Database Design

Project – Airbnb
CS 6360.001

Team Number -32

Team Members:
Saicharan Reddy Ragannagari SXR210079
Heenisha Reddy Bachugudem HXR210022

Project Description:

Using a service called Airbnb, which stands for "Air Bed and Breakfast," property owners may rent out their rooms to visitors looking for a place to stay. Travelers have the option of renting a communal area with individual rooms, a space for several people to share, or the entire property for themselves.

Brian Chesky and Joe Gebbia, two industrial designers who had just relocated to San Francisco, founded Airbnb in 2008. The couple decided to make up the money they needed by renting out their apartment to people who could not find motels to stay at while attending local trade exhibits because they couldn't afford the rent for their loft at the time. They provided air mattresses for their visitors to sleep on in the apartment's living room, and they prepared a fresh breakfast each morning. Since then, Airbnb has emerged as a pioneer in the peer-to-peer leasing of real estate.

How does it work?

Airbnb is generally used for the customers who are commuting from one place to another and need a place to take a rest. Users start by exploring the nearby places for their stay by filtering the contents like user experience, pet friendly, free breakfast, amenities which narrow the options.

Once the user is satisfied with the hotel requirements they will go ahead and book their stay by entering the check-in and check-out date along with the type of room required. While booking the room the customer also checks the cancellation policy. The booking may be prepaid reservation (Customer pays for the room before the check-in) or the postpaid reservation (Customer pays the amount to the front desk during their check-in). Once the customer is satisfied with all the fields, they will complete their reservation and Airbnb will receive the booking details of the customer.

Important Entities:

Airbnb Owner and **User** are the two important components in the domain. Those components are required to develop our Database model.

Airbnb- it is a hotel business which is based on hospitality for the customers. The reservations made by the customers are seen on the Airbnb websites and they will allocate the room based upon the customer’s reservation.

Customer- For the customer to check-in after the reservation booking, they should be minimum of 18years old, and they should possess a valid ID. They will make a reservation either using the Airbnb official site or by using the third-party websites.

Project Requirements:

User:

- A user entity is formed, where a user can be either the employee working in Airbnb company or the customer who wants to make a reservation. They login to the website using their credentials either as a host or as a customer.

Airbnb Organization:

- Airbnb is a huge organization which has many branches all over the world with thousands of employees working in it. Each employee has a different login credentials where they can see the reservations made for their branch.
- The Airbnb host can see the booking received in all the branches
- It receives the payment paid by the customer during the reservation.

Airbnb Property

- Airbnb property is divided into multiple types- house, apartment, service apartment, bungalows, guest suites, villas, guesthouse.
- The availability of the property is available in the database and the user can see the availability on the website and book accordingly.
- Each property might be different from one another, some may contain swimming pools, complimentary breakfast, parking, kitchen, smoking rooms etc.

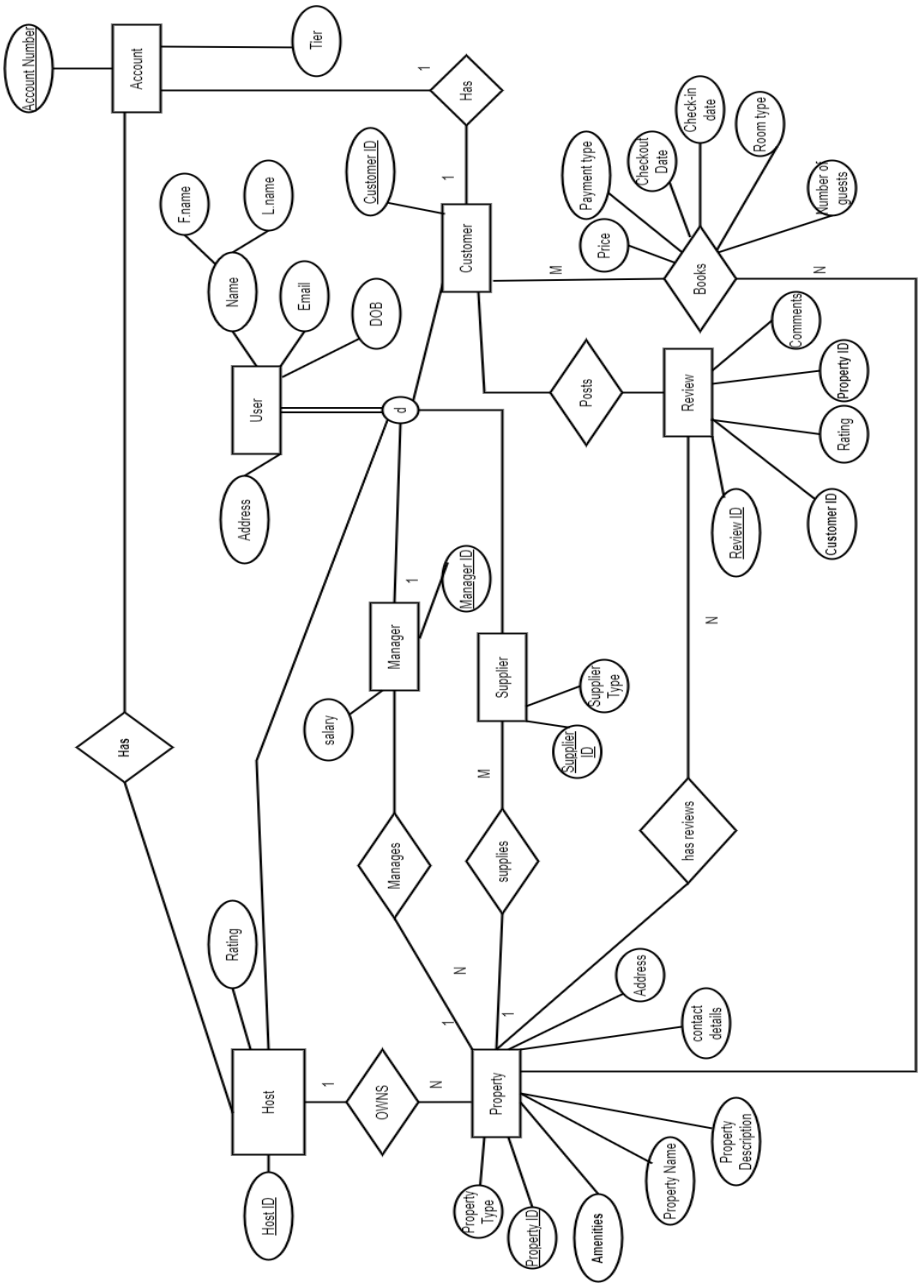
Booking:

- Whenever the user makes a booking, a new table is created which contains the user id along with the customer id. It also contains the details like check-in and check-out date along with the type of property a customer reserved.
- A customer can make many bookings and a property can also have multiple bookings.
- Generally, a booking is tied to the customer and the property.
- Once the booking is made, the total amount which also includes the tax will update in the table
- Cancellable bookings can be cancelled before the check-in date without any cancellation charges, non-cancellable bookings cannot be cancelled, and the customer will not receive the refund even though he does not check-in.

Review/Experience:

- The customer will post a review of their stay and the experience they had during their stay on the website.
- Each property will have multiple reviews, each review is posted by the different by the different customers.
- Customers can see the reviews while booking the room.

User EER Diagram



Specifications:

- The main element of the system will be the PERSON. There are 4 PERSONs under this category with their own unique ADDRESS, NAME (FIRST_NAME and LAST_NAME), DATE_OF_BIRTH, EMAIL
- The first one among them is the HOST. A HOST owns several PROPERTY and has his own unique HOST_ID and RATING.
- Each PROPERTY has its own unique PROPERTY_TYPE, PROPERTY_ID, PROPERTY_NAME, CONTACT_DETAILS, ADDRESS.
- The second one among PERSON is the MANAGER. The MANAGER manages the property and has his/her own unique MANAGER_ID and SALARY.
- The third among PERSON is the SUPPLIER. Based on the PROPERTY needs a SUPPLIER TYPE is chosen, which has its own unique SUPPLIER ID and that SUPPLIER supplies to the property.
- The fourth and the main among the PERSON is the CUSTOMER. A CUSTOMER has a unique CUSTOMER_ID and an ACCOUNT. This ACCOUNT has an ACCOUNT_NUMBER and TIER type. Tier can be something like platinum, gold or silver membership.
- The CUSTOMER books a PROPERTY, and the booking specifications are PRICE, PAYMENT_TYPE, CHECK_IN_DATE, CHECK_OUT_DATE, ROOM_TYPE, NUMBER_OF_GUESTS.
- After the CUSTOMERs stay at the PROPERTY, the CUSTOMER can share his experience in the form of a REVIEW. Each REVIEW has its own unique REVIEW_ID, RATING and COMMENTS.

Assumptions:

- One host will have many properties under his name
- Property will different property types depending on the customer's request.
- Supplier will supply the supplies to the property depending upon their requirement.
- Property can be booked by the customer based upon his requirements.
- Each customer can have only one account for the bookings
- Booking payments can either be prepaid or postpaid
- Employee salary is based on the area and position he is working in.
- Each Property of same type has a different property ID, and it is associated with the host.

Relations:

One to One binary Relations

- Each Customer can have one account. An account belongs to only one customer. (1:1)
- Each property has only one manager and Each manager can manage only one Property (1:1)

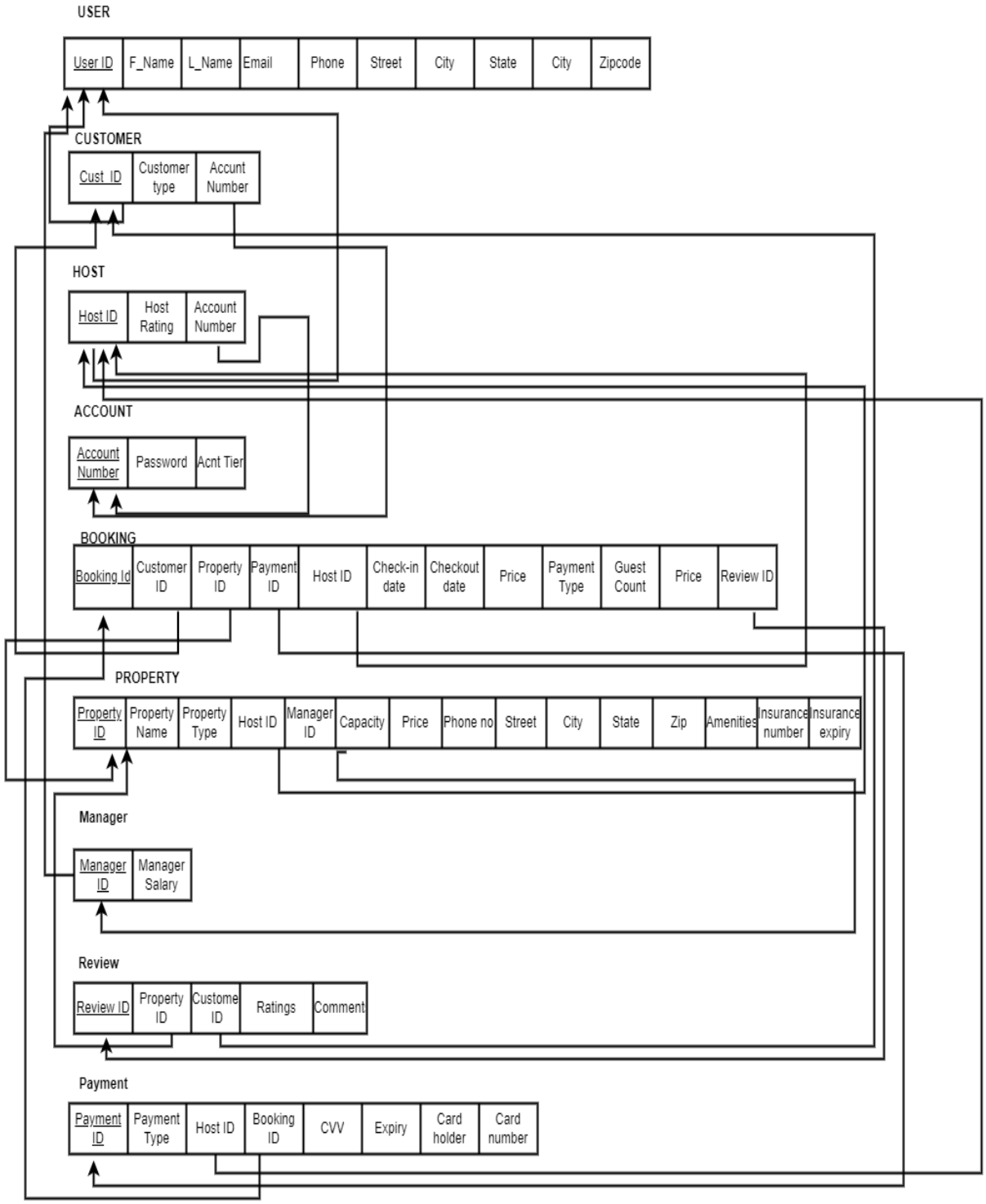
One to Many binary Relations:

- Each Host can have multiple properties. Each property will have only one owner (1: N)
- Each property can have multiple reviews and one customer can give multiple reviews to one property (1: N)

Many to Many binary Relations:

- Each customer can make multiple bookings and each property can have multiple bookings (M: N)
- Each property can have multiple suppliers and each supplier can also supply for the multiple properties (M: N)

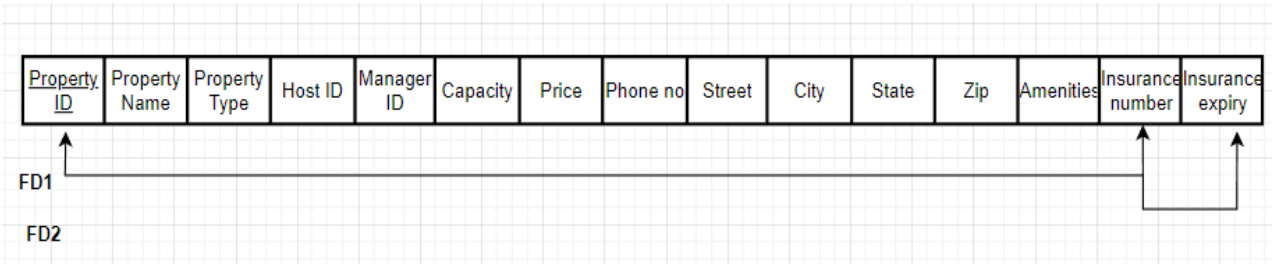
MAPZIPCODEG EER DIAGRAM INTO RELATIONAL SCHEMA:

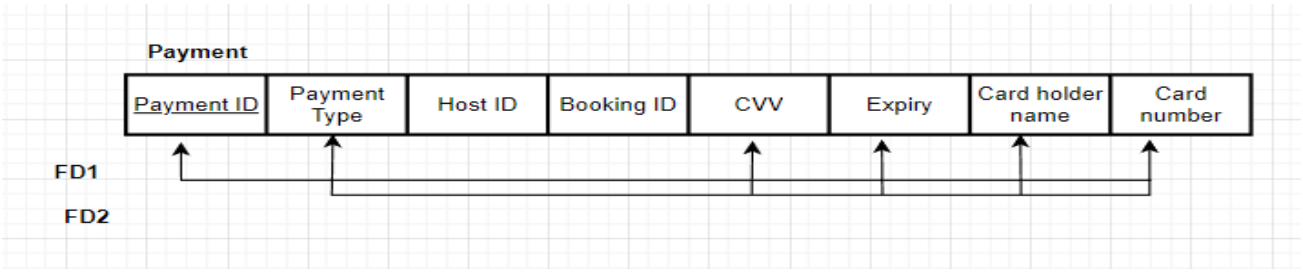


FUNCTIONAL DEPENDENCIES AND NORMALIZATION:

All our tables contain atomic values and there exists no partial dependency on the tables. Hence, our schema is **already in 1NF and 2NF.**

FUNCTIONAL DEPENDENCIES





> **Property { Property_ID,** *Host_ID,* property name, property type, capacity, price, phone, street, city, state, zip, insurance number, insurance expiry**}**

FD1: **Property_ID-->** Insurance number
FD2: Insurance number--> Insurance expiry date

Here FD2 violates 3NF as there exists a transitive dependency.

The new table is
Property { Property_ID, *Host_ID,* property_name, property_type, capacity, price, phone, street, city, state, zip, Insurance info**}**
Insurance_info{ Insurance_no, Insurance_exp_date**}**

Payment {PID, Payment type, *Host ID, booking ID*, Card_no, CVV, Card_holder, Expiry**}**

FD1: PDI -----> Card_no
FD2: card_no---> CVV, Expiry, Payment_Type, Card_holder

Here FD2 violates 3NF as there exists a transitive dependency.

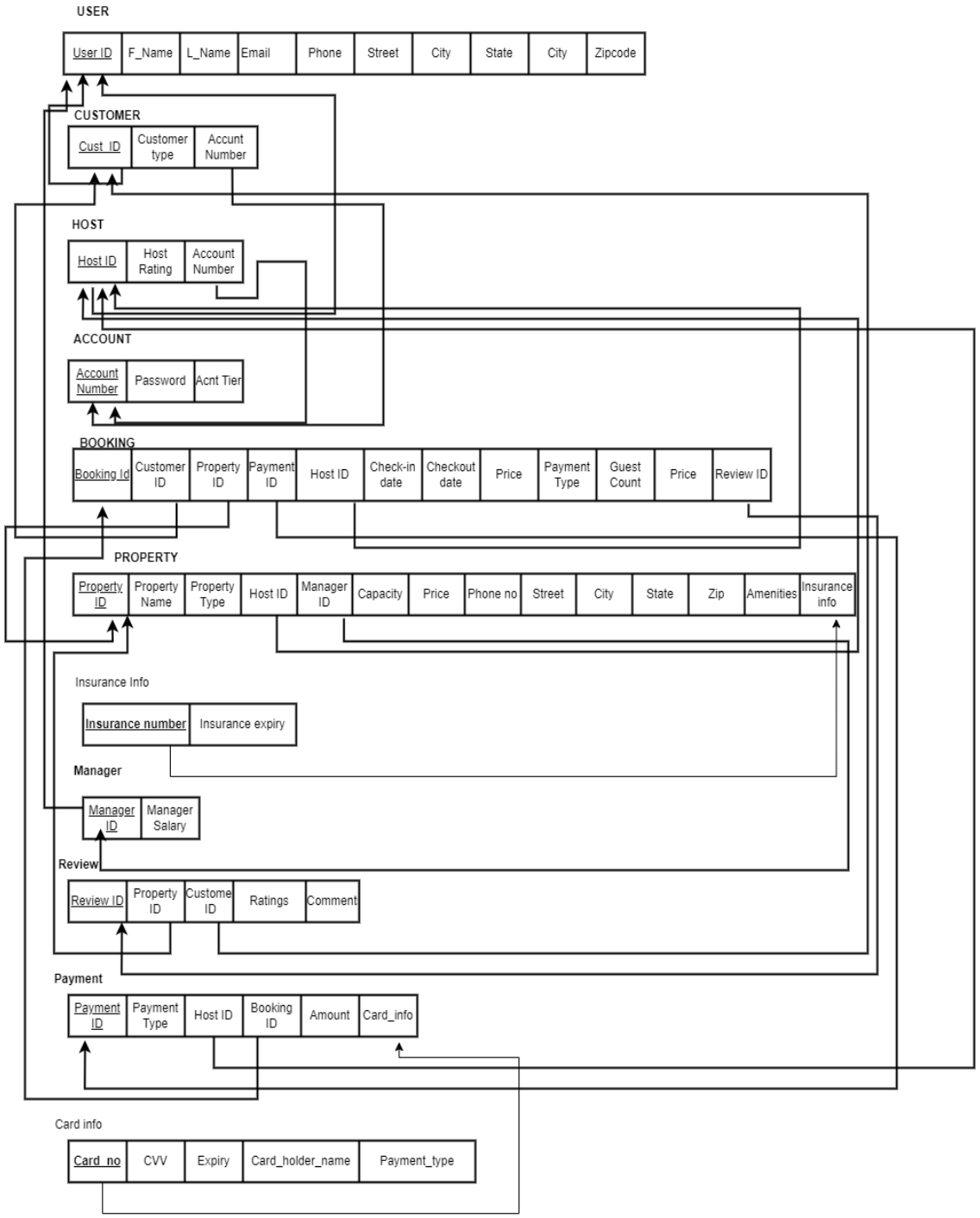
So, the new tables are:

Payment {PID, Payment type, *Host ID, booking ID*, Card_info**}**

Card_info { Card_no, CVV, Card_holder, Expiry**}**

*Primary Key: **Bold**
*****Foreign Key: Italics and underlined

FINAL RELATIONAL SCHEMA AFTER NORMALIZATION:



SQL CODE FOR CREATE TABLE & INSERT VALUES:

User:

```
CREATE TABLE Airbnb_USER
(
USER_ID INT NOT NULL,
F_name VARCHAR(50) NOT NULL,
L_Name VARCHAR(50) NOT NULL,
Ph_no INT NOT NULL,
Email VARCHAR(50) NOT NULL,
Street VARCHAR(50) NOT NULL,
City VARCHAR(30) NOT NULL,
State_name VARCHAR(30),
Zipcode INT NOT NULL,
PRIMARY KEY(User_ID) );


//INSERTING VALUES INTO THE TABLES

INSERT INTO AIRBNB_USER(USER_ID, F_name, L_Name, Ph_no, Email, Street, City,
State_name, Zipcode) VALUES(1111, 'Jim','Williams',1324567890, 'jw@gmail.com',
'Coit','Dallas','TX',78945);

INSERT INTO AIRBNB_USER(USER_ID, F_name, L_Name, Ph_no, Email, Street, City,
State_name, Zipcode) VALUES(2222, 'Jim','Williams',1324567890, 'jw@gmail.com',
'Coit','Dallas','TX',78945);

INSERT INTO AIRBNB_USER(USER_ID, F_name, L_Name, Ph_no, Email, Street, City,
State_name, Zipcode) VALUES(3333, 'Jim','Williams',1324567890, 'jw@gmail.com',
'Coit','Dallas','TX',78945);

INSERT INTO AIRBNB_USER(USER_ID, F_name, L_Name, Ph_no, Email, Street, City,
State_name, Zipcode) VALUES(4444, 'Jim','Williams',4132467890, 'jw@gmail.com',
'Coit','Dallas','TX',78945);

INSERT INTO AIRBNB_USER(USER_ID, F_name, L_Name, Ph_no, Email, Street, City,
State_name, Zipcode) VALUES(5555, 'Jim','Williams',1324567890, 'jw@gmail.com',
'Coit','Dallas','TX',78945);
```

INSERT INTO AIRBNB_USER(USER_ID, F_name, L_Name, Ph_no, Email, Street, City, State_name, Zipcode) VALUES(0147, 'Jim','Williams',1324567890, 'jw@gmail.com', 'Coit','Dallas','TX',78945);

INSERT INTO AIRBNB_USER(USER_ID, F_name, L_Name, Ph_no, Email, Street, City, State_name, Zipcode) VALUES(4656, 'Julie','Swan', 7418529630, 'js@gmail.com','Frankford','Austin', 'TX', 45678);

INSERT INTO AIRBNB_USER(USER_ID, F_name, L_Name, Ph_no, Email, Street, City, State_name, Zipcode) VALUES(8520, 'Joey','Buffay',2583691470, 'jb@gmail.com','Campbell','Richardson','TX', 15926);

INSERT INTO AIRBNB_USER(USER_ID, F_name, L_Name, Ph_no, Email, Street, City, State_name, Zipcode) VALUES(4928,'John','Smith', 3216549870, 'js@gmail.com','Ricky','Blach Springs','TX',59268);

INSERT INTO AIRBNB_USER(USER_ID, F_name, L_Name, Ph_no, Email, Street, City, State_name, Zipcode) VALUES(2598, 'Jack', 'Chandler', 3698521479, 'jc@gmail.com','Frankford','Green Lake','TX',35786);

INSERT INTO AIRBNB_USER(USER_ID, F_name, L_Name, Ph_no, Email, Street, City, State_name, Zipcode) VALUES(4859, 'George', 'Matt',2581472580, 'gm@gmail.com', 'Campbell', 'Austin','TX', 24356);

Account:

```
CREATE TABLE Account
(
Account_No INT NOT NULL,
Password VARCHAR(20) NOT NULL,
Account_Tier VARCHAR(20) NOT NULL,
);
```

//INSERTING VALUES INTO THE TABLES

INSERT INTO Account(Account_no, Password, Account_Tier) Values (25415, ‘Qwerty’, ‘Gold’);
INSERT INTO Account(Account_no, Password, Account_Tier) Values (25416, ‘Qwerty1’, ‘Blue’);
INSERT INTO Account(Account_no, Password, Account_Tier) Values (25417, ‘Qwerty2’, ‘Diamond’);

INSERT INTO Account(Account_no, Password, Account_Tier) Values (25418, ‘Qwerty3’, ‘Platinum’);

INSERT INTO Account(Account_no, Password, Account_Tier) Values (25419, ‘Qwerty4’, ‘Gold’);

INSERT INTO Account(Account_no, Password, Account_Tier) Values (25421, ‘Qwerty5’, ‘Platinum’);

INSERT INTO Account(Account_no, Password, Account_Tier) Values (25422, ‘Qwerty6’, ‘Gold’);

INSERT INTO Account(Account_no, Password, Account_Tier) Values (25423, ‘Qwerty7’, ‘Blue’);

INSERT INTO Account(Account_no, Password, Account_Tier) Values (25424, ‘Qwerty8’, ‘Gold’);

Customer:

```
CREATE TABLE Customer
(
 CID INT NOT NULL,
 CustomerType VARCHAR(15) NOT NULL ,
 PRIMARY KEY(CID),
Acnt_num INT NOT NULL,
 FOREIGN KEY (CID) REFERENCES Airbnb_User(USER_ID) on delete cascade
);
```

//INSERTING VALUES INTO THE TABLES

INSERT INTO Customer(CID, CustomerType, Acnt _num) VALUES(1111, 'Diamond', 25415);

INSERT INTO Customer(CID, CustomerType, Acnt _num) VALUES(2222, 'Platinum',25416);
INSERT INTO Customer(CID, CustomerType, Acnt _num) VALUES(3333, 'Silver',,25417);
INSERT INTO Customer(CID, CustomerType, Acnt _num) VALUES(4444, 'Diamond',25418);
INSERT INTO Customer(CID, CustomerType, Acnt _num) VALUES(5555,'Gold',25419);

Host:

```
CREATE TABLE Host (
  host_id INT NOT NULL,
  Rating INT NOT NULL,
 Acnt_num INT NOT NULL,
 Primary Key (host_id),
 FOREIGN KEY (host_id) REFERENCES Airbnb_User(USER_ID) on delete cascade
);

INSERT INTO Host(host_id, Rating, Acnt_num) Values (0147, 5, 25420);
INSERT INTO Host(host_id, Rating, Acnt_num) Values (4656, 5, 25421);
INSERT INTO Host(host_id, Rating, Acnt_num) Values (8520, 5, 25422);
INSERT INTO Host(host_id, Rating, Acnt_num) Values (4928, 5, 25423);
```

Manager

```
CREATE TABLE Manager
(
 MID INT NOT NULL,
 Salary INT NOT NULL,
 PRIMARY KEY(MID),
 FOREIGN KEY (MID) REFERENCES Airbnb_User(USER_ID)
);

INSERT INTO Manager(MID, salary) Values (8520, 65000);
INSERT INTO Manager(MID, salary) Values (4859, 69000)
```

Property

```
CREATE TABLE Property (
  property_id INT NOT NULL,
  property_type VARCHAR(15) NOT NULL,
  Property_name VARCHAR(20) NOT NULL,
  host_id INT NOT NULL,
  manager_id INT NOT NULL,
  capacity INT NOT NULL,
  price INT,
  Phone INT NOT NULL,
  street  VARCHAR(15),
  city VARCHAR(20),
  state VARCHAR(20),
  zip VARCHAR(5),
  Amenities VARCHAR(20),
  Insurance_num INT NOT NULL,
  PRIMARY KEY(property_id, property_type),
  FOREIGN KEY(host_id) REFERENCES airbnb_user(user_id) on delete cascade,
  FOREIGN KEY(manager_id) REFERENCES Manager(Mid) on delete cascade.
);


// Insert Values
INSERT INTO
Property(property_id,property_type,property_name,host_id,manager_id,capacity,price,pho
ne,street,city,zip,amenities,insurance_num)  Values(121,'hotel','Air',4928, 4859, 82, 120,
2143505577, '2370 W NW','richardson', 75211, 'Parking-food-wifi'', 21454);


INSERT INTO
Property(property_id,property_type,property_name,host_id,manager_id,capacity,price,pho
ne,street,city,zip,amenities,insurance_num)  Values(122,'hotel','Air',4656, 8520, 82, 120,
2143505577, '2370 W NW','richardson', 75211, 'Parking-food-wifi'', 21454);
```

Booking

```
CREATE TABLE Booking (
  Booking_number INT NOT NULL,
  property_id INT NOT NULL,
  host_id INT NOT NULL,
  Customer_id INT NOT NULL,
  guest_count INT NOT NULL,
  booking_price INT,
  check_in_date CHAR(10),
  check_out_date CHAR(10),
  Payment_Type VARCHAR(10) NOT NULL,
  Review_ID INT NOT NULL,
PRIMARY KEY(booking_number),
FOREIGN KEY(property_id) REFERENCES property(property_id)
on delete cascade,
FOREIGN KEY(host_id) REFERENCES host(host_id) on delete cascade,
FOREIGN KEY(Customer_id) REFERENCES customer(user_id) on delete cascade
);

INSERT INTO Booking(Booking_number, Property_id, host_id, Customer_id, guest_count,
booking_price, check_in_date, check_out_date, payment_type, Review_ID) values
(011,121,0147,1111,2,70, TO_DATE('2021-11-29','YYYY-MM-DD'),TO_DATE('2021- 12-
31','YYYY-MM-DD'), ‘cash’, 12);

INSERT INTO Booking(Booking_number, Property_id, host_id, Customer_id, guest_count,
booking_price, check_in_date, check_out_date, payment_type, Review_ID) values
(012,122,4656,2222,1,75, TO_DATE('2022-11-09','YYYY-MM-DD'),TO_DATE('2022- 11-
12','YYYY-MM-DD'), ‘cash’, 16);
```

Payment:

```
  CREATE TABLE payment (
  PID INT NOT NULL,
  booking_number INT NOT NULL,
  host_id VARCHAR(20) NOT NULL,
  CardNo INT NOT NULL
  PRIMARY KEY(PID,
  FOREIGN KEY(booking_number) REFERENCES Booking(booking_number) on delete
cascade,
  FOREIGN KEY(host_id) REFERENCES host(host_id) on delete cascade
);
```

//INSERTING VALUES INTO THE TABLES
INSERT INTO Payment(PID, booking_number,host_id,CardNo) VALUES(16592,011,0147, 1959483629);
INSERT INTO Payment(PID, booking_number,host_id,CardNo) VALUES(59682,012, 4656, 5789123963);

- **CARD_INFO:**

```
CREATE TABLE Card_info
(
CardNo int NOT NULL,
 CVV int NOT NULL,
Card_holder_name VARCHAR(30) NOT NULL,
 Expiry DATE NOT NULL,
Payment_type varchar(50) NOT NULL,
PRIMARY KEY(CardNo),
FOREIGN KEY (CardNo) REFERENCES Payment(CardNo) on delete cascade
);
```

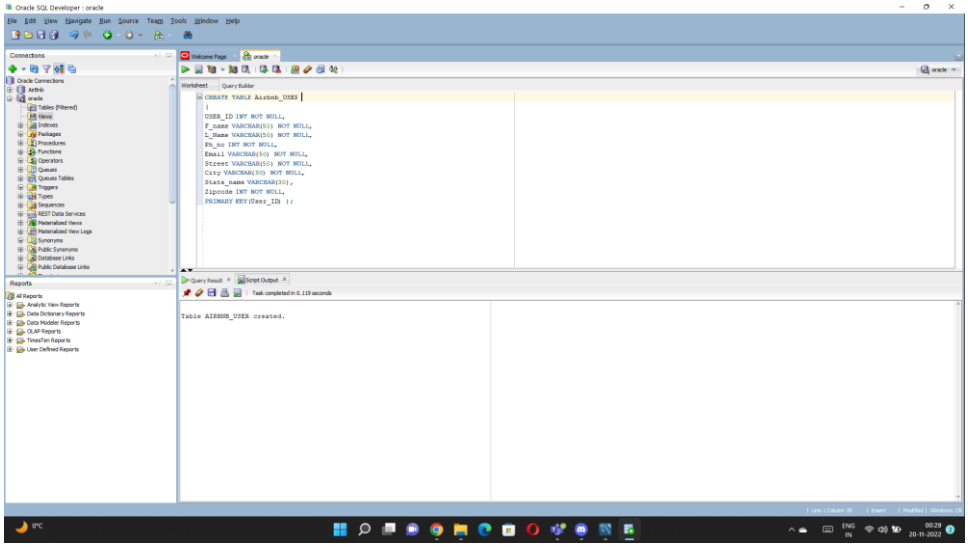
//INSERTING VALUES INTO THE TABLES

```
INSERT INTO Card_info(CardNo, CVV, Card_holder_name, Expiry,Payment_type)
VALUES(1959483629, 456, 'Julie', TO_DATE('2022-12-11','YYYY-MMDD'),
'Personalpayment');

INSERT INTO Card_info(CardNo, CVV, Card_holder_name, Expiry, Payment_type)
VALUES(5789123963, 951, 'Rachel', TO_DATE('2023-5-19','YYYY-MMDD'),
'Personalpayment');

INSERT INTO Card_info(CardNo, CVV, Card_holder_name, Expiry, Payment_type)
VALUES(9876548521, 258, 'Jaden', TO_DATE('2023-12-11','YYYY-MMDD'),
'Businesspayment');
```

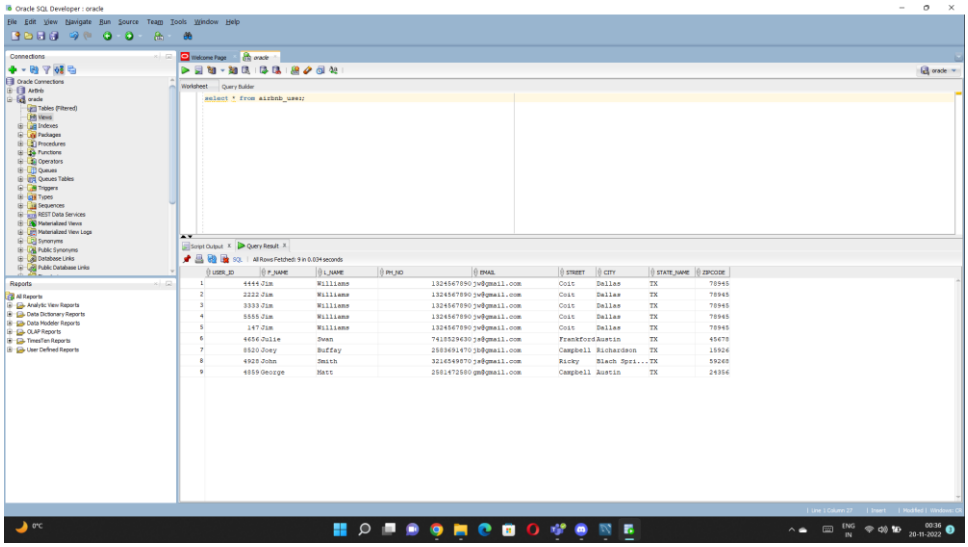
Review:

```
CREATE TABLE review (
  review_id INT NOT NULL,
  customer_id VARCHAR(20),
  property_id INT NOT NULL,
  property_type VARCHAR(15) NOT NULL,
  ratings INT ,
  comments VARCHAR(20),
PRIMARY KEY (review_id),
FOREIGN KEY (customer_id) REFERENCES customer (CID),
FOREIGN KEY (property_id) REFERENCES property (property_id)
on delete cascade
);
```

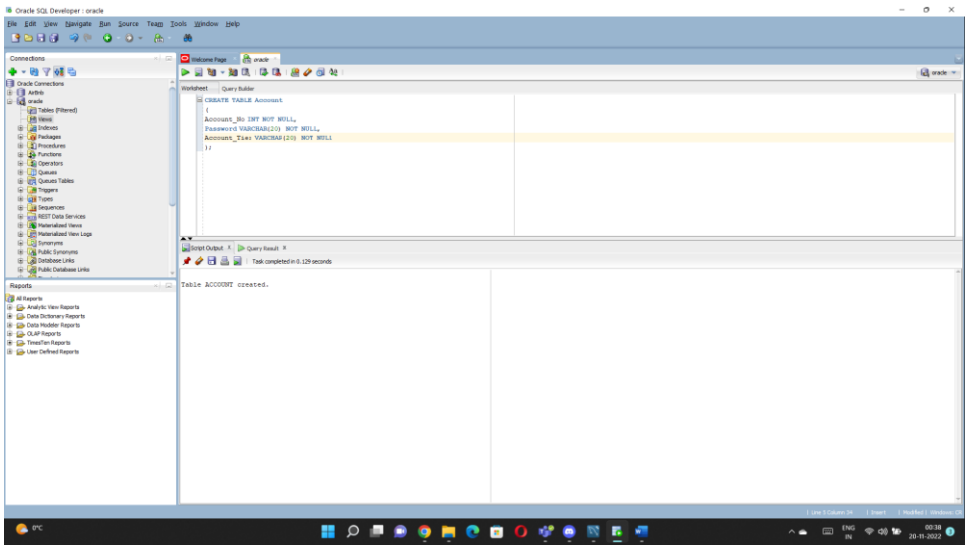
Snapshots:
Create table Airbnb_User:



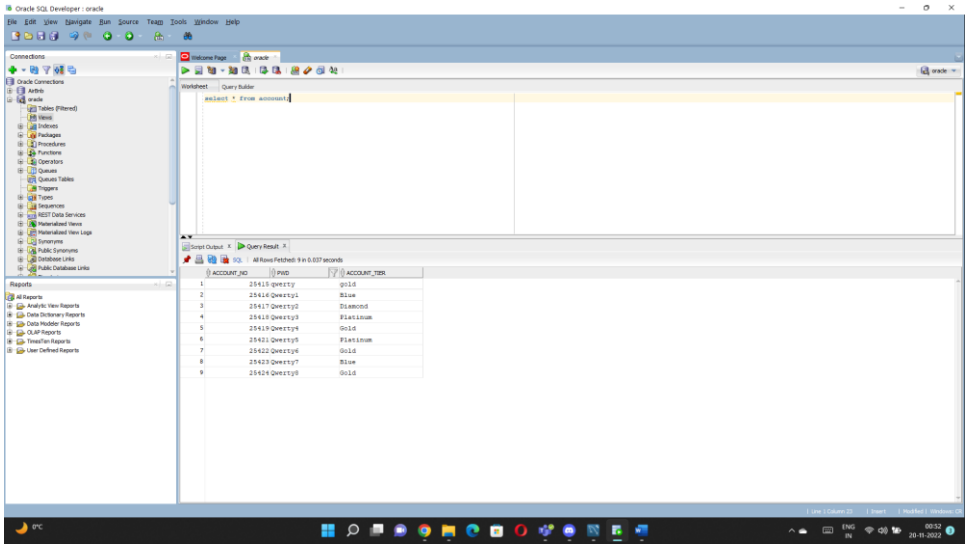
Insert Values:



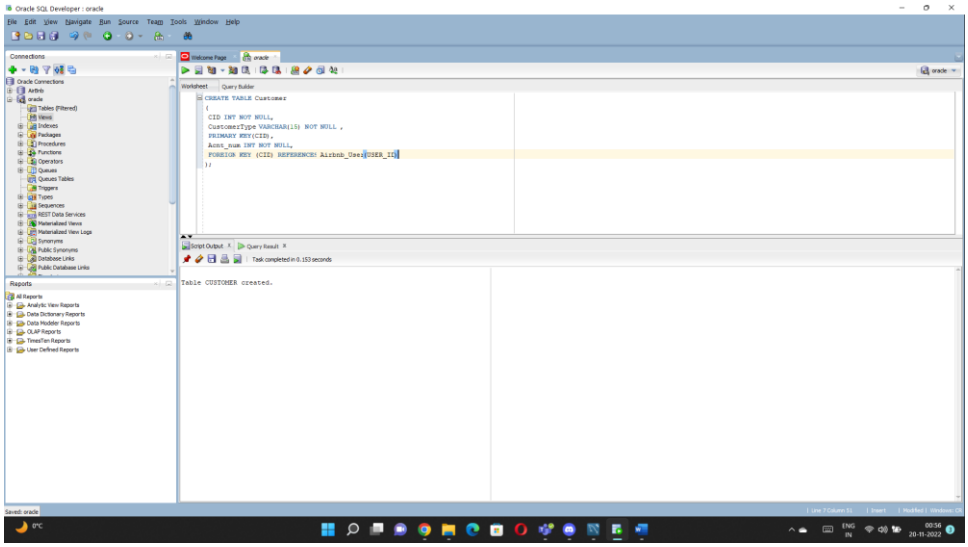
Create Account:



Insert Account Values:



Create Table Customer:



Insert Customer Values:

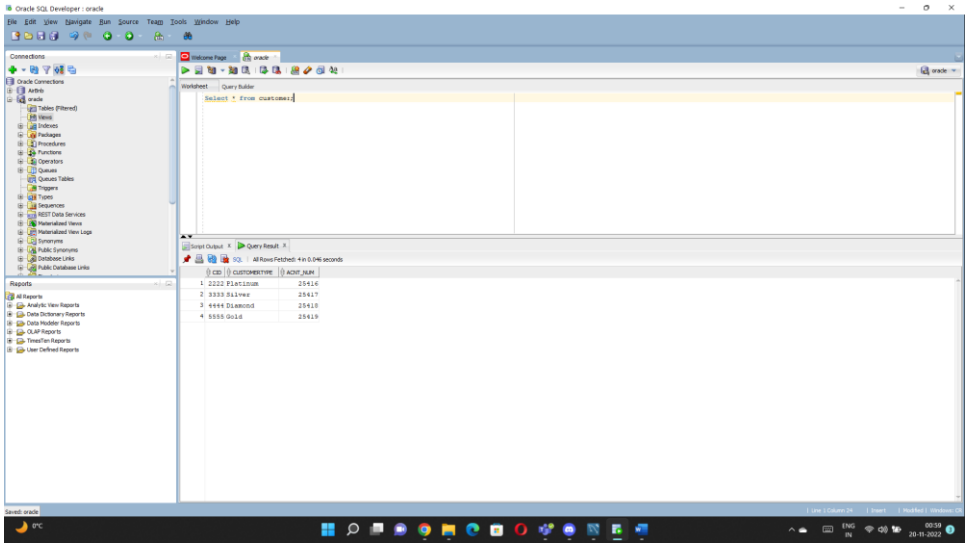
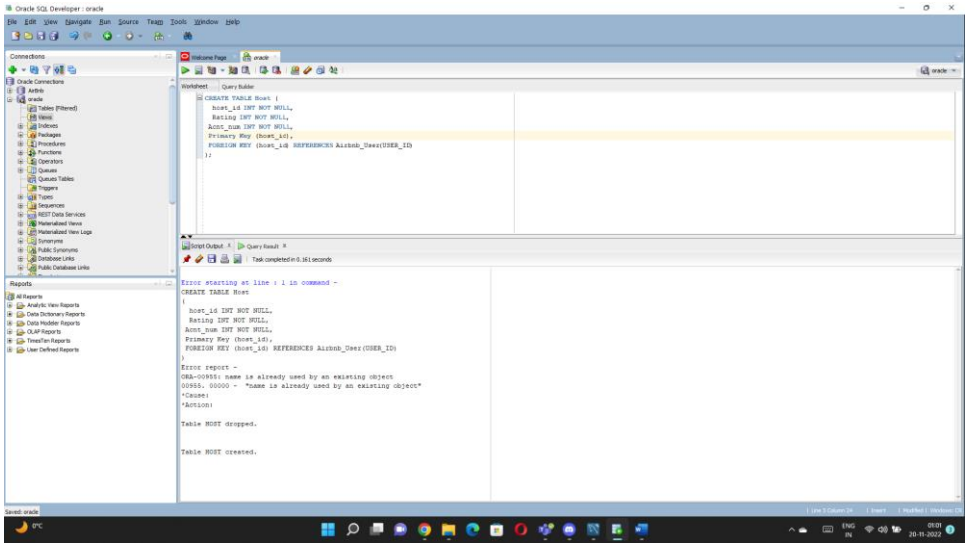
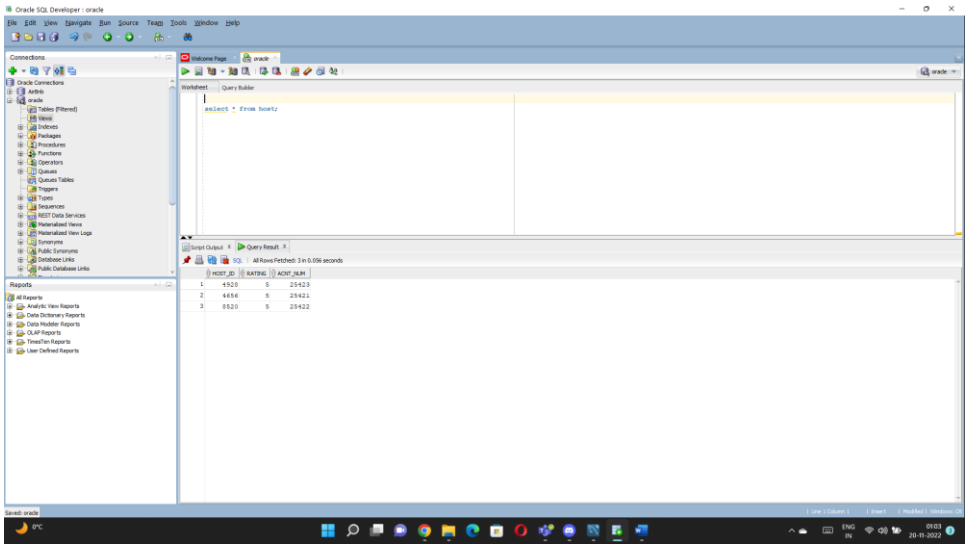


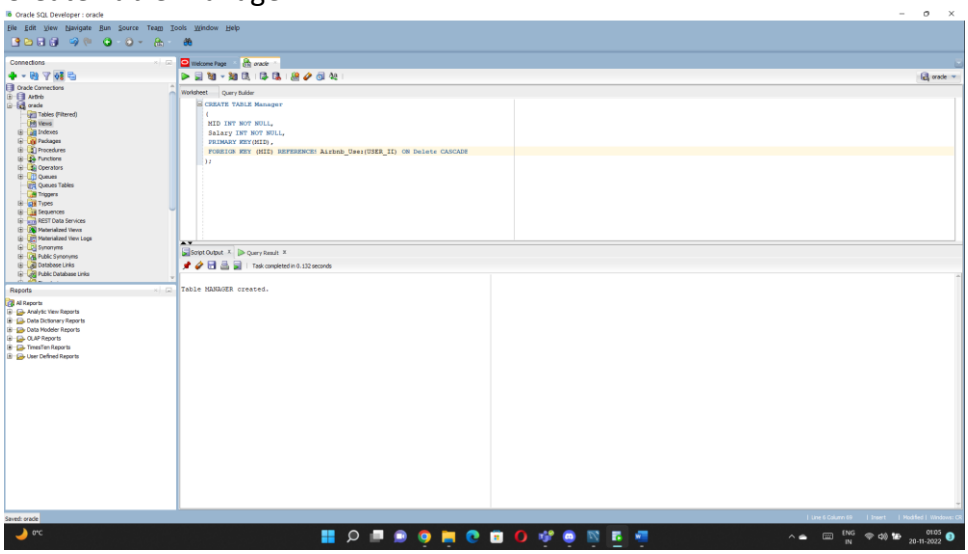
Table Host:



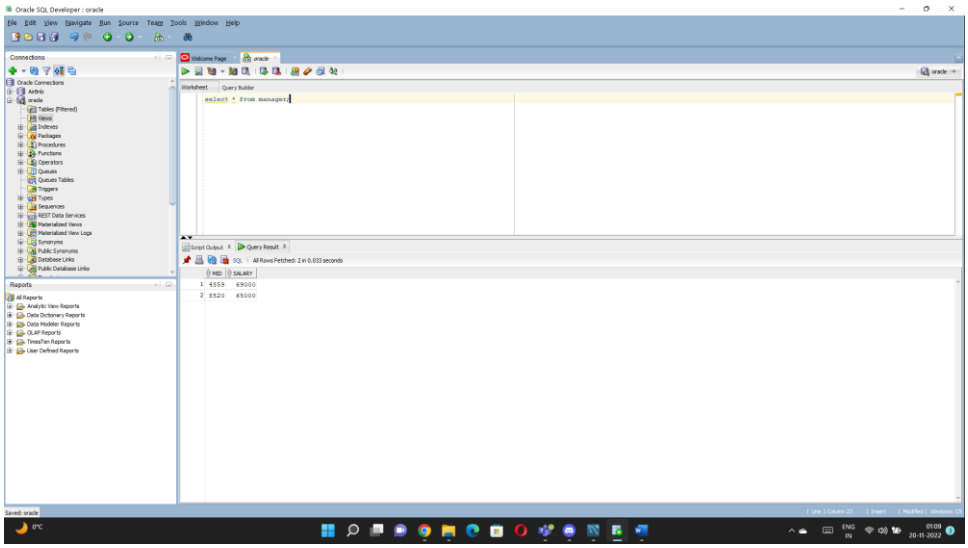
View Host:



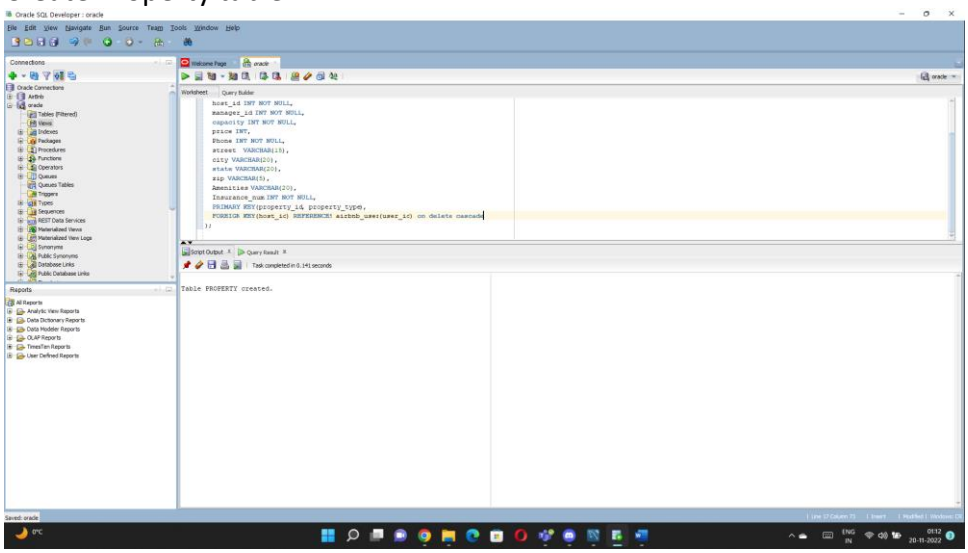
Create Table Manager:



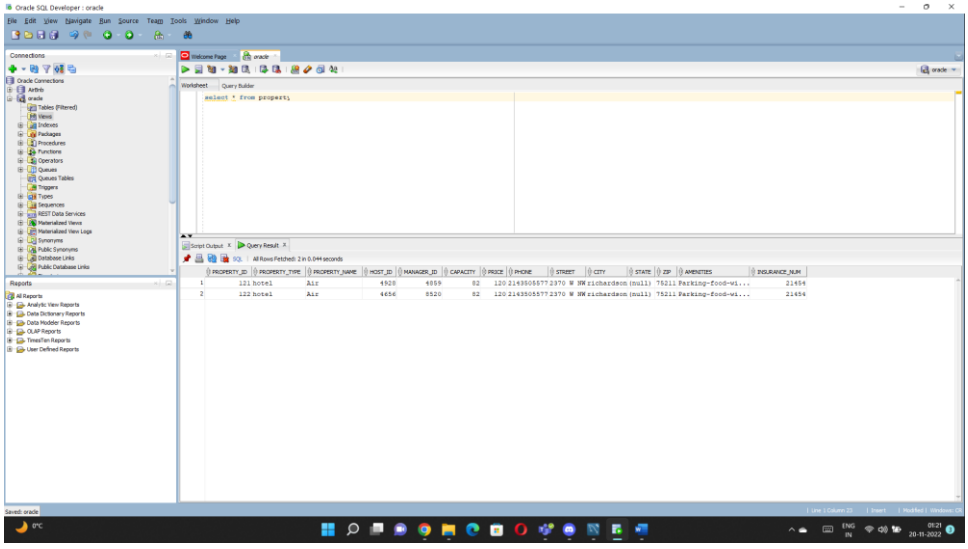
Values In Manager



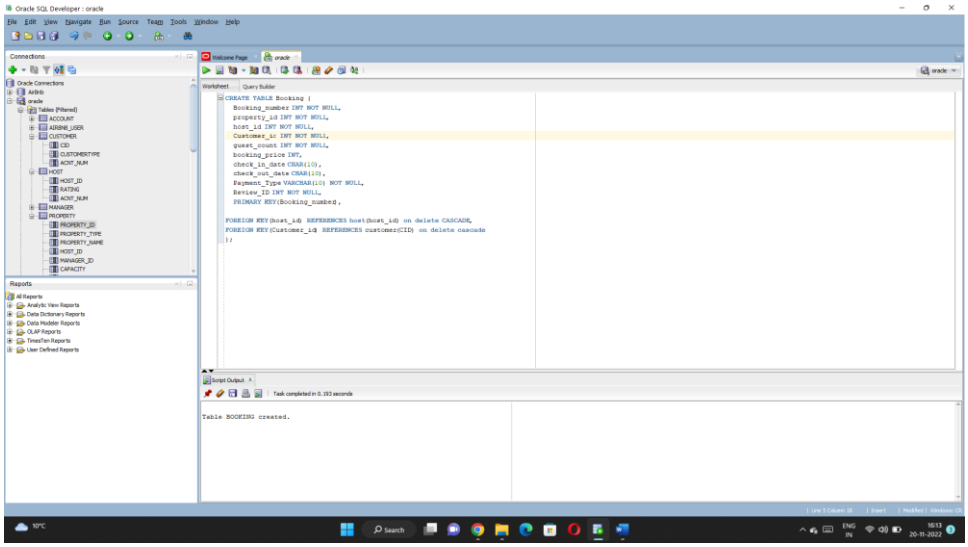
Create Property table:



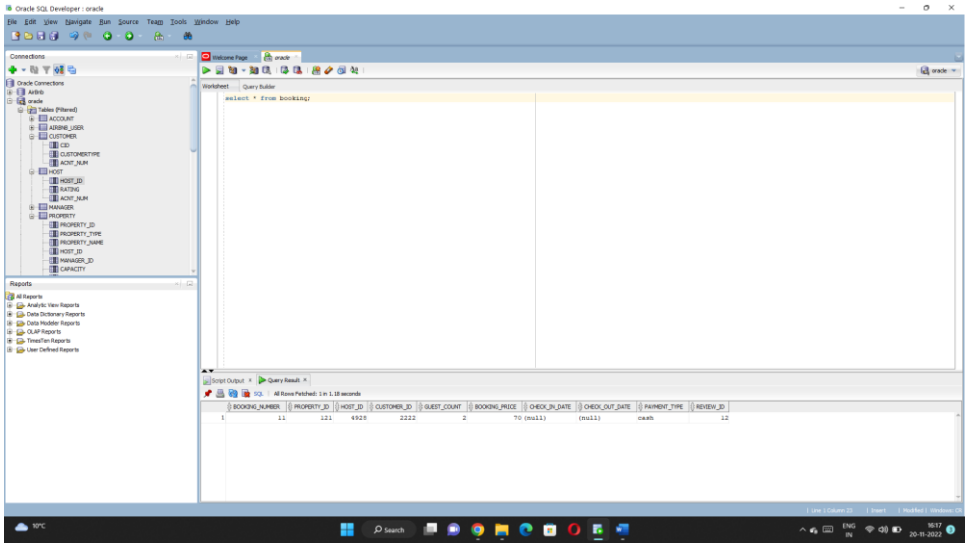
Insert Values:



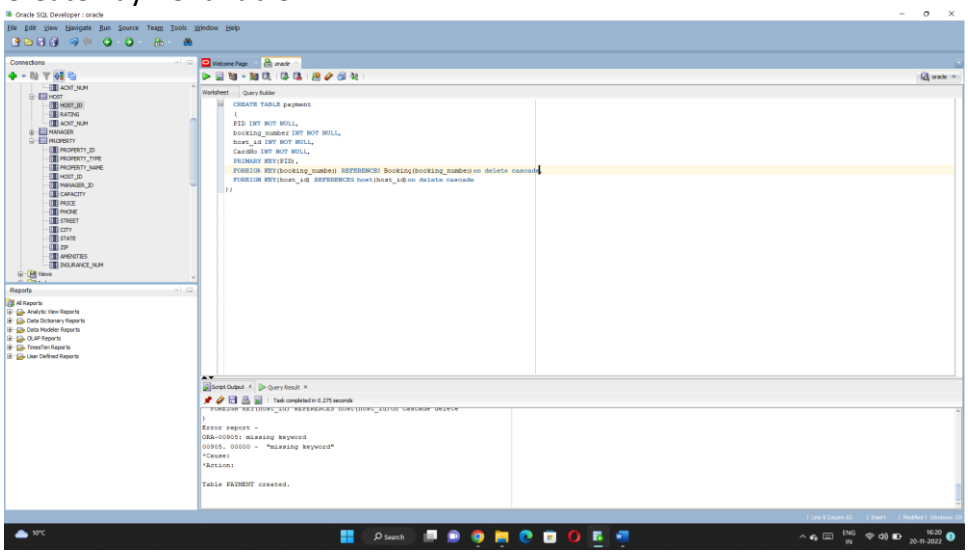
Create Table Booking:



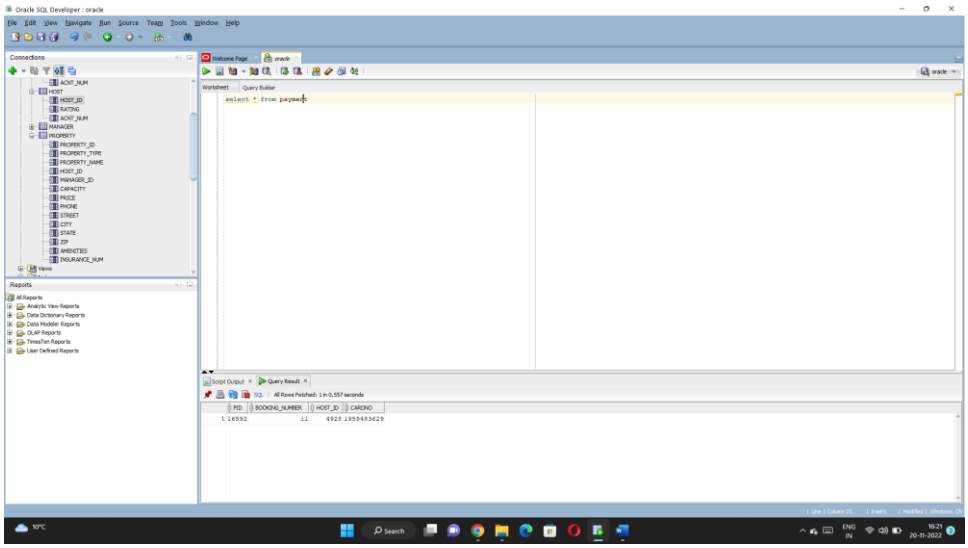
Insert Values:



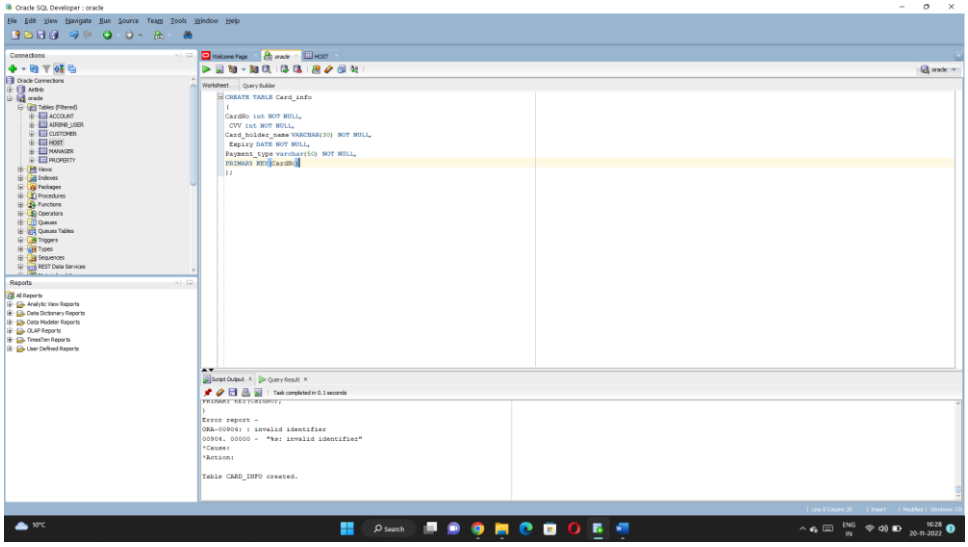
Create Payment Table:



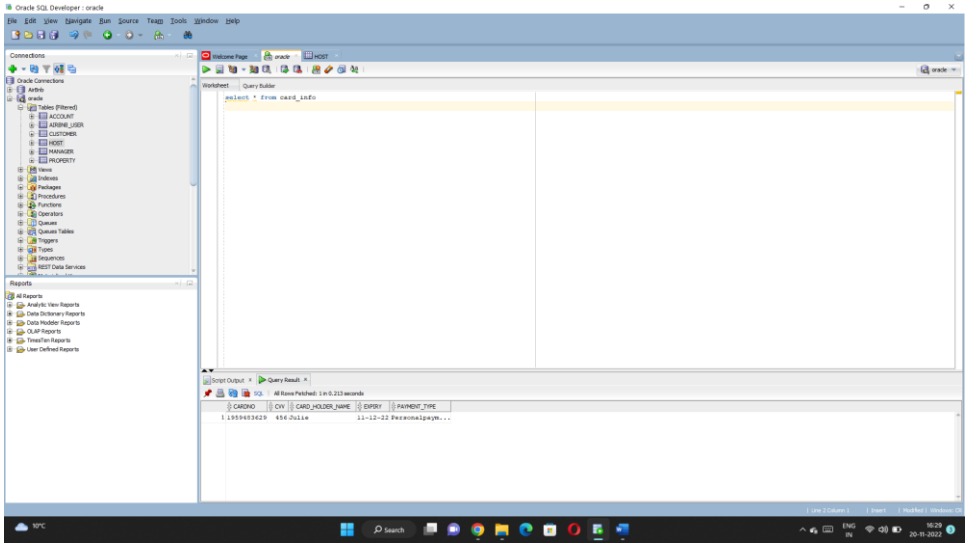
Insert Row:



Create Card Info Details:



Enter card details:



PL/SQL STATEMENTS

```
--Data for stored procedures
INSERT INTO CREDENTIAL VALUES (1, 'password');
INSERT INTO HOST VALUES (1, 'Ricky', 'Martin', 'rmartin@gmail.com', '6821231234');
INSERT INTO Property Values (121,'hotel','Air',4928, 4859, 82, 120, 2143505577, '2370 W
NW','richardson', 75211, 'Parking-food-wifi'', 21454);

INSERT INTO Property Values (122,'hotel','Air',4656, 8520, 82, 120, 2143505577, '2370 W
NW','richardson', 75211, 'Parking-food-wifi'', 21454);

create or replace PROCEDURE is_amenities_property AS

thisProperty property%ROWTYPE;

CURSOR all_amenities_available IS
SELECT * FROM Property WHERE amenities LIKE '1';

BEGIN
dbms_output.put_line('List of amenities available: ');
OPEN all_amenities_available;
LOOP
  FETCH all_ amenities_available INTO thisProperty;
  EXIT WHEN (all_ amenities_available %NOTFOUND);
  dbms_output.put_line(thisProperty.property_name);
END LOOP;
CLOSE all_amenities_available;
END;


SET SERVEROUT ON
EXECUTE is_amenities_available;

create or replace PROCEDURE property_has_wifi AS

thisProperty property%ROWTYPE;

CURSOR all_wifi_property IS
SELECT * FROM property WHERE wifi LIKE '1';

BEGIN
dbms_output.put_line('List of properties with wifi: ');
OPEN all_wifi_properties;
LOOP
  FETCH all_wifi_properties INTO thisProperty;
  EXIT WHEN (all_wifi_properties%NOTFOUND);
  dbms_output.put_line(thisProperty.property_name);

END LOOP;
CLOSE all_wifi_properties;
```

```
END;

SET SERVEROUT ON
EXECUTE property_has_wifi;

--data for trigger pwd_change_notification
insert into credentials values (2, 'pwd');
insert into customer values (2, 'Victor', 'Wooten', 'vwoot@hotmail.com', NULL);

create or replace trigger pwd_change_notification
 AFTER UPDATE OF PASSWD ON CREDENTIAL
 BEGIN
   dbms_output.put_line('password has been changed.');
 END;




-- test the trigger by updating password value
SET SERVEROUT ON
UPDATE CREDENTIAL SET PASSWD = 'pass-word' where user_id = 2;

--data for trigger check_customer_payment_details
insert into credential values(3, 'alliance');
insert into credential values(4, 'horde');
insert into host values(3, 'Jim', 'Williams', 'anduin_wrynn@battle.net',NULL);
insert into property values(3,'housing',3,75,NULL,'18002937684','Trade
District',NULL,'Stormwind city',NULL,'Azeroth',NULL);
insert into housing values(3,'0','0','0','1','1','1','0','0','5','5');
insert into customer values(4,'Garrosh','Hellscream','gscream@yahoo.com',NULL,NULL);

-- trigger to check payment details of customer before they can create a booking
CREATE OR REPLACE TRIGGER check_customer_payment_details
BEFORE INSERT ON BOOKING FOR EACH ROW
DECLARE
payment customer.payment_details%TYPE;
customer_id booking.customer_id%TYPE;
BEGIN
   SELECT payment_details INTO payment from customer c
   WHERE c.user_id = :new.customer_id;
   IF payment IS NULL THEN
      RAISE_APPLICATION_ERROR(-20000, 'Customer does not have payment details');
   END IF;
END;

--test trigger should raise error since the customer referenced has no payment details
insert into booking values(1,4,3,3,1,NULL,NULL);
```