Oracle Database Creation Script

This script is designed to create and manage essential database views within an Oracle environment. Integrates and displays key information from tables related to customers, bookings, hotels, and rooms.

* Creates and manages essential database views within an Oracle environment.
* Integrates and displays key information from tables related to customers, bookings, hotels, and rooms.
* Combines relevant data fields to provide concise and views.

**1. Creation tables (9 tables)**

-- Create Hcustomer table

CREATE TABLE HCustomer (

IDCustomer NUMBER PRIMARY KEY,

FirstName VARCHAR2(150),

LastName VARCHAR2(150)

);

-- Create HAddress table

CREATE TABLE HAddress (

IDAddress NUMBER PRIMARY KEY,

Street VARCHAR2(100),

City VARCHAR2(100),

Province VARCHAR2(50),

Postcode VARCHAR2(10)

);

-- Create HCustomer\_Address table

CREATE TABLE HCustomer\_Address (

IDCustomer\_Address NUMBER PRIMARY KEY,

StartDate DATE,

EndDate DATE,

Customer\_IDCustomer NUMBER,

Address\_IDAddress NUMBER,

CONSTRAINT FK\_CUSTOMER FOREIGN KEY (Customer\_IDCustomer) REFERENCES HCustomer(IDCustomer),

CONSTRAINT FK\_ADDRESS FOREIGN KEY (Address\_IDAddress) REFERENCES HAddress(IDAddress)

);

-- Create HBooking table

CREATE TABLE HBooking (

IDBooking NUMBER PRIMARY KEY,

BookingDate DATE,

TotalAmount DECIMAL(10, 2),

IDCUSTOMER NUMBER,

CONSTRAINT FK\_BOOKING\_CUSTOMER FOREIGN KEY (IDCUSTOMER) REFERENCES HCUSTOMER (IDCUSTOMER)

);

--Create HPAYMENT table

CREATE TABLE HPAYMENT (

IDPayment NUMBER PRIMARY KEY,

PaymentDate Date NOT NULL,

PaymentMethod VARCHAR2(20),

Amount NUMBER(10, 2) NOT NULL

);

-- Create HBooking\_Payment table

CREATE TABLE HBooking\_Payment (

IDBooking\_Payment NUMBER PRIMARY KEY,

StartDate DATE,

EndDate DATE,

BOOKING\_IDBOOKING NUMBER,

PAYMENT\_IDPAYMENT NUMBER,

CONSTRAINT FK\_BOOKING FOREIGN KEY (BOOKING\_IDBOOKING) REFERENCES HBOOKING (IDBOOKING),

CONSTRAINT FK\_PAYMENT FOREIGN KEY (PAYMENT\_IDPAYMENT) REFERENCES HPAYMENT (IDPAYMENT)

);

-- Create HHotel table

CREATE TABLE HHotel (

IDHotel NUMBER PRIMARY KEY,

HotelName VARCHAR2(100),

BOOKING\_IDBOOKING NUMBER,

CONSTRAINT FK\_HOTEL\_BOOKING FOREIGN KEY (BOOKING\_IDBOOKING) REFERENCES HBOOKING (IDBOOKING)

);

-- Create HRoom table

CREATE TABLE HRoom (

IDRoom NUMBER PRIMARY KEY,

RoomNumber NUMBER,

RoomType VARCHAR2(50),

Hotel\_IDHotel NUMBER,

CONSTRAINT FK\_ROOM\_HOTEL FOREIGN KEY (HOTEL\_IDHOTEL) REFERENCES HHOTEL (IDHOTEL)

);

-- Create HHotel\_Room table

CREATE TABLE HHotel\_Room (

IDHotel\_Room NUMBER PRIMARY KEY,

StartDate DATE,

EndDate DATE,

Hotel\_IDHotel NUMBER,

Room\_IDRoom NUMBER,

CONSTRAINT FK\_HOTEL FOREIGN KEY (HOTEL\_IDHOTEL) REFERENCES HHOTEL (IDHOTEL),

CONSTRAINT FK\_ROOM FOREIGN KEY (ROOM\_IDROOM) REFERENCES HROOM (IDROOM)

);

-- Insert values into HCustomerName

INSERT INTO HCustomer (IDCustomer, FirstName, LastName) VALUES (1, 'John', 'Doe');

INSERT INTO HCustomer (IDCustomer, FirstName, LastName) VALUES (2, 'Jane', 'Smith');

INSERT INTO HCustomer (IDCustomer, FirstName, LastName) VALUES (3, 'Michael', 'Johnson');

INSERT INTO HCustomer (IDCustomer, FirstName, LastName) VALUES (4, 'Emily', 'Davis');

INSERT INTO HCustomer (IDCustomer, FirstName, LastName) VALUES (5, 'Chris', 'Brown');

-- Insert values into HAddress

INSERT INTO HAddress (IDAddress, Street, City, Province, Postcode) VALUES (1, '123 Main St', 'Springfield', 'IL', '62704');

INSERT INTO HAddress (IDAddress, Street, City, Province, Postcode) VALUES (2, '456 Elm St', 'Springfield', 'IL', '62705');

INSERT INTO HAddress (IDAddress, Street, City, Province, Postcode) VALUES (3, '789 Oak St', 'Springfield', 'IL', '62706');

INSERT INTO HAddress (IDAddress, Street, City, Province, Postcode) VALUES (4, '101 Maple St', 'Springfield', 'IL', '62707');

INSERT INTO HAddress (IDAddress, Street, City, Province, Postcode) VALUES (5, '202 Pine St', 'Springfield', 'IL', '62708');

2. Integration of Key Customer, Booking, and Hotel Data

* Create the sequences.
* Create historical tables.
* Create the triggers for each table.
* Perform INSERT, UPDATE, and DELETE operations on the main tables.

-- [1. Create Sequences]

-- Sequence for HCustomer

CREATE SEQUENCE HCustomer\_SEQ

START WITH 100

INCREMENT BY 1

NOCACHE

NOCYCLE;

-- Sequence for HAddress

CREATE SEQUENCE HAddress\_SEQ

START WITH 100

INCREMENT BY 1

NOCACHE

NOCYCLE;

-- Sequence for HCustomer\_Address

CREATE SEQUENCE HCustomer\_Address\_SEQ

START WITH 100

INCREMENT BY 1

NOCACHE

NOCYCLE;

-- Sequence for HBooking

CREATE SEQUENCE HBooking\_SEQ

START WITH 100

INCREMENT BY 1

NOCACHE

NOCYCLE;

-- Sequence for HPAYMENT

CREATE SEQUENCE HPAYMENT\_SEQ

START WITH 100

INCREMENT BY 1

NOCACHE

NOCYCLE;

-- Sequence for HBooking\_Payment

CREATE SEQUENCE HBooking\_Payment\_SEQ

START WITH 100

INCREMENT BY 1

NOCACHE

NOCYCLE;

-- Sequence for HHotel

CREATE SEQUENCE HHotel\_SEQ

START WITH 100

INCREMENT BY 1

NOCACHE

NOCYCLE;

-- Sequence for HRoom

CREATE SEQUENCE HRoom\_SEQ

START WITH 100

INCREMENT BY 1

NOCACHE

NOCYCLE;

-- Sequence for HHotel\_Room

CREATE SEQUENCE HHotel\_Room\_SEQ

START WITH 100

INCREMENT BY 1

NOCACHE

NOCYCLE;

-- [2. Creat Historical Tables]

-- Historical table for HCustomer

CREATE TABLE HCustomer\_History (

HistoryID NUMBER PRIMARY KEY,

IDCustomer NUMBER,

FirstName VARCHAR2(150),

LastName VARCHAR2(150),

StartTime DATE,

EndTime DATE

);

-- Historical table for HAddress

CREATE TABLE HAddress\_History (

HistoryID NUMBER PRIMARY KEY,

IDAddress NUMBER,

Street VARCHAR2(100),

City VARCHAR2(100),

Province VARCHAR2(50),

Postcode VARCHAR2(10),

StartTime DATE,

EndTime DATE

);

-- Historical table for HCustomer\_Address

CREATE TABLE HCustomer\_Address\_History (

HistoryID NUMBER PRIMARY KEY,

IDCustomer\_Address NUMBER,

StartDate DATE,

EndDate DATE,

Customer\_IDCustomer NUMBER,

Address\_IDAddress NUMBER,

StartTime DATE,

EndTime DATE

);

-- Historical table for HBooking

CREATE TABLE HBooking\_History (

HistoryID NUMBER PRIMARY KEY,

IDBooking NUMBER,

BookingDate DATE,

TotalAmount DECIMAL(10, 2),

IDCUSTOMER NUMBER,

StartTime DATE,

EndTime DATE

);

-- Historical table for HPAYMENT

CREATE TABLE HPAYMENT\_History (

HistoryID NUMBER PRIMARY KEY,

IDPayment NUMBER,

PaymentDate Date,

PaymentMethod VARCHAR2(20),

Amount NUMBER(10, 2),

StartTime DATE,

EndTime DATE

);

-- Historical table for HBooking\_Payment

CREATE TABLE HBooking\_Payment\_History (

HistoryID NUMBER PRIMARY KEY,

IDBooking\_Payment NUMBER,

StartDate DATE,

EndDate DATE,

BOOKING\_IDBOOKING NUMBER,

PAYMENT\_IDPAYMENT NUMBER,

StartTime DATE,

EndTime DATE

);

-- Historical table for HHotel

CREATE TABLE HHotel\_History (

HistoryID NUMBER PRIMARY KEY,

IDHotel NUMBER,

HotelName VARCHAR2(100),

BOOKING\_IDBOOKING NUMBER,

StartTime DATE,

EndTime DATE

);

-- Historical table for HRoom

CREATE TABLE HRoom\_History (

HistoryID NUMBER PRIMARY KEY,

IDRoom NUMBER,

RoomNumber NUMBER,

RoomType VARCHAR2(50),

Hotel\_IDHotel NUMBER,

StartTime DATE,

EndTime DATE

);

-- Historical table for HHotel\_Room

CREATE TABLE HHotel\_Room\_History (

HistoryID NUMBER PRIMARY KEY,

IDHotel\_Room NUMBER,

StartDate DATE,

EndDate DATE,

Hotel\_IDHotel NUMBER,

Room\_IDRoom NUMBER,

StartTime DATE,

EndTime DATE

);

-- [3. Creating Triggers]

CREATE OR REPLACE TRIGGER trg\_HCustomer\_HISTORY

BEFORE INSERT OR UPDATE OR DELETE ON HCustomer

FOR EACH ROW

BEGIN

IF INSERTING THEN

INSERT INTO HCustomer\_History (

HistoryID, IDCustomer, FirstName, LastName, StartTime, EndTime

) VALUES (

HCustomer\_SEQ.NEXTVAL, :NEW.IDCustomer, :NEW.FirstName, :NEW.LastName, SYSDATE, NULL

);

ELSIF UPDATING THEN

UPDATE HCustomer\_History

SET EndTime = SYSDATE

WHERE IDCustomer = :OLD.IDCustomer

AND EndTime IS NULL;

INSERT INTO HCustomer\_History (

HistoryID, IDCustomer, FirstName, LastName, StartTime, EndTime

) VALUES (

HCustomer\_SEQ.NEXTVAL, :NEW.IDCustomer, :NEW.FirstName, :NEW.LastName, SYSDATE, NULL

);

ELSIF DELETING THEN

UPDATE HCustomer\_History

SET EndTime = SYSDATE

WHERE IDCustomer = :OLD.IDCustomer

AND EndTime IS NULL;

END IF;

END;

/

-- [Trigger for HAddress]

CREATE OR REPLACE TRIGGER trg\_HAddress\_HISTORY

BEFORE INSERT OR UPDATE OR DELETE ON HAddress

FOR EACH ROW

BEGIN

IF INSERTING THEN

INSERT INTO HAddress\_History (

HistoryID, IDAddress, Street, City, Province, Postcode, StartTime, EndTime

) VALUES (

HAddress\_SEQ.NEXTVAL, :NEW.IDAddress, :NEW.Street, :NEW.City, :NEW.Province, :NEW.Postcode, SYSDATE, NULL

);

ELSIF UPDATING THEN

UPDATE HAddress\_History

SET EndTime = SYSDATE

WHERE IDAddress = :OLD.IDAddress

AND EndTime IS NULL;

INSERT INTO HAddress\_History (

HistoryID, IDAddress, Street, City, Province, Postcode, StartTime, EndTime

) VALUES (

HAddress\_SEQ.NEXTVAL, :NEW.IDAddress, :NEW.Street, :NEW.City, :NEW.Province, :NEW.Postcode, SYSDATE, NULL

);

ELSIF DELETING THEN

UPDATE HAddress\_History

SET EndTime = SYSDATE

WHERE IDAddress = :OLD.IDAddress

AND EndTime IS NULL;

END IF;

END;

/

-- [Trigger for HCustomer\_Address]

CREATE OR REPLACE TRIGGER trg\_HCustomer\_Address\_HISTORY

BEFORE INSERT OR UPDATE OR DELETE ON HCustomer\_Address

FOR EACH ROW

BEGIN

IF INSERTING THEN

INSERT INTO HCustomer\_Address\_History (

HistoryID, IDCustomer\_Address, StartDate, EndDate, Customer\_IDCustomer, Address\_IDAddress, StartTime, EndTime

) VALUES (

HCustomer\_Address\_SEQ.NEXTVAL, :NEW.IDCustomer\_Address, :NEW.StartDate, :NEW.EndDate, :NEW.Customer\_IDCustomer, :NEW.Address\_IDAddress, SYSDATE, NULL

);

ELSIF UPDATING THEN

UPDATE HCustomer\_Address\_History

SET EndTime = SYSDATE

WHERE IDCustomer\_Address = :OLD.IDCustomer\_Address

AND EndTime IS NULL;

INSERT INTO HCustomer\_Address\_History (

HistoryID, IDCustomer\_Address, StartDate, EndDate, Customer\_IDCustomer, Address\_IDAddress, StartTime, EndTime

) VALUES (

HCustomer\_Address\_SEQ.NEXTVAL, :NEW.IDCustomer\_Address, :NEW.StartDate, :NEW.EndDate, :NEW.Customer\_IDCustomer, :NEW.Address\_IDAddress, SYSDATE, NULL

);

ELSIF DELETING THEN

UPDATE HCustomer\_Address\_History

SET EndTime = SYSDATE

WHERE IDCustomer\_Address = :OLD.IDCustomer\_Address

AND EndTime IS NULL;

END IF;

END;

/

-- [Trigger for HBooking]

CREATE OR REPLACE TRIGGER trg\_HBooking\_HISTORY

BEFORE INSERT OR UPDATE OR DELETE ON HBooking

FOR EACH ROW

BEGIN

IF INSERTING THEN

INSERT INTO HBooking\_History (

HistoryID, IDBooking, BookingDate, TotalAmount, IDCUSTOMER, StartTime, EndTime

) VALUES (

HBooking\_SEQ.NEXTVAL, :NEW.IDBooking, :NEW.BookingDate, :NEW.TotalAmount, :NEW.IDCUSTOMER, SYSDATE, NULL

);

ELSIF UPDATING THEN

-- Update EndTime for the existing record before inserting a new history record

UPDATE HBooking\_History

SET EndTime = SYSDATE

WHERE IDBooking = :OLD.IDBooking

AND EndTime IS NULL;

-- Insert new history record

INSERT INTO HBooking\_History (

HistoryID, IDBooking, BookingDate, TotalAmount, IDCUSTOMER, StartTime, EndTime

) VALUES (

HBooking\_SEQ.NEXTVAL, :NEW.IDBooking, :NEW.BookingDate, :NEW.TotalAmount, :NEW.IDCUSTOMER, SYSDATE, NULL

);

ELSIF DELETING THEN

-- Update EndTime for the record being deleted

UPDATE HBooking\_History

SET EndTime = SYSDATE

WHERE IDBooking = :OLD.IDBooking

AND EndTime IS NULL;

END IF;

END;

/

-- [Trigger for HPAYMENT]

CREATE OR REPLACE TRIGGER trg\_HPAYMENT\_HISTORY

BEFORE INSERT OR UPDATE OR DELETE ON HPAYMENT

FOR EACH ROW

BEGIN

IF INSERTING THEN

INSERT INTO HPAYMENT\_History (

HistoryID, IDPayment, PaymentDate, PaymentMethod, Amount, StartTime, EndTime

) VALUES (

HPAYMENT\_SEQ.NEXTVAL, :NEW.IDPayment, :NEW.PaymentDate, :NEW.PaymentMethod, :NEW.Amount, SYSDATE, NULL

);

ELSIF UPDATING THEN

UPDATE HPAYMENT\_History

SET EndTime = SYSDATE

WHERE IDPayment = :OLD.IDPayment

AND EndTime IS NULL;

INSERT INTO HPAYMENT\_History (

HistoryID, IDPayment, PaymentDate, PaymentMethod, Amount, StartTime, EndTime

) VALUES (

HPAYMENT\_SEQ.NEXTVAL, :NEW.IDPayment, :NEW.PaymentDate, :NEW.PaymentMethod, :NEW.Amount, SYSDATE, NULL

);

ELSIF DELETING THEN

UPDATE HPAYMENT\_History

SET EndTime = SYSDATE

WHERE IDPayment = :OLD.IDPayment

AND EndTime IS NULL;

END IF;

END;

/

-- [Trigger for HBooking\_Payment]

CREATE OR REPLACE TRIGGER trg\_HBooking\_Payment\_HISTORY

BEFORE INSERT OR UPDATE OR DELETE ON HBooking\_Payment

FOR EACH ROW

BEGIN

IF INSERTING THEN

INSERT INTO HBooking\_Payment\_History (

HistoryID, IDBooking\_Payment, StartDate, EndDate, BOOKING\_IDBOOKING, PAYMENT\_IDPAYMENT, StartTime, EndTime

) VALUES (

HBooking\_Payment\_SEQ.NEXTVAL, :NEW.IDBooking\_Payment, :NEW.StartDate, :NEW.EndDate, :NEW.BOOKING\_IDBOOKING, :NEW.PAYMENT\_IDPAYMENT, SYSDATE, NULL

);

ELSIF UPDATING THEN

UPDATE HBooking\_Payment\_History

SET EndTime = SYSDATE

WHERE IDBooking\_Payment = :OLD.IDBooking\_Payment

AND EndTime IS NULL;

INSERT INTO HBooking\_Payment\_History (

HistoryID, IDBooking\_Payment, StartDate, EndDate, BOOKING\_IDBOOKING, PAYMENT\_IDPAYMENT, StartTime, EndTime

) VALUES (

HBooking\_Payment\_SEQ.NEXTVAL, :NEW.IDBooking\_Payment, :NEW.StartDate, :NEW.EndDate, :NEW.BOOKING\_IDBOOKING, :NEW.PAYMENT\_IDPAYMENT, SYSDATE, NULL

);

ELSIF DELETING THEN

UPDATE HBooking\_Payment\_History

SET EndTime = SYSDATE

WHERE IDBooking\_Payment = :OLD.IDBooking\_Payment

AND EndTime IS NULL;

END IF;

END;

/

-- [Trigger for HHotel]

CREATE OR REPLACE TRIGGER trg\_HHotel\_HISTORY

BEFORE INSERT OR UPDATE OR DELETE ON HHotel

FOR EACH ROW

BEGIN

IF INSERTING THEN

INSERT INTO HHotel\_History (

HistoryID, IDHotel, HotelName, BOOKING\_IDBOOKING, StartTime, EndTime

) VALUES (

HHotel\_SEQ.NEXTVAL, :NEW.IDHotel, :NEW.HotelName, :NEW.BOOKING\_IDBOOKING, SYSDATE, NULL

);

ELSIF UPDATING THEN

UPDATE HHotel\_History

SET EndTime = SYSDATE

WHERE IDHotel = :OLD.IDHotel

AND EndTime IS NULL;

INSERT INTO HHotel\_History (

HistoryID, IDHotel, HotelName, BOOKING\_IDBOOKING, StartTime, EndTime

) VALUES (

HHotel\_SEQ.NEXTVAL, :NEW.IDHotel, :NEW.HotelName, :NEW.BOOKING\_IDBOOKING, SYSDATE, NULL

);

ELSIF DELETING THEN

UPDATE HHotel\_History

SET EndTime = SYSDATE

WHERE IDHotel = :OLD.IDHotel

AND EndTime IS NULL;

END IF;

END;

/

-- [Trigger for HRoom]

CREATE OR REPLACE TRIGGER trg\_HRoom\_HISTORY

BEFORE INSERT OR UPDATE OR DELETE ON HRoom

FOR EACH ROW

BEGIN

IF INSERTING THEN

INSERT INTO HRoom\_History (

HistoryID, IDRoom, RoomNumber, RoomType, Hotel\_IDHotel, StartTime, EndTime

) VALUES (

HRoom\_SEQ.NEXTVAL, :NEW.IDRoom, :NEW.RoomNumber, :NEW.RoomType, :NEW.Hotel\_IDHotel, SYSDATE, NULL

);

ELSIF UPDATING THEN

UPDATE HRoom\_History

SET EndTime = SYSDATE

WHERE IDRoom = :OLD.IDRoom

AND EndTime IS NULL;

INSERT INTO HRoom\_History (

HistoryID, IDRoom, RoomNumber, RoomType, Hotel\_IDHotel, StartTime, EndTime

) VALUES (

HRoom\_SEQ.NEXTVAL, :NEW.IDRoom, :NEW.RoomNumber, :NEW.RoomType, :NEW.Hotel\_IDHotel, SYSDATE, NULL

);

ELSIF DELETING THEN

UPDATE HRoom\_History

SET EndTime = SYSDATE

WHERE IDRoom = :OLD.IDRoom

AND EndTime IS NULL;

END IF;

END;

/

-- [Trigger for HHotel\_Room]

CREATE OR REPLACE TRIGGER trg\_HHotel\_Room\_HISTORY

BEFORE INSERT OR UPDATE OR DELETE ON HHotel\_Room

FOR EACH ROW

BEGIN

IF INSERTING THEN

INSERT INTO HHotel\_Room\_History (

HistoryID, IDHotel\_Room, StartDate, EndDate, Hotel\_IDHotel, Room\_IDRoom, StartTime, EndTime

) VALUES (

HHotel\_Room\_SEQ.NEXTVAL, :NEW.IDHotel\_Room, :NEW.StartDate, :NEW.EndDate, :NEW.Hotel\_IDHotel, :NEW.Room\_IDRoom, SYSDATE, NULL

);

ELSIF UPDATING THEN

UPDATE HHotel\_Room\_History

SET EndTime = SYSDATE

WHERE IDHotel\_Room = :OLD.IDHotel\_Room

AND EndTime IS NULL;

INSERT INTO HHotel\_Room\_History (

HistoryID, IDHotel\_Room, StartDate, EndDate, Hotel\_IDHotel, Room\_IDRoom, StartTime, EndTime

) VALUES (

HHotel\_Room\_SEQ.NEXTVAL, :NEW.IDHotel\_Room, :NEW.StartDate, :NEW.EndDate, :NEW.Hotel\_IDHotel, :NEW.Room\_IDRoom, SYSDATE, NULL

);

ELSIF DELETING THEN

UPDATE HHotel\_Room\_History

SET EndTime = SYSDATE

WHERE IDHotel\_Room = :OLD.IDHotel\_Room

AND EndTime IS NULL;

END IF;

END;

/

**3. Concise View Creation**

--[1. HCUSTOMER\_ADDRESS\_VIEW]

CREATE OR REPLACE FORCE EDITIONABLE VIEW "KYUNGAUSER"."HCUSTOMER\_ADDRESS\_VIEW" ("IDCUSTOMER", "CUSTOMERNAME", "FULLADDRESS", "ADDRESSDURATION") AS

SELECT

HCustomer.IDCustomer,

HCustomer.FirstName || ' ' || HCustomer.LastName AS CustomerName,

HAddress.Street || ', ' || HAddress.City || ', ' || HAddress.Province || ', ' || HAddress.Postcode AS FullAddress,

TO\_CHAR(HCustomer\_Address.StartDate, 'YYYY-MM-DD') || ' to ' ||

COALESCE(TO\_CHAR(HCustomer\_Address.EndDate, 'YYYY-MM-DD'), 'Present') AS AddressDuration

FROM

HCustomer

JOIN

HCustomer\_Address ON HCustomer.IDCustomer = HCustomer\_Address.Customer\_IDCustomer

JOIN

HAddress ON HCustomer\_Address.Address\_IDAddress = HAddress.IDAddress;

--[2. View for HHotel and HHotel\_Room]

CREATE OR REPLACE FORCE EDITIONABLE VIEW "KYUNGAUSER"."HOTEL\_ROOM\_VIEW" ("IDHOTEL", "HOTELROOMINFO", "ROOMDURATION") AS

SELECT

HHotel.IDHotel,

HHotel.HotelName || ' - Room ' || HRoom.RoomNumber || ' (' || HRoom.RoomType || ')' AS HotelRoomInfo,

TO\_CHAR(HHotel\_Room.StartDate, 'YYYY-MM-DD') || ' to ' ||

COALESCE(TO\_CHAR(HHotel\_Room.EndDate, 'YYYY-MM-DD'), 'Present') AS RoomDuration

FROM

HHotel

JOIN

HHotel\_Room ON HHotel.IDHotel = HHotel\_Room.Hotel\_IDHotel

JOIN

HRoom ON HHotel\_Room.Room\_IDRoom = HRoom.IDRoom;

-- [3. View for HCustomer and HCustomer\_Booking]

CREATE VIEW Customer\_Booking\_View AS

SELECT

HCustomer.IDCustomer,

HCustomer.FirstName,

HCustomer.LastName,

HBooking.BookingDate,

HBooking.TotalAmount,

HCustomer\_Booking.StartDate,

HCustomer\_Booking.EndDate,

FROM

HCustomer

JOIN

HCustomer\_Booking ON HCustomer.IDCustomer = HCustomer\_Booking.Customer\_IDCustomer

JOIN

HBooking ON HCustomer\_Booking.Booking\_IDBooking = HBooking.IDBooking

-- WHERE HCustomer\_Booking.EndDate IS NULL;