotelLocation

**VIEW**

Create View Customer\_Total\_spend as

select b.CustomerId,Sum(p.Total) as Total\_spend from BookingHistory

bh left join Payment p on bh.BookingId=p.BookingId

join Booking b on b.BookingId=bh.BookingId

group by b.CustomerId

select \* from Customer\_Total\_spend

Create View Active\_Subscriptions as

Select s.CustomerId,s.SubscriptionEndDate,s.SubscriptionId,c.EmailId

from Subscription s join Customer c on

c.CustomerId=s.CustomerId where s.SubscriptionEndDate>getdate()

select \* from Active\_Subscriptions

Create View getRoomsInBoston as

Select r.RoomId,h.HotelId,h.HotelName,h.ZipCode,hl.Area,h.AboutInfo,ht.HotelTypeDescriptions

,min(rt.PriceRange) as Minimum\_Price,min(r.Tariff) as Minimum\_Tariff

from Hotel h join HotelLocation hl on h.ZipCode=hl.ZipCode

join HotelType ht on h.HotelTypeId=ht.HotelTypeId

join Room r on r.HotelId=h.HotelId

join RoomType rt on rt.RoomTypeId=r.RoomTypeId

where hl.City='Boston' group by r.RoomId,h.HotelId,h.HotelName,h.ZipCode,hl.Area,

h.AboutInfo,ht.HotelTypeDescriptions

select \* from getRoomsInBoston

**PROCEDURE**

Create Proc Subscription\_reminder @months int

as

begin

select count(\*) as NoOfCustomersWhoNeedReminders from (

select c.CustomerId,s.SubscriptionId,s.SubscriptionEndDate,c.EmailId,

(select case when DATEDIFF(MONTH,GETDATE(),s.SubscriptionEndDate)>@months

then 'No Need for Reminder' else 'Need Reminder' End) as Reminder

from Subscription s right join Customer c

on s.CustomerId=c.CustomerId where

(select case when DATEDIFF(MONTH,GETDATE(),s.SubscriptionEndDate)>@months

then 'No Need for Reminder' else 'Need Reminder' End) = 'Need Reminder') s

select c.CustomerId,s.SubscriptionId,s.SubscriptionEndDate,c.EmailId,

(select case when DATEDIFF(MONTH,GETDATE(),s.SubscriptionEndDate)>6

then 'No Need for Reminder' else 'Need Reminder' End) as Reminder

from Subscription s right join Customer c

on s.CustomerId=c.CustomerId order by s.SubscriptionEndDate

end

exec Subscription\_reminder 6

Create Proc getBooking @customer varchar(5)

as

begin

select b.BookingId,c.CustomerName,c.EmailId,h.HotelName,h.AboutInfo,

r.RoomId,hl.ZipCode from Booking b join

RoomBookingConfirmation rb on b.BookingId=rb.BookingId

join Hotel h on h.HotelId=b.HotelId

join Room r on r.HotelId=b.HotelId

join HotelLocation hl on hl.ZipCode=h.ZipCode

join Customer c on c.CustomerId=b.CustomerId

where b.BookingId=@customer

end

exec getBooking 'B0002'

Create Proc UpdateRoomTariff @hotel varchar(5),@room varchar(5) ,@tariff int

as

begin

declare @hid varchar(5),@rid varchar(5),@p int

set @hid=@hotel

set @rid=@room

set @p=@tariff

update Room Set Tariff=@tariff where HotelId=@hid and RoomId=@room

Print('Updated Successfully')

Select \* from Room r where RoomId=@rid and HotelId=@hid

end

exec UpdateRoomTariff 'H0005','R0005',90

**ALTER PROCEDURE AND VIEW**

Alter Proc Subscription\_reminder @months int

as

begin

select count(\*) as NoOfCustomersWhoNeedReminders from (

select c.CustomerId,s.SubscriptionId,s.SubscriptionEndDate,c.EmailId,

(select case when DATEDIFF(MONTH,GETDATE(),s.SubscriptionEndDate)>@months

then 'No Need for Reminder' else 'Need Reminder' End) as Reminder

from Subscription s right join Customer c

on s.CustomerId=c.CustomerId where

(select case when DATEDIFF(MONTH,GETDATE(),s.SubscriptionEndDate)>@months

then 'No Need for Reminder' else 'Need Reminder' End) = 'Need Reminder') s

select c.CustomerId,s.SubscriptionId,s.SubscriptionEndDate,c.EmailId,

(select case when DATEDIFF(MONTH,GETDATE(),s.SubscriptionEndDate)>6

then 'No Need for Reminder' else 'Need Reminder' End) as Reminder

from Subscription s right join Customer c

on s.CustomerId=c.CustomerId order by s.SubscriptionEndDate

end

Alter Proc getBooking @customer varchar(5)

as

begin

select b.BookingId,c.CustomerName,c.EmailId,h.HotelName,h.AboutInfo,

r.RoomId,hl.ZipCode from Booking b join

RoomBookingConfirmation rb on b.BookingId=rb.BookingId

join Hotel h on h.HotelId=b.HotelId

join Room r on r.HotelId=b.HotelId

join HotelLocation hl on hl.ZipCode=h.ZipCode

join Customer c on c.CustomerId=b.CustomerId

where b.BookingId=@customer

end

Alter Proc UpdateRoomTariff @hotel varchar(5),@room varchar(5) ,@tariff int

as

begin

declare @hid varchar(5),@rid varchar(5),@p int

set @hid=@hotel

set @rid=@room

set @p=@tariff

update Room Set Tariff=@tariff where HotelId=@hid and RoomId=@room

Print('Updated Successfully')

Select \* from Room r where RoomId=@rid and HotelId=@hid

end

Alter View Customer\_Total\_spend as

select b.CustomerId,Sum(p.Total) as Total\_spend from BookingHistory

bh left join Payment p on bh.BookingId=p.BookingId

join Booking b on b.BookingId=bh.BookingId

group by b.CustomerId

Alter View Active\_Subscriptions as

Select s.CustomerId,s.SubscriptionEndDate,s.SubscriptionId,c.EmailId

from Subscription s join Customer c on

c.CustomerId=s.CustomerId where s.SubscriptionEndDate>getdate()

Alter View getRoomsInBoston as

Select r.RoomId,h.HotelId,h.HotelName,h.ZipCode,hl.Area,h.AboutInfo,ht.HotelTypeDescriptions

,min(rt.PriceRange) as Minimum\_Price,min(r.Tariff) as Minimum\_Tariff

from Hotel h join HotelLocation hl on h.ZipCode=hl.ZipCode

join HotelType ht on h.HotelTypeId=ht.HotelTypeId

join Room r on r.HotelId=h.HotelId

join RoomType rt on rt.RoomTypeId=r.RoomTypeId

where hl.City='Boston' group by r.RoomId,h.HotelId,h.HotelName,h.ZipCode,hl.Area,

h.AboutInfo,ht.HotelTypeDescriptions

**INDEX**

create nonclustered index HotelName

on Hotel (HotelName ASC);

create nonclustered index DOB

on Customer (DOB ASC);

create nonclustered index Persons

on Room (NumberOfPersons ASC);

create nonclustered index RAvailable

on Room (Availablity ASC);

**FUNCTION**

create FUNCTION dbo.FinalAmount(@Total bigint, @DiscountPercentage int)

RETURNS decimal(8,2)

AS

BEGIN

RETURN ((@Total)-((@Total\*@DiscountPercentage)/100))

END

SELECT p.PaymentId,p.BookingId,p.Total,dbo.FinalAmount(p.Total,d.DiscountPercentage) as Final\_Amount FROM

Offer o JOIN Discount d ON o.OfferId=d.OfferId JOIN Booking b ON d.CustomerId=b.CustomerId

JOIN Payment p ON b.BookingId=p.BookingId where o.OfferValidity>b.BookingDate;

CREATE FUNCTION dbo.TypeReview (

@ReviewDescription varchar(100)

)

RETURNS varchar(12) AS

BEGIN

DECLARE @return\_value varchar(12);

SET @return\_value = 'not detected';

IF (@ReviewDescription like '%good%' or @ReviewDescription like '%perfect%' or @ReviewDescription like '%excellent%') SET @return\_value = 'good review';

IF (@ReviewDescription like '%bad%' or @ReviewDescription like '%worse%' or @ReviewDescription like '%worst%') SET @return\_value = 'bad review';

RETURN @return\_value

END;

Select CustomerId,HotelId,ReviewDescription,

dbo.TypeReview(ReviewDescription) as Review\_Type From Review;

CREATE FUNCTION dbo.Age(@DOB date)

RETURNS INT AS

BEGIN

RETURN (year(getDate())-year(@DOB))

END

SELECT CustomerId,CustomerName,Gender,EmailId,Phone,DOB,dbo.Age(DOB) as AGE

FROM Customer;

Create FUNCTION dbo.StarDescriptions (

@HotelTypeDescriptions varchar(100)

)

RETURNS varchar(14) AS

BEGIN

DECLARE @return\_value varchar(14),@compare\_value int;

SET @compare\_value= SUBSTRING(@HotelTypeDescriptions,1,1)

IF (@compare\_value<=3) SET @return\_value = 'comfort hotel';

IF (@compare\_value>3 and @compare\_value<=5) SET @return\_value = 'luxury hotel';

IF (@compare\_value=7) SET @return\_value = 'royale hotel';

RETURN @return\_value

END;

SELECT h.HotelId,h.HotelName,h.AboutInfo,dbo.StarDescriptions(t.HotelTypeDescriptions) as Star\_Description

From HotelType t JOIN Hotel h ON t.HotelTypeId=h.HotelTypeId;

**ENCRYPTION**

ALTER TABLE Customer ADD ENCRYPTED\_Email VarBINARY(100);

Create Master Key Encryption by PASSWORD = 'Th!sHereIs7ust$@nd0m5tu33'

SELECT name KeyName,

symmetric\_key\_id KeyID,

key\_length KeyLength,

algorithm\_desc KeyAlgorithm

FROM sys.symmetric\_keys;

CREATE CERTIFICATE Certificate\_EmailEncrypt WITH SUBJECT = 'Protect my Email';

CREATE SYMMETRIC KEY EmailEncryptor

WITH ALGORITHM = AES\_256

ENCRYPTION BY CERTIFICATE Certificate\_EmailEncrypt;

SELECT name CertName,

certificate\_id CertID,

pvt\_key\_encryption\_type\_desc EncryptType,

issuer\_name Issuer

FROM sys.certificates;

SELECT name KeyName,

symmetric\_key\_id KeyID,

key\_length KeyLength,

algorithm\_desc KeyAlgorithm

FROM sys.symmetric\_keys;

OPEN SYMMETRIC KEY EmailEncryptor DECRYPTION BY CERTIFICATE Certificate\_EmailEncrypt;

UPDATE Customer SET ENCRYPTED\_Email = ENCRYPTBYKEY(KEY\_GUID('EmailEncryptor'), EmailId) FROM Customer;

GO

CLOSE SYMMETRIC KEY EmailEncryptor

SELECT \* FROM Customer;

**TRIGGER**

CREATE TABLE CustomerLog(

LogId INT IDENTITY PRIMARY KEY,

CustomerId VARCHAR(5),

CustomerName VARCHAR(255) NOT NULL,

LastUpdatedOn DATETIME NOT NULL,

OperationPerformed CHAR(10) NOT NULL,

CHECK(OperationPerformed = 'INSERT' or OperationPerformed='DELETE')

);

CREATE TRIGGER CustomerAuditLog

ON Customer

AFTER INSERT, DELETE

AS

BEGIN

SET NOCOUNT ON;

INSERT INTO CustomerLog(

CustomerId,

CustomerName,

LastUpdatedOn,

OperationPerformed

)

SELECT

i.CustomerId,

i.CustomerName,

GETDATE(),

'INSERT'

FROM

inserted i

END

Insert into Customer(

CustomerId,

CustomerName,

Gender,

EmailId,

Phone,

DOB

)

Values

('C0012','Vivek','Male','Vivekhana@gmail.com',8529639514,'06/28/1992')

CREATE TRIGGER EncryptionTrigger

ON Customer

AFTER INSERT

AS

BEGIN

OPEN SYMMETRIC KEY EmailEncryptor DECRYPTION BY CERTIFICATE Certificate\_EmailEncrypt;

UPDATE Customer SET ENCRYPTED\_Email = ENCRYPTBYKEY(KEY\_GUID('EmailEncryptor'), EmailId) FROM Customer;

CLOSE SYMMETRIC KEY EmailEncryptor;

END

SELECT \* FROM Customer

SELECT \* FROM CustomerLog