Project Report

AirBNB Database System

Course: CS 6360(Database Design)
Instructor: Dr. Nurcan Yuruk
Group Number - 28

Pravin Ghorpade
Department of Computer Science
The University of Texas at Dallas
Dallas, USA
psg190003
psg190003@utdallas.edu

Sanjay Jayraj
Department of Computer Science
The University of Texas Dallas
Dallas, USA
sxj190087
sxj190087@utdallas.edu

Akash R B
Department of Computer Science
The University of Texas Dallas
Dallas, USA
axr190117
axr190117@utdallas.edu

AIRBNB-

Airbnb, operates an online marketplace for lodging, primarily homestays for vacation rentals, and tourism activities. It is based in San Francisco, California. The platform is accessible via website and mobile app. Airbnb does not own any of the listed properties; instead, it profits by receiving commission from each booking. The company was founded in 2008 by Brian Chesky, Nathan Blecharczyk and Joe Gebbia. Airbnb is a shortened version of its original name, AirBedandBreakfast.com

How Airbnb Work?

- AIRBNB has a number of users which are either customers or hosts
- The customers select the location of their choice, the dates of their choice choice and book one of the houses that are listed.
- Upon booking, the customer makes the payment out of which part of the payment goes to the AIRBNB enterprise and the other part goes to the Host whose house is been booked by the Customer.
- Customers can book a house, cancel a booking and if a booking is cancelled then the payment is refunded.
- Hosts can put up any number of houses on rent and can take them of anytime.

Business Rules:

For User

- User can make account on website.
- The User can then decide whether he wants to become a Customer to rent the houses or he wants to become the Host to put up his house on the rent.
- One User can either be a Customer, Host or both Customer and a Host.
- The Customer can make any number of bookings under his account.
- The Host can put up any number of houses on rent under his account.
- The user (both Customer and Host) can give feedbacks to the AIRBNB Enterprise.
- The Customer can rate the Host upon his experience and the Host can rate the Customer upon his experience.
- The Customer can review the house he had rented.
- The person can deactivate or activate his account any number of times

For an Enterprise

- The AirBNB receive feedbacks from their customers and hosts. The feedback can be regarding the services, the website or the enterprise.
- The AirBNB can view all the statistics such as the total number of customers, total number of hosts, total number of houses, total number of houses in each locations, customer who has the highest number of bookings, host who has the maximum number of houses up for rent, host who has received good ratings, the customers who have received bad ratings and so on.
- AIRBNB maintains a backup of all the data that it has in its database, be it regarding the person information, the booking information or the payment information.
- Different employees of AIRBNB will have different levels of access to the database.

Other Business Rules

- If a person is a customer and wants to delete his account, then the AIRBNB should check whether the customer has any forthcoming bookings under his name. If he does, then he would not be able to delete his account.
- If a person is a host and wants to deactivate his account, then the AIRBNB should automatically delete the house listings so that no one can book the houses whose host does not exist.
- If a booking is made, then the payment table should automatically be populated and the earnings of the Enterprise and the Host should be segregated.
- If the booking is cancelled by the customer the payment made by the customer should be refunded immediately.
- If a house has been booked by the customer for a range of days, then some other customer should not be able to book the same house within that range.

Entities:

- person: user of the system. It can be a customer or a host.
- customer: person who uses the system for booking the property
- host: person who uses the system to list his property.
- property: host can list his property in the listings. It can be a hotel/home/apartment.
- property_facilities: facilities available in the property.
- location: location of the property
- feedback : feedback given by the customer
- ratings: ratings of each property based on the feedback given by the customers
- booking: details of the booking made by the customer.
- payment: customer can pay in the system and it will be received by the host
- cancellation: customer can cancel his booking
- refund: customer will get his refund if he cancels the booking
- airbnb_earning: some part of the payment will be the earning of Airbnb
- host_earning: remaining part of the payment will be the earning of host.

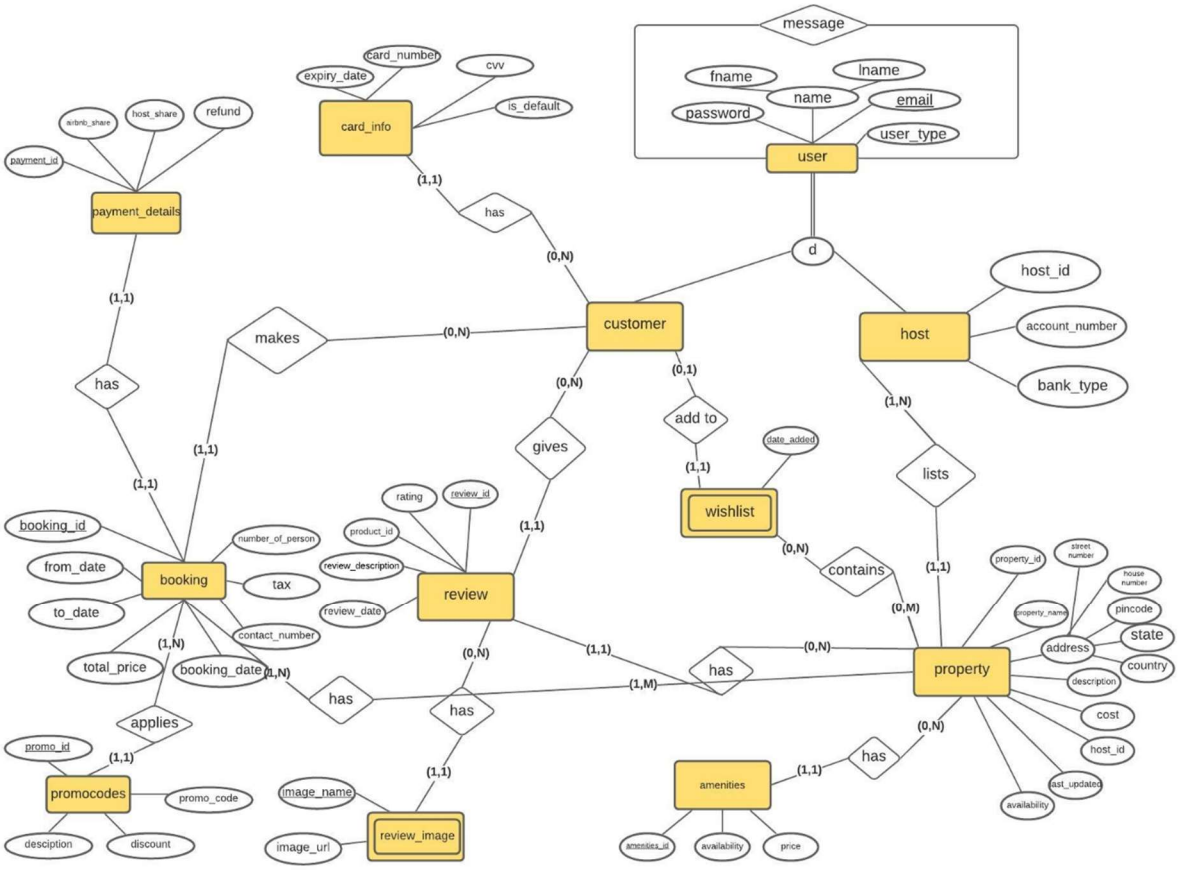
Functional Requirements:

- A person can register– A user can register as a customer or a host
- A customer can make a booking- a customer can make multiple bookings.
- A host can list property- this contains the details of a property.
- A customer can give review- a customer can write reviews about a property.
- A customer can add property to a Whishlist- customer can save the property which they like but not yet ready to book
- A customer can add property to a shopping cart
- Property can be of multiple category- it defines the category of a property for example: hotel, resorts, private house, room, lodge.
- A customer can have multiple Card info- customer saved card in the account
- A customer can add images to its review- each review can have images associated with it
- A host can add images and description to its property - each property can have multiple images associated with it, so taken in separate table.

Relationships:

- Person-contact_details: each person can have multiple contact details saved, while each contact detail will have only 1 person linked. Thus, cardinality is 1: N
- customer-card_info: each customer can have many cards, while each card is associated with 1 buyer. Thus, cardinality is 1: N
- Customer-booking: Customer can book a property. A customer can book many properties, while each booking is linked with only 1 customer. Thus, cardinality is 1: N
- Booking-property: each booking can contain many properties, and each property can come in multiple bookings. Thus, cardinality is M: N
- host-property: host lists his property. Each host can list many properties, while each property has only 1 host. Thus, cardinality is 1: N
- Customer-reviews: customer can write a review. Each customer can write multiple reviews, while each review is written by only one customer. Thus, cardinality is 1: N
- Review-products: each review is listed under 1 property while each property can have many reviews. Thus, cardinality is 1: N
- Customer-whishlist: each customer can have only 1 whishlist, and each whishlist has 1 customer. Thus, cardinality is 1:1.
- Customer- cart: each customer has only 1cart while each cart is associated with 1 customer. Thus, cardinality is 1:1.
- Whishlist-property: each whishlist contains many properties, and each properties can be in many like lists. Thus, cardinality is N:M
- Shopping_cart-property: each cart contains many properties, and each property can be in many carts. Thus, cardinality is N:M
- Property-category: each property has only 1 category while each category has many properties. Thus, cardinality is 1: N
- Review-review_images: each review has many review images while each review image is linked with 1 review. Thus, cardinality is 1: N
- Property-property_images: each property has many property images while each property image is linked with 1 property. Thus, cardinality is 1: N
- Order-card_info: each order has a card linked with it, while each card can be used for many orders. Thus, cardinality is 1: N
- Order-contact details: each order has 1 contact detail (address, phone), while each contact detail is linked with multiple orders. Thus, cardinality is 1: N

Entity-Relationship Diagram:



E-R diagram for Airbnb

Relational Schema:

To map ER diagram into a relational schema, we considered the following mapping rules.

For each 1: 1 binary relationship, in the total participation entity add the primary key of the other entity as the foreign key.

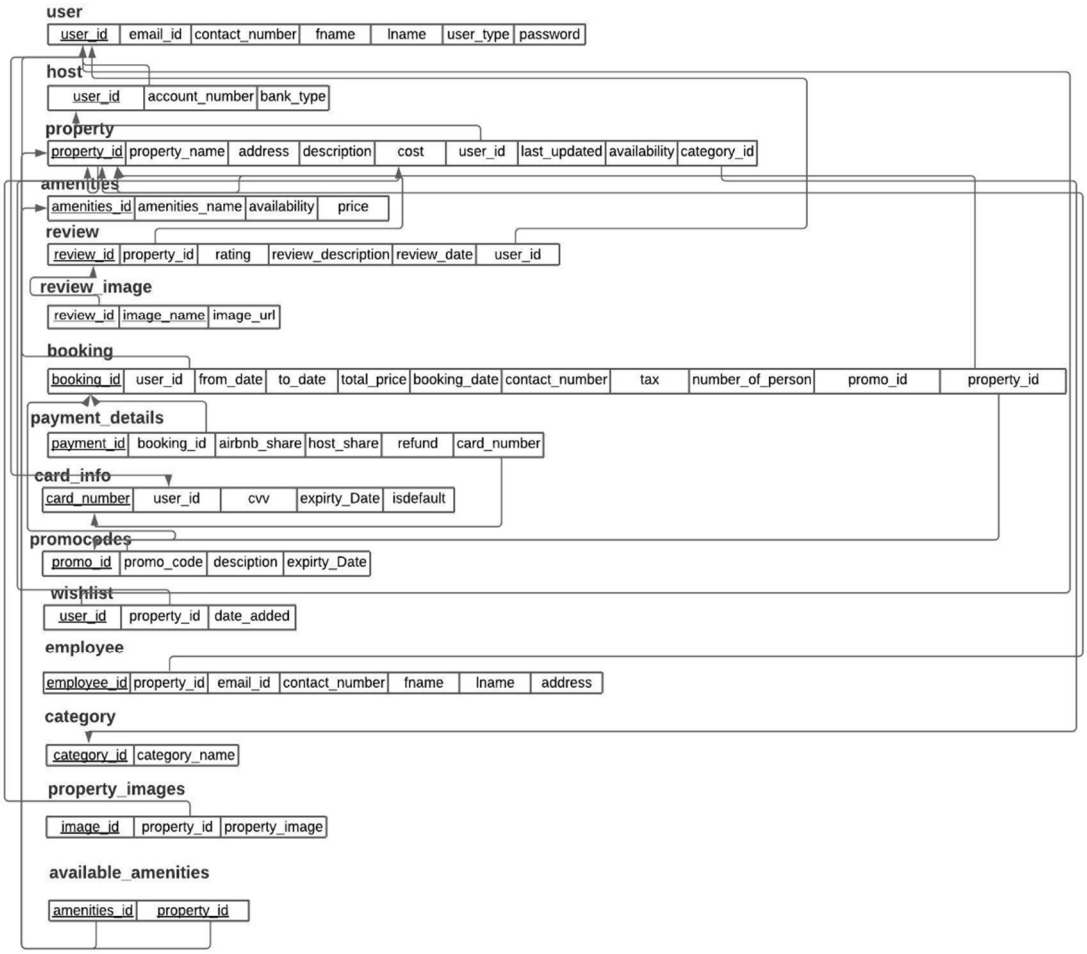
For 1: N binary relationship, add to the entity on the N side the primary key of the other entity as the foreign key.

For M: N binary relationship, make a new entity with foreign key as the primary key of the two participating entities. Their combination forms the new primary key.

- In customer table we have person_id as foreign key.
- In host table we have person_id as foreign key.
- In property table we have host_id, and category_id as foreign keys.
- In booking table we have customer_id as foreign key.
- In card_info we have customer_id as foreign key.
- In reviews we have customer_id, property_id as foreign keys.
- In whishlist we have customer_id as foreign key.
- In shopping cart we have customer_id as foreign key.
- In contact_details we have customer_id as foreign key.
- In review_images we have review_id as foreign key.
- In property_images we have property_id as foreign key.
- We make a new table name property_whishlist which as property_id and whishlist_id as foreign key.
- We make a new table name property_shopping_cart which as property_id and customer_id as foreign key.
- We make a new table name property_booking which as property_id and booking_id as foreign key.

After converting ER model into relational tables we can say that the resultant tables do not violate any conditions of the 3NF. Thus the tables are in 3NF normalized

Relational Schema



Relational Schema for Airbnb

Significant Procedures:

- Register customer: invoked by customer responsible for
  - registering user given email, fname, lname and password.
- Register host: invoked by customer responsible for
  - registering user given email, fname, lname and password.
- Book property: Given a customer id, book property is responsible for the following –
  - To iterate over the shopping cart of a particular customer and remove each item from customer's cart.
  - While removing each product, sum up the price of each property to the total price for the order.
  - Fetch the default card details set by the customer from the list of card details for that buyer from the card_info table.
  - To add an entry in the booking table, containing details of the invoice (total price, details of the property, tax, number of days, card_details used for the order, contact details used for the booking)
  - To invokes a trigger responsible for updating the availability of each property being rented in that order.
- Give review:
  - A customer can add a review by providing property_id, customer_id, review, rating, and image_url (if any). It adds a review in the review table and image of it in the image table. After execution It invokes two triggers.
    - update_property_rating
    - update_hosts_rating.
- Add_contact_details: adds details about address and the user's number.
- Add_card_info: add cards information like card number, expiry_date etc. A user can add mupltiple card details and can set a single to card_info to be used by default.
- Add_property: adds a property given on a rent by hosts. It also asks for the image URL if any, A host can upload multiple images for a property.
- Add_to_whishlist: adds a property to the wishlist of a customer, given the buyer id.
- Populate_property_categories: adds all the available categories of property in it
- Cancel_booking: cancel the given booking

Important Triggers:

- Update available property:
  Trigger is invoked whenever a customer books a property. “Update available property” is responsible for updating the value of available units for each property in the property table that is being booked by a customer.
  The procedures iterate over all the entries of book_property table for a specific booking and iteratively updates each property in the property table.
  If the quantity reaches 0, the property is marked as sold out, setting its available_unit value as 0.

- Update_property_rating:
  It is responsible for updating the rating of a property every time a customer gives a review by averaging the earlier rating with this host rating.
  It also updates the count of rating given for that particular property.

Project Implementation Details :

1. user table

```
SELECT * FROM `user`
```

Profiling [Edit inlin

Show all | Number of rows: 25 Filter rows: Search this table Sort by key: None

+ Options

	user_id	email_id	contact_number	fname	lname	user_type	password
Edit Copy Delete	2	pravin.ghorpade@gmail.com	4694271986	Pravin	Ghorpade	host	123456789
Edit Copy Delete	4	sanjay.jayraj@gmail.com	4545454545	Sanjay	Jayraj	host	123456789
Edit Copy Delete	6	Akash R B	8798451243	Akash	R B	host	987654321
Edit Copy Delete	9	RobertTCruz@dayrep.com	4016613641	Robert	T. Cruz	customer	ud7eLae0Ahch
Edit Copy Delete	10	KatharinaFWatson@armyspy.com	3108151798	Katharina	Watson	customer	ahRoh4ax
Edit Copy Delete	13	JohnALeasure@armyspy.com	9378406216	John	Leasure	customer	Kah4iewoo

2. host table

```
SELECT * FROM `host`
```

Show all | Number of rows: 25 Filter rows: Search this table Sort by key: None

Options

	host_id	user_id	account_number	bank_type
Edit Copy Delete	akash1	6	444444444	Wells Fargo
Edit Copy Delete	pravin1	2	142354645	chase
Edit Copy Delete	sanjay1	4	55555555555	chase

3.amenities

```
SELECT * FROM `amenities`
```

Show all | Number of rows: 25 Filter rows: Search this table Sort by key: None

+ Options

	amenities_id	name	availability	price
Edit Copy Delete	amn_1	kitchen	1	30
Edit Copy Delete	amn_2	heating	0	25
Edit Copy Delete	amn_3	air conditioning	1	10
Edit Copy Delete	amn_4	washer	1	5
Edit Copy Delete	amn_5	breakfast	1	5
Edit Copy Delete	amn_6	TV	1	10

4. available amenities

```
SELECT * FROM `available_amenities`
```

Show all | Number of rows: 25 Filter rows: Search this table Sort by key: None

+ Options

amenities_id	property_id
amn_2	123460
amn_1	123459
amn_3	123459
amn_4	123459
amn_6	123459
amn_5	123459

5.booking

```
SELECT * FROM `booking`
```

Profiling [Edit inline] [Edit] [Explain SQL] [Create PHP code] [R

Show all | Number of rows: 25 Filter rows: Search this table Sort by key: None

+ Options

	booking_id	user_id	from_date	to_date	total_price	booking_date	contact_number	tax	number_of_person	promo_id	booking_status
Edit Copy Delete	457894	13	2021-05-11	2021-05-13	2500	2021-05-03	123456789	31	2	promo_1	Booked
Edit Copy Delete	785425	10	2021-06-16	2021-06-17	1000	2021-04-07	75842459	20	1	promo_2	

6.card_info

```
SELECT * FROM `card_info`
```

Show all | Number of rows: 25 Filter rows: Search this table Sort by key: None

+ Options

	card_number	user_id	cvv	expiry_date	isdefault
Edit Copy Delete	4242424242424242	9	245	2024-05-13	1
Edit Copy Delete	5353535353535353	13	11	2025-12-09	0

Check all With selected: Edit Copy Delete Export

7.category

```
SELECT * FROM `category`
```

Show all | Number of rows: 25 Filter rows: Search this table Sort by key: None

+ Options

	category_id	category_name
Edit Copy Delete	1	hotels and resorts
Edit Copy Delete	2	independent house
Edit Copy Delete	3	shared room
Edit Copy Delete	4	guesthouse

8.employee

```
SELECT * FROM `employee`
```

Profili

Show all | Number of rows: 25 Filter rows: Search this table Sort by key: None

+ Options

	employee_id	property_id	email_id	contact_number	fname	lname	address
Edit Copy Delete	556546	123460	sam.sum@airbnb.com	4665445612	Sam	Summers	Dallas
Edit Copy Delete	556547	123459	mark.twin@airbnb.com	4245153554	Mark	Twin	South Dallas

9.host

```
SELECT * FROM `host`
```

Show all | Number of rows: 25 Filter rows: Search this table Sort by key: None

+ Options

	host_id	user_id	account_number	bank_type
Edit Copy Delete	akash1	6	444444444	Wells Fargo
Edit Copy Delete	pravin1	2	142354645	chase
Edit Copy Delete	sanjay1	4	55555555555	chase

10.payment_details

```
SELECT * FROM `payment_details`
```

Show all | Number of rows: 25 Filter rows: Search this table Sort by key: None

+ Options

	payment_id	booking_id	airbnb_share	host_share	refund	card_number
Edit Copy Delete	147852369	457894	30	2510	0	42424242424242
Edit Copy Delete	456146	457894	201	121	0	353542246465
Edit Copy Delete	963258741	785425	10	990	0	5353535353535353

11.promo codes

```
SELECT * FROM `promocodes`
```

Show all | Number of rows: 25 Filter rows: Search this table Sort by key: None

Options

	promo_id	promo_code	description	expiry_date
Edit Copy Delete	promo_1	SAVE30	Save 30% on your booking	2021-05-12
Edit Copy Delete	promo_2	SUMMER50	Save 50% on your summer holiday	2021-05-19

12.property_image

```
SELECT * FROM `property_image`
```

Show all | Number of rows: 25 Filter rows: Search this table Sort by key: None

+ Options

property_id	property_image
123460	City View Image 1
123460	City view Image 2

13.review

```
SELECT * FROM `review`
```

Show all | Number of rows: 25 Filter rows: Search this table Sort by key: None

+ Options

	review_id	property_id	rating	review_description	review_date
Edit Copy Delete	11	123460	3	Awesome Experience	2021-05-09
Edit Copy Delete	review_1	123460	4	It was a nice experience	2021-05-04
Edit Copy Delete	review_2	123459	2	Worst experience	2021-05-03

14.review_image

```
SELECT * FROM `review_image`
```

Show all | Number of rows: 25 Filter rows: Search this table Sort by key: None

Options

review_id	image_name	image_url
review_1	living room	living room image url
review_2	view	view image url

16. wishlist

```
SELECT * FROM `wishlist`
```

Profilin

Show all | Number of rows: 25 Filter rows: Search this table Sort by key: None

+ Options

user_id	property_id	date_added
13	123460	2021-05-04
13	123459	2021-05-01

SQL code for AirBnb DB system

Create table and add constraints

```
--
-- Table structure for table `amenities`
--

CREATE TABLE `amenities` (
  `amenities_id` varchar(20) NOT NULL,
  `name` varchar(32) NOT NULL,
  `availability` tinyint(1) NOT NULL,
  `price` int(11) NOT NULL
) ENGINE=InnoDB DEFAULT CHARSET=utf8;

--
-- Dumping data for table `amenities`
--

INSERT INTO `amenities` (`amenities_id`, `name`, `availability`, `price`) VALUES
('amn_1', 'kitchen', 1, 30),
('amn_2', 'heating', 0, 25),
('amn_3', 'air conditioning', 1, 10),
('amn_4', 'washer', 1, 5),
('amn_5', 'breakfast', 1, 5),
('amn_6', 'TV', 1, 10);

-- --------------------------------------------------------

--
-- Table structure for table `available_amenities`
--

CREATE TABLE `available_amenities` (
  `amenities_id` varchar(20) NOT NULL,
  `property_id` int(11) NOT NULL
) ENGINE=InnoDB DEFAULT CHARSET=utf8;

--
-- Dumping data for table `available_amenities`
--

INSERT INTO `available_amenities` (`amenities_id`, `property_id`) VALUES
('amn_2', 123460),
('amn_1', 123459),
```

```
('amn_3', 123459),
('amn_4', 123459),
('amn_6', 123459),
('amn_5', 123459);

-- --------------------------------------------------------

--
-- Table structure for table `booking`
--

CREATE TABLE `booking` (
  `booking_id` varchar(10) NOT NULL,
  `user_id` int(11) NOT NULL,
  `from_date` date NOT NULL,
  `to_date` date NOT NULL,
  `total_price` decimal(10,0) NOT NULL,
  `booking_date` date NOT NULL,
  `contact_number` varchar(15) NOT NULL,
  `tax` decimal(10,0) NOT NULL,
  `number_of_person` int(3) NOT NULL,
  `promo_id` varchar(10) DEFAULT NULL,
  `booking_status` varchar(10) NOT NULL
) ENGINE=InnoDB DEFAULT CHARSET=utf8;

--
-- Dumping data for table `booking`
--

INSERT INTO `booking` (`booking_id`, `user_id`, `from_date`, `to_date`, `total_price`, `booking_date`,
`contact_number`, `tax`, `number_of_person`, `promo_id`, `booking_status`) VALUES
('457894', 13, '2021-05-11', '2021-05-13', '2500', '2021-05-03', '123456789', '31', 2, 'promo_1',
'Booked'),
('785425', 10, '2021-06-16', '2021-06-17', '1000', '2021-04-07', '75842459', '20', 1, 'promo_2', '');

-- --------------------------------------------------------

--
-- Table structure for table `card_info`
--

CREATE TABLE `card_info` (
  `card_number` varchar(20) NOT NULL,
  `user_id` int(11) NOT NULL,
```

```
  `cvv` int(11) NOT NULL,
  `expiry_date` date NOT NULL,
  `isdefault` tinyint(1) NOT NULL
) ENGINE=InnoDB DEFAULT CHARSET=utf8;

--
-- Dumping data for table `card_info`
--

INSERT INTO `card_info` (`card_number`, `user_id`, `cvv`, `expiry_date`, `isdefault`) VALUES
('4242424242424242', 9, 245, '2024-05-13', 1),
('5353535353535353', 13, 11, '2025-12-09', 0);

-- --------------------------------------------------------

--
-- Table structure for table `category`
--

CREATE TABLE `category` (
  `category_id` varchar(20) NOT NULL,
  `category_name` varchar(32) NOT NULL
) ENGINE=InnoDB DEFAULT CHARSET=utf8;

--
-- Dumping data for table `category`
--

INSERT INTO `category` (`category_id`, `category_name`) VALUES
('1', 'hotels and resorts'),
('2', 'independent house'),
('3', 'shared room'),
('4', 'guesthouse');

-- --------------------------------------------------------

--
-- Table structure for table `employee`
--

CREATE TABLE `employee` (
  `employee_id` int(11) NOT NULL,
  `property_id` int(11) NOT NULL,
  `email_id` varchar(32) NOT NULL,
```

```
  `contact_number` varchar(15) NOT NULL,
  `fname` varchar(20) NOT NULL,
  `lname` varchar(20) NOT NULL,
  `address` varchar(32) NOT NULL
) ENGINE=InnoDB DEFAULT CHARSET=utf8;

--
-- Dumping data for table `employee`
--

INSERT INTO `employee` (`employee_id`, `property_id`, `email_id`, `contact_number`, `fname`, `lname`,
`address`) VALUES
(556546, 123460, 'sam.sum@airbnb.com', '4665445612', 'Sam', 'Summers', 'Dallas'),
(556547, 123459, 'mark.twin@airbnb.com', '4245153554', 'Mark', 'Twin', 'South Dallas');

-- --------------------------------------------------------

--
-- Table structure for table `host`
--

CREATE TABLE `host` (
  `host_id` varchar(20) NOT NULL,
  `user_id` int(11) NOT NULL,
  `account_number` varchar(20) NOT NULL,
  `bank_type` varchar(20) NOT NULL
) ENGINE=InnoDB DEFAULT CHARSET=utf8;

--
-- Dumping data for table `host`
--

INSERT INTO `host` (`host_id`, `user_id`, `account_number`, `bank_type`) VALUES
('akash1', 6, '444444444', 'Wells Fargo'),
('pravin1', 2, '142354645', 'chase'),
('sanjay1', 4, '55555555555', 'chase');

-- --------------------------------------------------------

--
-- Table structure for table `payment_details`
--

CREATE TABLE `payment_details` (
```

```
  `payment_id` varchar(20) NOT NULL,
  `booking_id` varchar(10) NOT NULL,
  `airbnb_share` decimal(10,0) NOT NULL,
  `host_share` decimal(10,0) NOT NULL,
  `refund` decimal(10,0) NOT NULL,
  `card_number` varchar(20) NOT NULL
) ENGINE=InnoDB DEFAULT CHARSET=utf8;

--
-- Dumping data for table `payment_details`
--

INSERT INTO `payment_details` (`payment_id`, `booking_id`, `airbnb_share`, `host_share`, `refund`,
`card_number`) VALUES
('147852369', '457894', '30', '2510', '0', '42424242424242'),
('456146', '457894', '201', '121', '0', '353542246465'),
('963258741', '785425', '10', '990', '0', '5353535353535353');

--
-- Triggers `payment_details`
--
DELIMITER $$
CREATE TRIGGER `booking_status` AFTER INSERT ON `payment_details` FOR EACH ROW update booking
set booking_status='Booked' where booking_id = new.booking_id
$$
DELIMITER ;

-- --------------------------------------------------------

--
-- Table structure for table `promocodes`
--

CREATE TABLE `promocodes` (
  `promo_id` varchar(10) NOT NULL,
  `promo_code` varchar(10) NOT NULL,
  `description` varchar(32) NOT NULL,
  `expiry_date` date NOT NULL
) ENGINE=InnoDB DEFAULT CHARSET=utf8;

--
-- Dumping data for table `promocodes`
--
```

```
INSERT INTO `promocodes` (`promo_id`, `promo_code`, `description`, `expiry_date`) VALUES
('promo_1', 'SAVE30', 'Save 30% on your booking', '2021-05-12'),
('promo_2', 'SUMMER50', 'Save 50% on your summer holiday', '2021-05-19');

-- --------------------------------------------------------

--
-- Table structure for table `property`
--

CREATE TABLE `property` (
  `property_id` int(11) NOT NULL,
  `property_name` varchar(32) NOT NULL,
  `address` varchar(32) NOT NULL,
  `desciption` varchar(32) NOT NULL,
  `cost` int(11) NOT NULL,
  `host_id` varchar(20) NOT NULL,
  `last_updated` date NOT NULL,
  `availability` int(11) NOT NULL,
  `category_id` varchar(20) NOT NULL,
  `review_count` int(11) NOT NULL
) ENGINE=InnoDB DEFAULT CHARSET=utf8;

--
-- Dumping data for table `property`
--

INSERT INTO `property` (`property_id`, `property_name`, `address`, `desciption`, `cost`, `host_id`,
`last_updated`, `availability`, `category_id`, `review_count`) VALUES
(123456, 'Hotel Phoenix', 'Frankford Rd, Dallas', 'Hotel and Resort', 1000, 'pravin1', '2021-05-08', 10, '1',
0),
(123457, 'Tiny House', 'Coit Road, Dallas', 'Independent house with 2 bedroom', 750, 'sanjay1', '2021-05-
09', 1, '2', 0),
(123459, 'Emrald Leaf', 'Central Dallas', 'Entire Apartment', 560, 'akash1', '2021-05-09', 2, '4', 0),
(123460, 'City view', 'South Dallas', 'Shared room in 2 persons', 150, 'pravin1', '2021-05-09', 1, '3', 1);

-- --------------------------------------------------------

--
-- Table structure for table `property_image`
--

CREATE TABLE `property_image` (
  `property_id` int(11) NOT NULL,
```

```
  `property_image` varchar(20) NOT NULL
) ENGINE=InnoDB DEFAULT CHARSET=utf8;

--
-- Dumping data for table `property_image`
--

INSERT INTO `property_image` (`property_id`, `property_image`) VALUES
(123460, 'City View Image 1'),
(123460, 'City view Image 2');

-- --------------------------------------------------------

--
-- Table structure for table `review`
--

CREATE TABLE `review` (
  `review_id` varchar(20) NOT NULL,
  `property_id` int(11) NOT NULL,
  `rating` float NOT NULL,
  `review_description` varchar(155) NOT NULL,
  `review_date` date NOT NULL
) ENGINE=InnoDB DEFAULT CHARSET=utf8;

--
-- Dumping data for table `review`
--

INSERT INTO `review` (`review_id`, `property_id`, `rating`, `review_description`, `review_date`) VALUES
('11', 123460, 3, 'Awesome Experience', '2021-05-09'),
('review_1', 123460, 4, 'It was a nice experience', '2021-05-04'),
('review_2', 123459, 2, 'Worst experience', '2021-05-03');

--
-- Triggers `review`
--
DELIMITER $$
CREATE TRIGGER `updateReviewCount` AFTER INSERT ON `review` FOR EACH ROW BEGIN
DECLARE no_of_reviews INTEGER;
SELECT review_count INTO no_of_reviews FROM property WHERE   property_id  = NEW.property_id;
UPDATE property SET review_count = no_of_reviews +1 WHERE property_id = NEW.property_id;
END
$$
```

```
DELIMITER ;

-- --------------------------------------------------------

--
-- Table structure for table `review_image`
--

CREATE TABLE `review_image` (
  `review_id` varchar(11) NOT NULL,
  `image_name` varchar(32) NOT NULL,
  `image_url` varchar(32) NOT NULL
) ENGINE=InnoDB DEFAULT CHARSET=utf8;

--
-- Dumping data for table `review_image`
--

INSERT INTO `review_image` (`review_id`, `image_name`, `image_url`) VALUES
('review_1', 'living room', 'living room image url'),
('review_2', 'view', 'view image url');

-- --------------------------------------------------------

--
-- Table structure for table `user`
--

CREATE TABLE `user` (
  `user_id` int(11) NOT NULL,
  `email_id` text NOT NULL,
  `contact_number` varchar(13) NOT NULL,
  `fname` text NOT NULL,
  `lname` text NOT NULL,
  `user_type` varchar(10) NOT NULL,
  `password` varchar(20) NOT NULL
) ENGINE=InnoDB DEFAULT CHARSET=utf8;

--
-- Dumping data for table `user`
--

INSERT INTO `user` (`user_id`, `email_id`, `contact_number`, `fname`, `lname`, `user_type`, `password`)
VALUES
```

```
(2, 'pravin.ghorpade@gmail.com', '4694271986', 'Pravin', 'Ghorpade', 'host', '123456789'),
(4, 'sanjay.jayraj@gmail.com', '4545454545', 'Sanjay', 'Jayraj', 'host', '123456789'),
(6, 'Akash R B', '8798451243', 'Akash', 'R B', 'host', '987654321'),
(9, 'RobertTCruz@dayrep.com\r\n', '4016613641', 'Robert ', 'T. Cruz\r\n', 'customer', 'ud7eLae0Ahch'),
(10, 'KatharinaFWatson@armyspy.com\r\n', '3108151798', 'Katharina ', 'Watson', 'customer',
'ahRoh4ax'),
(13, 'JohnALeasure@armyspy.com\r\n', '9378406216', 'John ', 'Leasure', 'customer', 'Kah4iewoo');

-- --------------------------------------------------------

--
-- Table structure for table `wishlist`
--

CREATE TABLE `wishlist` (
  `user_id` int(11) NOT NULL,
  `property_id` int(11) NOT NULL,
  `date_added` date NOT NULL
) ENGINE=InnoDB DEFAULT CHARSET=utf8;

--
-- Dumping data for table `wishlist`
--

INSERT INTO `wishlist` (`user_id`, `property_id`, `date_added`) VALUES
(13, 123460, '2021-05-04'),
(13, 123459, '2021-05-01');

--
-- Indexes for dumped tables
--

--
-- Indexes for table `amenities`
--
ALTER TABLE `amenities`
  ADD PRIMARY KEY (`amenities_id`);

--
-- Indexes for table `available_amenities`
--
ALTER TABLE `available_amenities`
  ADD KEY `aa_amenities_id` (`amenities_id`),
  ADD KEY `aa_property_id` (`property_id`);
```

```
--
-- Indexes for table `booking`
--
ALTER TABLE `booking`
  ADD PRIMARY KEY (`booking_id`),
  ADD KEY `bk_user_id_1` (`user_id`),
  ADD KEY `bk_promo_id` (`promo_id`);

--
-- Indexes for table `card_info`
--
ALTER TABLE `card_info`
  ADD PRIMARY KEY (`card_number`),
  ADD KEY `ci_user_id_1` (`user_id`);

--
-- Indexes for table `category`
--
ALTER TABLE `category`
  ADD PRIMARY KEY (`category_id`);

--
-- Indexes for table `employee`
--
ALTER TABLE `employee`
  ADD PRIMARY KEY (`employee_id`),
  ADD KEY `em_property_id_1` (`property_id`);

--
-- Indexes for table `host`
--
ALTER TABLE `host`
  ADD UNIQUE KEY `host_id` (`host_id`) USING BTREE,
  ADD KEY `user_id` (`user_id`);

--
-- Indexes for table `payment_details`
--
ALTER TABLE `payment_details`
  ADD PRIMARY KEY (`payment_id`),
  ADD KEY `pd_booking_id_1` (`booking_id`);

--
```

```
-- Indexes for table `promocodes`
--
ALTER TABLE `promocodes`
  ADD PRIMARY KEY (`promo_id`);

--
-- Indexes for table `property`
--
ALTER TABLE `property`
  ADD PRIMARY KEY (`property_id`),
  ADD KEY `fk_host_id_1` (`host_id`),
  ADD KEY `pr_category_id_1` (`category_id`);

--
-- Indexes for table `property_image`
--
ALTER TABLE `property_image`
  ADD KEY `pi_property_id_1` (`property_id`);

--
-- Indexes for table `review`
--
ALTER TABLE `review`
  ADD PRIMARY KEY (`review_id`),
  ADD KEY `rw_property_id_1` (`property_id`);

--
-- Indexes for table `review_image`
--
ALTER TABLE `review_image`
  ADD KEY `ri_review_id_1` (`review_id`);

--
-- Indexes for table `user`
--
ALTER TABLE `user`
  ADD PRIMARY KEY (`user_id`),
  ADD KEY `user_id` (`user_id`);

--
-- Indexes for table `wishlist`
--
ALTER TABLE `wishlist`
  ADD KEY `wl_user_id_1` (`user_id`),
```

```
  ADD KEY `wl_property_id_1` (`property_id`);

--
-- AUTO_INCREMENT for dumped tables
--

--
-- AUTO_INCREMENT for table `employee`
--
ALTER TABLE `employee`
  MODIFY `employee_id` int(11) NOT NULL AUTO_INCREMENT, AUTO_INCREMENT=556549;

--
-- AUTO_INCREMENT for table `property`
--
ALTER TABLE `property`
  MODIFY `property_id` int(11) NOT NULL AUTO_INCREMENT, AUTO_INCREMENT=123461;

--
-- AUTO_INCREMENT for table `user`
--
ALTER TABLE `user`
  MODIFY `user_id` int(11) NOT NULL AUTO_INCREMENT, AUTO_INCREMENT=14;

--
-- Constraints for dumped tables
--

--
-- Constraints for table `available_amenities`
--
ALTER TABLE `available_amenities`
  ADD CONSTRAINT `aa_amenities_id` FOREIGN KEY (`amenities_id`) REFERENCES `amenities`
(`amenities_id`),
  ADD CONSTRAINT `aa_property_id` FOREIGN KEY (`property_id`) REFERENCES `property`
(`property_id`);

--
-- Constraints for table `booking`
--
ALTER TABLE `booking`
  ADD CONSTRAINT `bk_promo_id` FOREIGN KEY (`promo_id`) REFERENCES `promocodes` (`promo_id`),
  ADD CONSTRAINT `bk_user_id_1` FOREIGN KEY (`user_id`) REFERENCES `user` (`user_id`);
```

```
--
-- Constraints for table `card_info`
--
ALTER TABLE `card_info`
  ADD CONSTRAINT `ci_user_id_1` FOREIGN KEY (`user_id`) REFERENCES `user` (`user_id`);

--
-- Constraints for table `employee`
--
ALTER TABLE `employee`
  ADD CONSTRAINT `em_property_id_1` FOREIGN KEY (`property_id`) REFERENCES `property`
(`property_id`);

--
-- Constraints for table `host`
--
ALTER TABLE `host`
  ADD CONSTRAINT `fk_user_id_1` FOREIGN KEY (`user_id`) REFERENCES `user` (`user_id`);

--
-- Constraints for table `payment_details`
--
ALTER TABLE `payment_details`
  ADD CONSTRAINT `pd_booking_id_1` FOREIGN KEY (`booking_id`) REFERENCES `booking`
(`booking_id`);

--
-- Constraints for table `property`
--
ALTER TABLE `property`
  ADD CONSTRAINT `fk_host_id_1` FOREIGN KEY (`host_id`) REFERENCES `host` (`host_id`),
  ADD CONSTRAINT `pr_category_id_1` FOREIGN KEY (`category_id`) REFERENCES `category`
(`category_id`);

--
-- Constraints for table `property_image`
--
ALTER TABLE `property_image`
  ADD CONSTRAINT `pi_property_id_1` FOREIGN KEY (`property_id`) REFERENCES `property`
(`property_id`);

--
-- Constraints for table `review`
--
```

```
ALTER TABLE `review`
  ADD CONSTRAINT `rw_property_id_1` FOREIGN KEY (`property_id`) REFERENCES `property`
(`property_id`);

--
-- Constraints for table `review_image`
--
ALTER TABLE `review_image`
  ADD CONSTRAINT `ri_review_id_1` FOREIGN KEY (`review_id`) REFERENCES `review` (`review_id`);

--
-- Constraints for table `wishlist`
--
ALTER TABLE `wishlist`
  ADD CONSTRAINT `wl_property_id_1` FOREIGN KEY (`property_id`) REFERENCES `property`
(`property_id`),
  ADD CONSTRAINT `wl_user_id_1` FOREIGN KEY (`user_id`) REFERENCES `user` (`user_id`);
COMMIT;
```

Stored Procedure

```
CREATE OR REPLACE PROCEDURE add_card_info (
card_number IN varchar,
user_id          IN int,
cvv                        IN NUMBER
expiry_date IN DATE,
) AS
BEGIN
INSERT INTO card_info VALUES (card_number, user_id, cvv, expiry_date,0);

END add_card_info;
--------------------------------------------------------------------------------


CREATE OR REPLACE PROCEDURE set_default_card_info(
card_number_var        IN INT,
user_id_var IN int,)AS
BEGIN
UPDATE card_info SET
is_default = 1 WHERE
user_id = user_id_var AND card_number = card_id_var;


END set_default_card_info;
--------------------------------------------------------------------------------

CREATE OR REPLACE PROCEDURE add_property(
property_id      IN INTEGER,
property_name IN VARCHAR,
address          IN VARCHAR,
description IN VARCHAR,
cost               IN       INT,
host_id          IN       VARCHAR,
last_updated IN DATE,
availability IN INT,
category_id IN  VARCHAR
image_url        IN VARCHAR) AS
BEGIN
INSERT INTO product VALUES (
property_id, property_name, address,description,cost,host_id,
last_updated,availability,category_id);
```

```
INSERT INTO property_image VALUES ( product_id,
image_url
);

END add_product;
-----------------------------------------------------------------------------------------------------
CREATE OR REPLACE PROCEDURE add_to_wish_list (
user_id IN INT,
property_id IN INT,
)AS
BEGIN
INSERT INTO wish_list VALUES ( user_id,
product_id
sysdate
);

END add_to_wish_list;


-----------------------------------------------------------------------------------------------------
CREATE OR REPLACE PROCEDURE populate_property_categories AS BEGIN
INSERT INTO category VALUES ( '1',
'hotels and resorts'
);

INSERT INTO category VALUES ( '2',
'independent house'
);

INSERT INTO category VALUES ( '3',
'shared room'
);

INSERT INTO category VALUES ( '4',
'guesthouse'
);

END populate_product_categories;

-----------------------------------------------------------------------------------
```

Triggers

```
CREATE TRIGGER `booking_status_1`
AFTER INSERT ON `payment_details`
FOR EACH ROW
update booking set booking_status='Booked' where booking_id = new.booking_id
```

```
DROP TRIGGER IF EXISTS `updateReviewCount`;
CREATE DEFINER=`root`@`localhost`
 TRIGGER `updateReviewCount`
AFTER INSERT ON `review`
FOR EACH ROW
BEGIN
DECLARE no_of_reviews INTEGER;
SELECT review_count
INTO no_of_reviews
FROM property WHERE property_id = NEW.property_id;
UPDATE property SET review_count = no_of_reviews +1
WHERE property_id = NEW.property_id;
END
```

```
Triggers `payment_details`-
DELIMITER $$
CREATE TRIGGER `booking_status` AFTER INSERT ON `payment_details` FOR EACH ROW update booking
set booking_status='Booked' where booking_id = new.booking_id
$$
DELIMITER ;
```