Specification Document

1) What roles (person/user groups) are there?

- hosts
- guests
- administrators

2) What actions do these roles perform?

- host:
    - makes profile
    - rent out rooms/places/houses
    - chat with potential guests
    - charge payment
- guest:
    - makes profile
    - browse places to rent
    - find deals
    - chat with hosts
    - provide information
    - make bookings
    - make payments
- administrators:
    - store and manage host and guest information
    - store and manage conversation messages between host and guest
    - store and manage address locations of rooms/homes
    - store and manage reviews, host levels
    - store and manage payment transactions
    - manage availability of locations for rent

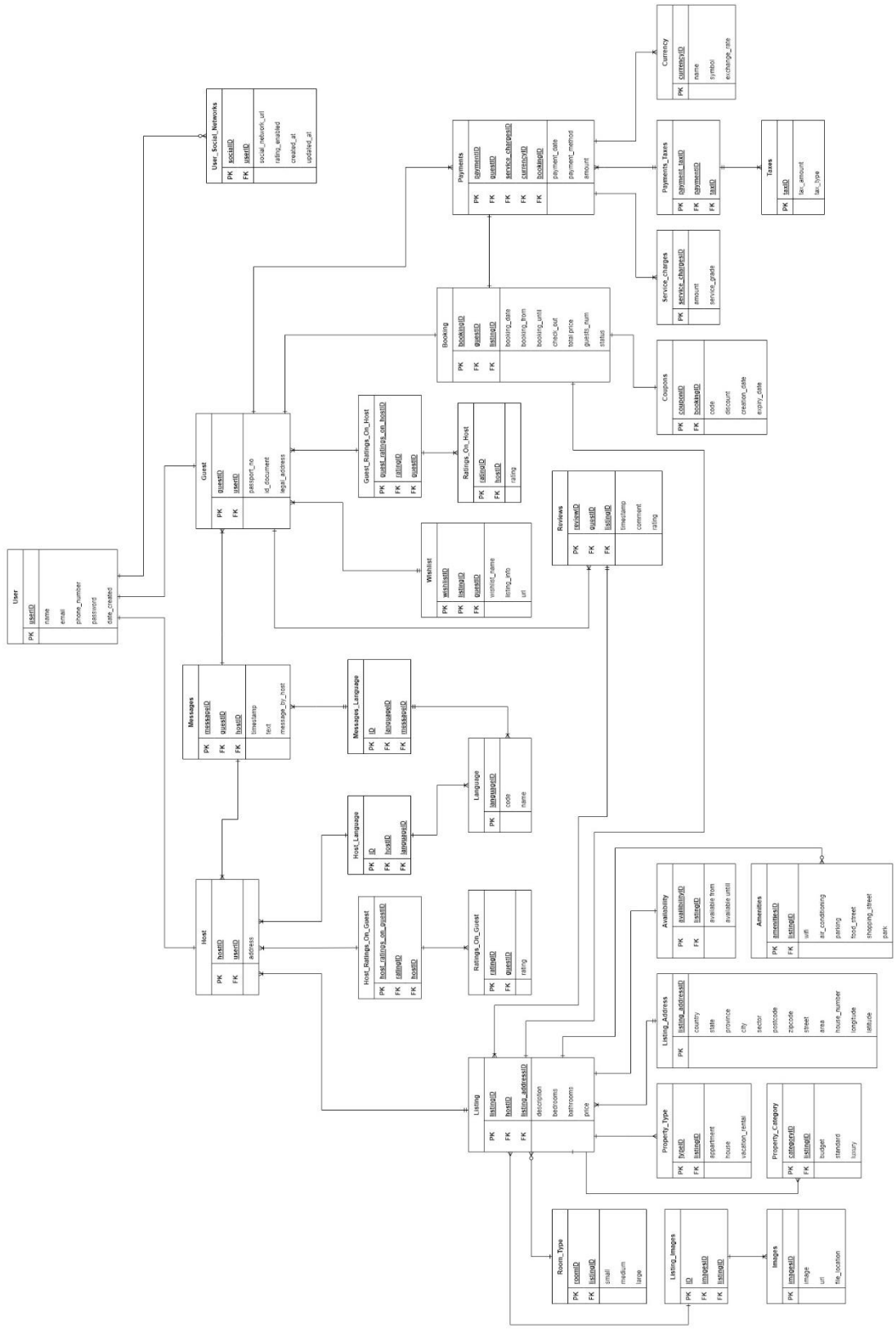
3) Which data and functions are required?

Data	Functions
host data	How many locations does a host own?
guest data	How many guest users from the guest user base make bookings in a given time period?
locations data	Which locations have been booked the most?
transaction data	Which were the largest transactions?
price changes data	What is the sum of all transactions in a given month?
guest reviews data	How many guests give good reviews to a given location?

Half Page Summary:

We have to create a database for AirBnB business. The first step is to find out the necessary entities and their attributes that will be included in the database. For this we consider how many different groups are involved in the process. These are the hosts, guests and AirBnB's administrators. Then we identify how the guests and hosts will behave, how they will interact with the AirBnB website. How administrators will behave and what they will do with users – the guests and hosts. Their different actions become part of the database entities such as: guests make account on website, browse places, talk to hosts and make bookings. Similarly, hosts make account on website, upload photos of their rooms offered for rent, update prices etcetera. AirBnB's administrators store information in their database such as information of hosts and guests, locations, reviews, transaction details, coupons and deals etcetera. After all entities with their attributes have been identified, an Entity-Relationship Model is created and their cardinality is defined.

Entity Relationship Model



Data Dictionary

Entity	Description
User	Contains information about registered users of the platform such as name, email, password and account details. Attributes :- INTEGER: userID VARCHAR: name, email, password, phone_number DATETIME: date_created
Host	Contains information about individuals who have listed properties for rent, such as their name, contact information, and property listings. Attributes :- INTEGER: hostID, userID VARCHAR: address
Listing	Contains information about the properties available for rent such as location, description, number of bedrooms, and photos. Attributes :- INTEGER: hostID, listingID, listing_addressID, bedrooms, bathrooms DECIMAL: price VARCHAR: description
Listing_Address	Contains information about the locations of properties, including the city, state, and country. Attributes :- VARCHAR: country, state, province, city, sector, postcode, zipcode, street, area, house_num INTEGER: listing_addressID DECIMAL: longitude, latitude
Amenities	Contains information about the amenities available at each property, such as Wi-Fi, air conditioning, and parking. Attributes :- INTEGER: amenitiesID, listingID BOOLEAN: wifi, air_conditioning, parking, food_street, park, shopping_street, park
Propery_Type	Contains information about the type of property, such as apartment, house, or vacation rental. Attributes :- INTEGER: typeID, listingID BOOLEAN: appartment, house, vacation_rental
Property_Category	Contains information about the category of property, such as budget, standard, or luxury. Attributes :- INTEGER: categoryID, listingID BOOLEAN: budget, standard, luxury

Availibility	Contains information about the availability of each property, including the dates that are available. Attributes :- INTEGER: availabilityID, listingID DATE: available_from, available_until
Room_Type	Contains information about room size including small, medium and large Attributes:- INTEGER: roomID, listingID BOOLEAN: small, medium, large
Images	Contains information about the images associated with each property, including the file name and the URL. Attributes :- INTEGER: imageID BLOB: image TEXT: url, file_location
Listing_Images	Junction table between listing table and images table. Attributes:- INTEGER: ID, listingID, imageID
Guest	Contains information about individuals who rent out properties and become guests. Includes passport number of a guest and other identification documents. Attributes :- INTEGER: guestID, userID VARCHAR: passport_no, legal_address, id_document
Booking	Contains information about confirmed bookings, including the dates of the stay, the guests, and the payment details. Attributes :- INTEGER: bookingID, guestID, listingID, guest_num DATE: booking_from, booking_until DATETIME: booking_date, check_in, check_out DECIMAL: total_price TEXT: status
Payments	Contains information about payments made through the platform, including the amount, the payment method, and the date. Attributes :- INTEGER: paymentID, guestID, service_chargesID, currencyID, bookingID DATETIME: payment_date VARCHAR: payment_method DECIMAL: amount
Service_Charges	Contains information about the service fees charged by Airbnb, including the amount and the grade of service. Attributes :- INTEGER: serviceID VARCHAR: service_grade DECIMAL: amount

Taxes	Contains information about taxes charged on bookings, including the amount and the type of tax. Attributes :- INTEGER: taxID, VARCHAR: tax_type DECIMAL: tax_amount
Payments_Taxes	Junction table between payments table and taxes table. Attributes:- INTEGER: payments_taxesID, paymentID, taxID
Coupons	Contains information about coupons offered by Airbnb, including the code, the discount amount, expiration date etc. Attributes :- INTEGER: bookingID, couponID VARCHAR: code DECIMAL: discount DATETIME: creation_date, expiry_date
Currency	Contains information about the currency used for each booking, including the name, the symbol, and the exchange rate. Attributes :- INTEGER: currencyID VARCHAR: name, symbol DECIMAL: exchange_rate
Reviews	Contains information about reviews written by guests about their stay, including the date, the property, and the rating. Attributes :- INTEGER: reviewID, listingID, guestID, rating DATETIME: timestamp TEXT: comment
Messages	Contains information about messages exchanged between guests and hosts, such as the date, the subject, and the content. Attributes :- INTEGER: messageID, guestID, hostID DATETIME: timestamp VARCHAR: text BOOLEAN: message_by_host
Language	Contains information about the language preferences of users, including the language code and the display name. Attributes :- INTEGER: languageID VARCHAR: name, code
Messages_Language	Junction table between messages table and language table. Attributes:- INTEGER: ID, messageID, languageID

Host_Language	Junction table between host table and language table. Attributes:- INTEGER: ID, hostID, languageID
Ratings_On_Guest	Contains information about the ratings of a guest from the hosts. Atributes :- INTEGER: ratingID, guestID, rating
Host_Ratings_On_Guest	Junction table between host table and ratings_on_guest table. Attributes:- INTEGER: ID, ratingID, hostID
Ratings_On_Host	Contains information about the ratings of a host from the guests. Atributes :- INTEGER: ratingID, hostID, rating
Guest_Ratings_On_Host	Junction table between guest table and ratings_on_host table. Attributes:- INTEGER: ID, ratingID, guestID
User_Social_Networks	Contains information about social networks linked to users' profile for rating. Includes network name, creation and update dates and other details. Attributes :- INTEGER: socialID, userID VARCHAR: social_network BOOLEAN: rating_enabled DATETIME: created_at, updated_at
Wishlist	Contains information about the places the guests would like to rent in future in cluding the name of their list and the place. Attributes :- INTEGER: wishlistID, guestID, listingID VARCHAR: wishlist_name TEXT: listing_info, url

Airbnb Database

PROJECT – BUILD A DATAMART IN SQL

(DLBDSPBDM01)

DATE: 10-12-2024

NAME: MUHAMMAD MOHSIN KHAN

MATRICULATION NO.: 92123820

TUTOR: DOGER MUSHARAF

Contents

Introduction

The task is to create a database for the Airbnb business, so they can store data and monitor their business activities.

In the next section, a brief look will be taken at each table included in the database.

The rationale for the table will be discussed.

And, actual implementation of the tables in a Database Management System will be shown in picture.

Test cases for each table will be used to test functionality of the database. MariaDB and HeidiSQL frontend (which ships with MariaDB) software is used for this purpose.

Finally, some concluding remarks will be discussed about structure and potential improvements of the database design.

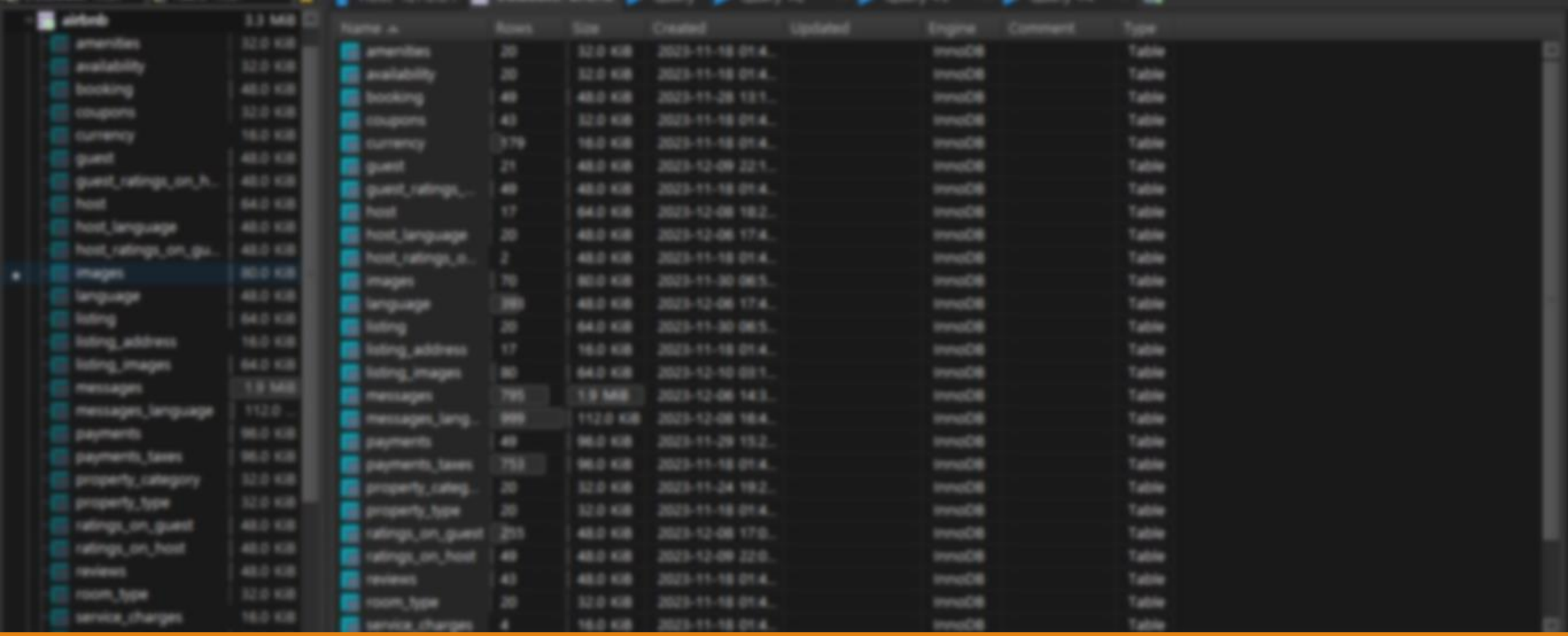


Figure 1

Database Tables

TABLE: user

```
CREATE TABLE `user` (
`userID` INTEGER(11) NOT NULL AUTO_INCREMENT,
`name` VARCHAR(100) NOT NULL,
`email` VARCHAR(100) NOT NULL,
`phone_number` VARCHAR(13) NOT NULL,
`password` VARCHAR(200) NOT NULL,
`date_created` DATETIME NOT NULL,
PRIMARY KEY (userID) );
```

Rationale:

Data needs to be stored about all users, a table is needed that can store user's name, email, phone number and password; and information privy to Airbnb such as user id and account creation date.

Test case:

```
SELECT * FROM user;
```

userID	name	email	phone_number	password	date_created
1	Kuruma Ikusai	kurumakun@japan.com	03534643523	vrejkgnljvdfwe312553	2023-10-06 05:25:57
2	Adam B. Junior	adamjunior@gmail.com	04126436347	tetete34S.	2023-08-10 03:22:11
3	Anna Hamza	annahamza9@gmail.com	05705466793	lhCTqABtQJjl5UP	2023-10-16 22:50:08
4	Emma Omar	emmaomar4@outlook.com	02965292380	e8su47w1cfxpHyO	2023-08-05 20:10:18
5	Jeronimo Mateo	jeronimomateo6@gmail.com	04542877133	FbHqVG3khGBCd1G	2023-09-11 10:04:48
6	Lena Mateo	lenamateo7@gmail.com	08636902364	2nrrnb3444waLWw7nE	2023-10-21 09:17:17
7	Omar Hatimi	omarhatimi7@hotmail.com	06161195341	QntZ6siz8	2023-10-22 22:01:53
8	Hussein Agustin	husseinagustin8@gmail.com	08032261939	jSvMD83fJrz0qGn	2023-09-08 11:26:48
9	Hasnaa Selim	hasnaaselim0@hotmail.com	06071849909	ftM4vcQe0BwA7h6	2023-09-30 11:06:07
10	Mila Nathan	milanathan6@microsoft.com	02585352003	u9dl7Jc8bM	2023-10-29 13:38:00
11	Miguel Angel Benjamin	miguelangelbenjamin3@hotmail.com	08141809877	J9PCuZbnhOtzSv8	2023-10-07 16:41:21
12	Sara Omar	saraomar2@microsoft.com	09012730365	u1uYNXD4ER6X	2023-08-10 17:33:11
13	Viktoria Noah	viktorianoah2@outlook.com	04087149004	Jy4GYqylum0gpR	2023-10-17 13:57:56
14	Mia Oliver	miaoliver4@yahoo.com	05040346441	Tbgaed2Wed95rak	2023-09-25 10:14:16
15	Felipe Thomas	felipethomas5@gmail.com	04552269639	JZp555RXCqCqXeUS	2023-08-01 07:15:20
16	Jacob Abdel-Rahman	jacobabdel-rahman7@gmail.com	01376473475	HM7cy1Q2aUb6JAo	2023-09-29 15:00:18
17	Shaimaa Saeed	shaimaasaeed4@yahoo.com	07646640697	1J3eCwI8wrrYLB	2023-10-17 16:37:10
18	Lea Hamza	leahamza5@hotmail.com	01926040235	3FKFxbwI5KClJfW	2023-10-22 08:35:41
19	Thiago Ibrahim	thiagoibrahim9@yahoo.com	08320815283	0yeVDyBTmC4kv	2023-10-24 08:31:30
20	Juan Bilal	juanbilal3@yahoo.com	05039234208	aQKBs7d7b3OmTUT	2023-10-15 08:07:51

Figure 2

TABLE: host

```
CREATE TABLE `host` (
`hostID` INTEGER(11) NOT NULL AUTO_INCREMENT,
`userID` INTEGER(11) NOT NULL,
`address` VARCHAR(200) NOT NULL,
PRIMARY KEY (hostID),
CONSTRAINT `fk_userid_host` FOREIGN KEY
(`userID`) REFERENCES `user` (`userID`) );
```

Test case:

```
SELECT * FROM host;
```

Rationale:

Users that are hosts can be stored in their separate table with the specific information data required by Airbnb. This also helps create hierarchical relationships between host table and related tables.

hostID	userID	address
1	1	323-1274, Toyogaoka, Tsukigata-cho Kabato-...
2	2	Suite 762 2303 Ariel Prairie, Schummbury, TN...
3	3	Suite 245 4255 Cummings Turnpike, Juliansid...
4	4	Apt. 337 Theodor-Heuss-Ring 92b, West Alia,...
5	5	Wiembachallee 7, Berndberg, BE 72528, Germ...
6	6	Bloque Manuel, 4 Puerta 876, Santiago de C...
7	7	Jl. Kartini No. 85, Bantul, SS 24756, Indonesia
8	8	Apt. 670 Jl. Rasuna Said No. 99, Tanjung Jab...
9	9	Rambla Luis Miguel, 9, Bilbao, Rio 64215, Arg...
10	10	Apt. 518 415 Shemeka Garden, Effertzhaven, ...
11	11	Via Gallo 836, Quarto Olimpia, PE 42018, Italy
12	12	1 hoog Kalikaplantsoen 595, Oud Osgeest, O...
13	13	6729 Owen Mount, Theoton, IA 84715, Austr...
14	14	55936 Marvin Plains, Fisherhaven, HI 77329-5...
15	15	Suite 670 6742 Koelpin Locks, North Denismo...
16	16	608 Isaias Forks, Miaview, FL 49802
17	17	Apt. 992 Pfarrer-Klein-Str. 67b, Lianstadt, TH ...
18	18	Zimmer 258 Mühlenweg 90b, Bruhnsstadt, S...
19	19	Apt. 129 Weidenstr. 8, Tischlerberg, HH 86694
20	20	Quarzstr. 2, Schön Carolinscheid, NI 36707

Figure 3

TABLE: listing

```
CREATE TABLE `listing` (
`listingID` INTEGER(11) AUTO_INCREMENT,
`hostID` INTEGER(11) NOT NULL,
`listing_addressID` INTEGER(11),
`description` TEXT NOT NULL,
`bedrooms` INTEGER(1) NOT NULL,
`bathrooms` INTEGER(1) NOT NULL,
`price` DECIMAL(10,5) NOT NULL,
PRIMARY KEY (`listingID`),
CONSTRAINT `fk_hostid_listing`
FOREIGN KEY (`hostID`)
REFERENCES `host` (`hostID`) );
ALTER TABLE `listing` ADD CONSTRAINT
`fk_listing_addressid_listing`
FOREIGN KEY (`listing_addressID`)
REFERENCES `listing_address`
(`listing_addressID`);
```

Test case:

```
SELECT * FROM listing;
```

Rationale:

This table will store general information of registered hosts' places. It will be used to show info to the guests looking for a place on Airbnb's website or app.

listingID	hostID	listing_addressID	description	bedrooms	bathrooms	price
1	1	1	This charming studio apartment is nestled in ...	4	3	250.00000
2	2	2	A four-bedroom house in a family-friendly n...	4	2	100.00000
3	3	3	This sleek loft apartment offers breathtaking ...	2	1	150.00000
4	4	4	A picturesque cottage surrounded by lush gr...	2	2	120.00000
5	5	5	A picturesque cottage surrounded by lush gr...	3	3	220.00000
6	6	6	A cozy bungalow just steps away from the b...	5	7	300.00000
7	7	7	This elegant townhouse, steeped in history, f...	2	2	170.00000
8	8	8	A stylish duplex with a modern aesthetic. Thi...	3	3	250.00000
9	9	9	Located in a historic district, this meticulousl...	4	5	195.00000
10	10	10	A unique eco-friendly home surrounded by ...	3	3	160.00000
11	11	11	A luxurious penthouse with floor-to-ceiling w...	5	8	350.00000
12	12	12	Set on acres of farmland, this renovated far...	4	5	300.00000
13	13	13	A contemporary townhome in a vibrant neig...	2	4	280.00000
14	14	14	A serene lakeside property with panoramic vi...	3	3	240.00000
15	15	15	Located in the heart of the arts district, this l...	7	5	400.00000
16	16	16	A five-bedroom house in a quiet suburban n...	5	4	230.00000
17	17	17	A classic brownstone with historic charm and...	4	4	245.00000
18	18	18	A studio apartment in an iconic Art Deco bui...	1	1	140.00000
19	19	19	A charming chalet nestled in the mountains, ...	2	2	320.00000
20	20	20	A sleek condo overlooking the river, featurin...	1	1	150.00000

Figure 4

TABLE: listing_address

```
CREATE TABLE `listing_address` (
`listing_addressID` INT(11) NOT NULL
AUTO_INCREMENT,
`country` VARCHAR(50) NOT NULL,
`state` VARCHAR(50),
`province` VARCHAR(50) NOT NULL,
`city` VARCHAR(50) NOT NULL,
`sector` VARCHAR(50),
`postcode` VARCHAR(10),
`zipcode` INTEGER(10),
`area` VARCHAR(20),
`street` VARCHAR(50),
`house_number` VARCHAR(20) NOT NULL,
`longitude` DECIMAL(9,6) NOT NULL,
`latitude` DECIMAL(8,6) NOT NULL,
PRIMARY KEY (`listing_addressID`) );
```

Test case:

```
SELECT * FROM
listing_address;
```

Rationale:

This table stores addresses of all listings. Stores both address locations and geographic locations. PK is used as FK in `listing` table since many listings can have the same address, like a host owning multiple apartments in the same building.

listing_addressID	country	state	province	city	sector	postc
1	Japan	(NULL)	Hokkaido	Tsukigata-cho Kabato-gun	(NULL)	(N
2	USA	Nebraska	Osgeest	Oud	Kalikaplantsoen	(N
3	USA	FLorida	Miaview	Miaview	(NULL)	(N
4	Australia	Iowa	Throton	Owen Mount	(NULL)	84
5	USA	Hawaii	Hawaii	Fisherhaven	(NULL)	77
6	Germany	Hamburg	Tischlerberg	Tischlerberg	Weidenstr. 8	86
7	Germany	Thuringia	West Alia	Alia	(NULL)	(N
8	USA	Florida	Effertzhaven	Shemeka	(NULL)	10
9	Indonesia	Mahkamah Agung	Tanjung Jabung Timur	Tanjung Jabung Timur	(NULL)	33
10	Germany	Thuringia	Lianstadt	Lianstadt	(NULL)	17
11	Mexico		Ara	Santiago de Compostela	Puerta	15
12	Indonesia	(NULL)	Bantul	Kartini	(NULL)	24
13	Hungary	Nidiea	Carolinscheid	Schon Carolinscheid	(NULL)	(N
14	Argentina	(NULL)	Rio	Bilbao	9	64
15	USA	Washington	West	Julianside	Cummings Turnpike	89
16	USA	New York	North Denismouth	Locks	(NULL)	76
17	USA	Texas	Schummbery	Ariel Prairie	(NULL)	29
18	Italy	(NULL)	Peserie	Olimipa	(NULL)	42
19	Germany	(NULL)	Brezdtene	Berndberg	(NULL)	79
20	Germany	(NULL)	Silldazstadt	Bruhnsstadt	(NULL)	42

Figure 5

TABLE: property_category

```
CREATE TABLE `property_category` (
`categoryID` INTEGER(11) NOT NULL
AUTO_INCREMENT,
`listingID` INTEGER(11) NOT NULL,
`budget` BOOLEAN NOT NULL,
`standard` BOOLEAN NOT NULL,
`luxury` BOOLEAN NOT NULL,
PRIMARY KEY (`categoryID`),
CONSTRAINT `fk_listingid_property_category`
FOREIGN KEY (`listingID`) REFERENCES
`listing` (`listingID`) );
```

Test case:

```
SELECT * FROM
property_category;
```

Rationale:

This table stores the kind of a property. If in case there is a need to make changes to the kinds of properties, this table can be separately changed without affecting the overall structure of the database.

categoryID	listingID	budget	standard	luxury
1	1	0	1	0
2	2	1	0	0
3	3	0	1	0
4	4	1	0	0
5	5	0	1	0
6	6	0	0	1
7	7	0	1	0
8	8	0	0	1
9	9	0	0	1
10	10	0	1	0
11	11	0	0	1
12	12	0	0	1
13	13	0	0	1
14	14	0	0	1
15	15	0	0	1
16	16	0	0	1
17	17	0	0	1
18	18	0	1	0
19	19	0	0	1
20	20	1	0	0

Figure 6

TABLE: property_type

```
CREATE TABLE `property_type` (
`typeID` INTEGER(11) NOT NULL AUTO_INCREMENT,
`listingID` INTEGER(11),
`appartment` BOOLEAN NOT NULL,
`house` BOOLEAN NOT NULL,
`vacation_rental` BOOLEAN NOT NULL,
PRIMARY KEY (`typeID`),
CONSTRAINT `fk_listingid_property_type` FOREIGN
KEY (`listingID`) REFERENCES
`listing` (`listingID`) );
```

Test case:

```
SELECT * FROM property_type;
```

Rationale:

This table stores the type of properties. Any future additions of types can be stored in this table because it references to listing table so it would not break the database.

typeID	listingID	appartment	house	vacation_rental
1	1	1	0	0
2	2	0	1	0
3	3	1	0	0
4	4	0	0	1
5	5	0	0	1
6	6	0	1	0
7	7	0	0	1
8	8	0	0	1
9	9	0	0	1
10	10	0	1	0
11	11	0	0	1
12	12	0	0	1
13	13	0	1	0
14	14	0	0	1
15	15	1	0	0
16	16	0	1	0
17	17	0	1	0
18	18	1	0	0
19	19	1	0	0
20	20	1	0	0

Figure 7

TABLE: amenities

```
CREATE TABLE `amenities` (
`amenitiesID` INTEGER(11) NOT NULL
AUTO_INCREMENT,
`listingID` INTEGER(11),
`wifi` BOOLEAN NOT NULL,
`air_conditioning` BOOLEAN NOT NULL,
`parking` BOOLEAN NOT NULL,
`food_street` BOOLEAN,
`shopping_street` BOOLEAN,
`park` BOOLEAN,
PRIMARY KEY (`amenitiesID`),
CONSTRAINT `fk_listingid_amenities`
FOREIGN KEY (`listingID`) REFERENCES
`listing` (`listingID`) );
```

Test case:

```
SELECT * FROM amenities;
```

Rationale:

Each listing is going to have multiple amenities so a separate table can be used to store the amenities referencing that specific listing.

amenitiesID	listingID	wifi	air_conditioning	parking	food_street	shopping_street	park
1	1	1	1	1	0	0	(NULL)
2	2	1	1	0	0	0	0
3	3	1	1	1	1	0	0
4	4	1	1	0	0	0	0
5	5	1	1	0	0	0	0
6	6	1	1	0	0	0	0
7	7	1	1	0	0	0	0
8	8	1	1	0	0	0	0
9	9	1	1	0	0	0	0
10	10	1	1	0	0	0	0
11	11	1	1	0	0	0	0
12	12	1	1	0	0	0	0
13	13	1	1	0	0	0	0
14	14	1	1	0	0	0	0
15	15	1	1	0	0	0	0
16	16	1	1	0	0	0	0
17	17	1	1	0	0	0	0
18	18	1	1	0	0	0	0
19	19	1	1	0	0	0	0
20	20	1	1	0	0	0	0

Figure 8

TABLE: availability

```
CREATE TABLE `availability` (
`availabilityID` INTEGER(11) NOT NULL
AUTO_INCREMENT,
`listingID` INTEGER(11),
`available_from`  DATE NOT NULL,
`available_until` DATE NOT NULL,
PRIMARY KEY (`availabilityID`),
CONSTRAINT `fk_listingid_availability`
FOREIGN KEY (`listingID`) REFERENCES
`listing` (`listingID`) );
```

Test case:

```
SELECT * FROM availability;
```

Rationale:

Since a listing will have many booked slots and many free slots, a separate table can be used to store available dates for a listing

availabilityID	listingID	available_from	available_until
1	1	2023-12-04	2023-12-14
2	2	2023-12-05	2023-12-15
3	3	2023-12-08	2023-12-18
4	4	2023-12-10	2023-12-20
5	5	2023-12-12	2023-12-22
6	6	2023-12-15	2023-12-25
7	7	2023-12-18	2023-12-28
8	8	2023-12-20	2023-12-30
9	9	2023-12-23	2024-01-02
10	10	2023-12-25	2024-01-04
11	11	2023-12-28	2024-01-07
12	12	2023-12-30	2024-01-09
13	13	2024-01-02	2024-01-12
14	14	2024-01-04	2024-01-14
15	15	2024-01-07	2024-01-17
16	16	2024-01-09	2024-01-19
17	17	2024-01-12	2024-01-22
18	18	2024-01-14	2024-01-24
19	19	2024-01-17	2024-01-27
20	20	2024-01-19	2024-01-29

Figure 9

TABLE: room_type

```
CREATE TABLE `room_type` (
`roomID` INTEGER(11) NOT NULL AUTO_INCREMENT,
`listingID` INTEGER(11) NOT NULL,
`small` BOOLEAN NOT NULL,
`medium` BOOLEAN NOT NULL,
`large` BOOLEAN NOT NULL,
PRIMARY KEY (`roomID`),
CONSTRAINT `fk_listingid_room_type`
FOREIGN KEY (`listingID`)
REFERENCES `listing` (`listingID`) );
```

Test case:

```
SELECT * FROM room_type;
```

Rationale:

Other then a property's type or category, some guests would like to know how big, on average, individual rooms are. This table is used to store that information. Implementation detail such as actual room size and ranges is for Airbnb to decide.

roomID	listingID	small	medium	large
1	1	1	0	0
2	2	0	1	0
3	3	1	0	0
4	4	0	0	1
5	5	0	0	1
6	6	0	1	0
7	7	0	0	1
8	8	0	0	1
9	9	0	0	1
10	10	0	1	0
11	11	0	0	1
12	12	0	0	1
13	13	0	1	0
14	14	0	0	1
15	15	1	0	0
16	16	0	1	0
17	17	0	1	0
18	18	1	0	0
19	19	0	0	1
20	20	1	0	0

Figure 10

TABLE: images

```
CREATE TABLE `images` (
`imageID` INTEGER(15) NOT NULL AUTO_INCREMENT,
`image` MEDIUMBLOB,
`url` TEXT,
`file_location` TEXT,
PRIMARY KEY (`imageID`) );
```

Rationale:

Images for listings need to be stored. Images can be stored directly in the database or images can be stored to some other server where the urls of images can be stored, finally, images can be stored on a local storage and file paths can be stored in the database.

Test case:

```
SELECT * FROM images;
```

imageID	image	url	file_location
54	0x3435334134443646363837333639364532303...	www.unsplash.com/99-films-48mTwDzizqE	E:images99-films-48mTwDzizqE-unspla
55	0x3435334134443646363837333639364532303...	www.unsplash.com/alexandra-gorn-JIUjvqe2Z...	E:imagesalexandra-gorn-JIUjvqe2ZHg-
56	0x3435334134443646363837333639364532303...	www.unsplash.com/amira-aboalnaga-O7WjrXi...	E:imagesamira-aboalnaga-O7WjrXiKy_s
57	0x3435334134443646363837333639364532303...	www.unsplash.com/andrea-davis-qZTgRKioXcE	E:imagesandrea-davis-qZTgRKioXcE-ur
58	0x3435334134443646363837333639364532303...	www.unsplash.com/andy-vult-zwZpdhoTbU0	E:imagesandy-vult-zwZpdhoTbU0-unsp
59	0x3435334134443646363837333639364532303...	www.unsplash.com/ashley-byrd-yzkTCP4uc9E	E:imagesashley-byrd-yzkTCP4uc9E-uns
60	0x3435334134443646363837333639364532303...	www.unsplash.com/barthelemy-de-mazenod-r...	E:imagesarthelemy-de-mazenod-r_zK
61	0x3435334134443646363837333639364532303...	www.unsplash.com/bill-mackie-hK-ZADiFGvk	E:imagesill-mackie-hK-ZADiFGvk-unsp
62	0x3435334134443646363837333639364532303...	www.unsplash.com/chastity-cortijo-R-w5Q-4M...	E:imageschastity-cortijo-R-w5Q-4Mqm
63	0x3435334134443646363837333639364532303...	www.unsplash.com/christian-koch-D_4R9CcYZ...	E:imageschristian-koch-D_4R9CcYZOk-
64	0x3435334134443646363837333639364532303...	www.unsplash.com/christopher-jolly-GqbU78...	E:imageschristopher-jolly-GqbU78bdJF
65	0x3435334134443646363837333639364532303...	www.unsplash.com/collov-home-design-H-1j_...	E:imagescollov-home-design-H-1j_s0d
66	0x3435334134443646363837333639364532303...	www.unsplash.com/curology-ycEKahEaO5U	E:imagescurology-ycEKahEaO5U-unspl
67	0x3435334134443646363837333639364532303...	www.unsplash.com/deborah-cortelazzi-gREqu...	E:imagesdeborah-cortelazzi-gREquCUX
68	0x3435334134443646363837333639364532303...	www.unsplash.com/digital-marketing-agency-...	E:imagesdigital-marketing-agency-ntw
69	0x3435334134443646363837333639364532303...	www.unsplash.com/dillon-kydd-XGvwt544g8k	E:imagesdillon-kydd-XGvwt544g8k-uns
70	0x3435334134443646363837333639364532303...	www.unsplash.com/drew-coffman-jUOaONoX...	E:imagesdrew-coffman-jUOaONoXJQk-
71	0x3435334134443646363837333639364532303...	www.unsplash.com/eduardo-freire-1UH-uVzTi...	E:imageseduardo-freire-1UH-uVzTiDU-
72	0x3435334134443646363837333639364532303...	www.unsplash.com/erik-mclean-rFovKJV0llw	E:imageserik-mclean-rFovKJV0llw-unsp
73	0x3435334134443646363837333639364532303...	www.unsplash.com/frames-for-your-heart-2d...	E:imagesframes-for-your-heart-2d4lAQ

Figure 11

TABLE: listing_images

```
CREATE TABLE `listing_images` (
`ID` INTEGER(11) NOT NULL AUTO_INCREMENT,
`imageID` INTEGER(11) NOT NULL,
`listingID` INTEGER(11) NOT NULL,
PRIMARY KEY (`ID`),
CONSTRAINT `fk_imageid_listing_images` FOREIGN
KEY (`imageID`) REFERENCES `images`
(`imageID`),
CONSTRAINT
`fk_listingid_listing_images`
FOREIGN KEY (`listingID`)
REFERENCES `listing` (`listingID`) );
```

Test case:

```
SELECT * FROM listing_images;
```

Rationale:

Junction table for relating `listing` and `images` tables together.

ID	listingID	imageID
1	1	94
2	1	74
3	1	68
4	2	120
5	2	110
6	2	90
7	2	100
8	3	85
9	3	70
10	3	115
11	4	62
12	4	117
13	4	75
14	5	82
15	5	77
16	5	89
17	6	116
18	6	55
19	6	57
20	6	83
21	6	115

Figure 12

TABLE: guest

```
CREATE TABLE `guest` (
`guestID` INTEGER(11) NOT NULL AUTO_INCREMENT,
`userID` INTEGER(11) NOT NULL,
`passport_number` VARCHAR(14) NOT NULL,
`legal_address` VARCHAR(200) NOT NULL,
`id_document` VARCHAR(20),
PRIMARY KEY (`guestID`),
CONSTRAINT `fk_userid_guest`
FOREIGN KEY (`userID`)
REFERENCES `user` (`userID`) );
```

Rationale:

Users that sign up as guests can be identified in this table. Information specific to guest users is stored in this table.

Test case:

```
SELECT * FROM guest;
```

guestID	userID	passport_number	legal_address	id_document
2	21	HE0VQDJJ5	Suite 646 33510 Rogahn Field, Abshireton, A...	29102588967962
3	22	LZUHH6YGS	Parkstr. 21b, Jordanberg, BE 05624	55348262616537
4	23	BCE2DYBCV	Huerta Rosario 92, Elche, Can 61862	57681088232861
5	24	D8CPI63N3	Solar Ana Luisa, 9, Telde, Leo 01864	95261828871503
6	25	Y0D84WAZZ	Esc. 551 Extrarradio Antonio Candelaria 7 Pu...	77647104758315
7	26	FKC6XIG6N	Apt. 771 Jl. Rasuna Said No. 70, Palopo, GO ...	77689884600535
8	27	ODEAEGDXE	Via Renato 29, Appartamento 76, Rosalba lid...	29265276991812
9	28	L9HQX2VHR	Incrocio Ross 67, Ricci salentino, RA 47723	54876332406756
10	29	BE3KVPP6Q	1 hoog Tijmesweg 579 III, Oost Sofieberg, O...	71701764798962
11	30	HE3HE2MEW	Brüder-Bonhoeffer-Str. 7, Corvinscheid, HH 6...	83253877177886
12	31	EXODXEV5I	Im Kreuzbruch 98c, Neu Kenny, ST 84686	26795435038778
13	32	RIUK39IUN	Löchergraben 39a, Bad Markhagen, BE 47166	34622302754095
14	33	QM4PHUNMV	Zimmer 148 Hannah-Höch-Str. 84, Hoffmannf...	17691310037845
15	34	VKQ0OZ1OI	al. Frątczak 80283, Działdowo, DŚ 19-158	12942865981513
16	35	5QLC7SIKP	ul. Szeląg 705, Łuków, ŁD 26-836	52286231787854
17	36	8XNLC4XFQ	Apt. 499 830 Ritchie Pass, South Augustabur...	70233886625005
18	37	QDGP7QT3V	Suite 327 996 Thomas Summit, East Harrymo...	74408508387985
19	38	WOAYK0UB1	7507 Impasse Du Moulin, 66596 Athis-Mons	14027842250455
20	39	96H48OI2Q	Apt. 890 724 Sherly Islands, Edwardoberg, N...	66347792998084
21	40	Z16SIA51S	Chun Dong Lu 508hao, City Area - Minxing ...	34085627645616

Figure 13

TABLE: booking

```
CREATE TABLE `booking` (
`bookingID` INTEGER(20) NOT NULL
AUTO_INCREMENT,
`guestID` INTEGER(11) NOT NULL,
`listingID` INTEGER(11) NOT NULL,
`check_in` DATETIME,
`check_out` DATETIME,
`total_price` DECIMAL NOT NULL,
`guest_num` INTEGER NOT NULL,
`status` TEXT NOT NULL,
PRIMARY KEY (`bookingID`),
CONSTRAINT `fk_guestid_booking`
FOREIGN KEY (`guestID`) REFERENCES
`guest` (`guestID`),
CONSTRAINT `fk_listingid_booking`
FOREIGN KEY (`listingID`) REFERENCES
`listing` (`listingID`) );
```

Rationale:

A guest can make booking for a specific listing. Many guests can make bookings for many listings, so a table is needed to store the relevant data with this relationship.

bookingID	guestID	listingID	booking_date	booking_from	booking_until	check_in	check_out
40	15	6	2023-12-17 00:00:00	2023-12-29	2024-01-05	2023-12-29 11:22:56	2024-01-05 19:1
38	12	16	2024-01-09 00:00:00	2024-01-23	2024-01-30	2024-01-23 20:56:14	2024-01-30 04:5
35	12	16	2024-01-20 00:00:00	2024-01-23	2024-02-12	2024-01-23 23:27:09	2024-02-12 10:5
25	10	17	2024-01-27 00:00:00	2024-01-25	2024-02-09	2024-01-25 02:37:14	2024-02-09 19:4
36	12	19	2024-02-21 00:00:00	2024-01-30	2024-02-14	2024-01-30 01:51:54	2024-02-14 08:3
33	11	1	2024-01-16 00:00:00	2023-12-16	2023-12-31	2023-12-16 11:35:29	2023-12-31 19:0
43	15	13	2024-02-08 00:00:00	2024-01-15	2024-01-22	2024-01-15 21:08:09	2024-01-22 09:0
42	15	4	2023-11-29 00:00:00	2023-12-23	2024-01-12	2023-12-23 09:39:06	2024-01-12 22:3
39	13	12	2024-02-12 00:00:00	2024-01-13	2024-01-20	2024-01-13 23:21:59	2024-01-20 23:0
44	16	6	2023-12-04 00:00:00	2023-12-29	2024-01-28	2023-12-29 18:44:27	2024-01-28 03:1
37	12	16	2024-01-12 00:00:00	2024-01-23	2024-02-07	2024-01-23 05:57:50	2024-02-07 12:0
14	7	2	2023-11-26 00:00:00	2023-12-19	2024-01-03	2023-12-19 04:43:23	2024-01-03 03:3
20	9	13	2023-12-10 00:00:00	2024-01-15	2024-02-14	2024-01-15 07:09:32	2024-02-14 02:3
21	9	8	2023-12-28 00:00:00	2024-01-03	2024-01-23	2024-01-03 07:30:54	2024-01-23 11:1
3	3	12	2023-12-12 00:00:00	2024-01-13	2024-02-12	2024-01-13 23:55:45	2024-02-12 02:5
7	5	18	2024-03-01 00:00:00	2024-01-28	2024-02-27	2024-01-28 16:04:44	2024-02-27 22:3
15	7	8	2023-12-10 00:00:00	2024-01-03	2024-01-18	2024-01-03 15:13:50	2024-01-18 06:2
17	8	17	2024-02-18 00:00:00	2024-01-25	2024-02-09	2024-01-25 08:16:23	2024-02-09 19:0
48	16	19	2024-01-24 00:00:00	2024-01-30	2024-02-14	2024-01-30 22:49:01	2024-02-14 07:1
11	6	18	2024-02-09 00:00:00	2024-01-28	2024-02-17	2024-01-28 13:17:33	2024-02-17 22:3

Figure 14

Test case:

```
SELECT * FROM booking WHERE
total_price > 1000 ORDER BY
total_price;
```

TABLE: payments

```
CREATE TABLE `payments` (
`paymentID` INTEGER(11) NOT NULL
AUTO_INCREMENT,
`guestID` INTEGER(11) NOT NULL,
`service_chargesID` INTEGER(5) NOT NULL,
`currencyID` INTEGER(11) NOT NULL,
`bookingID` INTEGER(11) NOT NULL,
`payment_date` DATETIME NOT NULL,
`payment_method` VARCHAR(20) NOT NULL,
`amount` DECIMAL NOT NULL,
PRIMARY KEY (`paymentID`),
CONSTRAINT `fk_guestid_payments` FOREIGN KEY
(`guestID`) REFERENCES `guest` (`guestID`),
CONSTRAINT `fk_bookingid_payments`
FOREIGN KEY (`bookingID`) REFERENCES `booking`
(`bookingID`) );

ALTER TABLE `payments`
ADD CONSTRAINT `fk_service_chargesid_payments`
FOREIGN KEY (`service_chargesID`) REFERENCES
`service_charges` (`service_chargesID`),
ADD CONSTRAINT `fk_currency_payments` FOREIGN
KEY (`currencyID`) REFERENCES `currency`
(`currencyID`);
```

<u>Rationale:</u>

All the payments made by guests in their respective currencies for the bookings need to be stored. Specific company data such as service charges or date of payment also need to be stored. This table stores all such relevant data.

Test case:

```
SELECT * FROM payments WHERE
payment_date > 2023-09-01 AND
amount > 5000
ORDER BY payment_date;
```

paymentID	guestID	service_chargesID	currencyID	bookingID	payment_date	payment_method	amount
13	7	2	14	13	2023-11-14 01:07:04	Debit Card	
46	16	5	50	46	2023-11-19 01:46:36	Debit Card	17
8	5	4	8	8	2023-11-20 21:09:28	Debit Card	1,99
4	3	4	4	4	2023-11-22 12:10:43	Debit Card	35
14	7	3	15	14	2023-11-26 16:50:06	Apple Pay	17
44	16	3	48	44	2023-12-04 06:44:48	Credit Card	4
6	4	3	6	6	2023-12-06 03:15:41	Paypal	
31	11	1	35	31	2023-12-07 12:39:11	Apple Pay	79
32	11	4	36	32	2023-12-08 01:38:41	Paypal	4
1	2	2	1	1	2023-12-10 03:49:53	Apple Pay	2
20	9	1	22	20	2023-12-10 10:46:53	Alipay	1
3	3	4	3	3	2023-12-12 02:58:24	Paypal	14
27	10	2	30	27	2023-12-13 06:58:13	Debit Card	
9	6	4	9	9	2023-12-13 21:05:35	Paypal	
40	15	3	44	40	2023-12-17 13:03:52	Paypal	
5	4	5	5	5	2023-12-22 23:55:20	Paypal	33
34	12	1	38	34	2023-12-25 07:26:17	Credit Card	4,20
21	9	1	23	21	2023-12-28 04:59:37	Alipay	
41	15	4	45	41	2023-12-31 02:40:48	Debit Card	29
26	10	2	29	26	2024-01-04 08:43:11	Google Pay	129,48

Figure 15

TABLE: currency

```
CREATE TABLE `currency` (
`currencyID` INTEGER(11) NOT NULL
AUTO_INCREMENT,
`name` VARCHAR(50) NOT NULL,
`symbol` VARCHAR NOT NULL,
`exchange_rate` DECIMAL NOT NULL,
PRIMARY KEY (`currencyID`) );
```

Test case:

```
SELECT * FROM currency;
```

Rationale:

A table is needed to store all currencies and their conversion rates.

currencyID	name	symbol	exchange_rate
1	Cardano	ADA	3.8744821230
2	United Arab Emirates Dirham	AED	3.6712605427
3	Afghan Afghani	AFN	78.0975506215
4	Albanian Lek	ALL	100.0681534738
5	Armenian Dram	AMD	384.8825108436
6	NL Antillean Guilder	ANG	1.7859402916
7	Angolan Kwanza	AOA	825.1369828635
8	Argentine Peso	ARS	349.8441351812
9	Australian Dollar	AUD	1.5680003013
10	Avalanche	AVAX	0.0990323139
11	Aruban Florin	AWG	1.7900000000
12	Azerbaijani Manat	AZN	1.7000000000
13	Bosnia-Herzegovina Convertible Mark	BAM	1.8308402762
14	Barbadian Dollar	BBD	2.0000000000
15	Bangladeshi Taka	BDT	109.5624138972
16	Bulgarian Lev	BGN	1.8218602400
17	Bahraini Dinar	BHD	0.3760000000
18	Burundian Franc	BIF	2,830.9409224292
19	Bermudan Dollar	BMD	1.0000000000
20	Binance	BNB	0.0045955926
21	Brunei Dollar	BND	1.3648102530

Figure 16

TABLE: taxes

```
CREATE TABLE `taxes` (
`taxID` INTEGER(11) NOT NULL AUTO_INCREMENT,
`tax_amount` DECIMAL NOT NULL,
`tax_type` VARCHAR(20) NOT NULL,
PRIMARY KEY (`taxID`) );
```

Test case:

```
SELECT * FROM taxes;
```

Rationale:

Airbnb will charge some amount as tax from the customers and also pay for their own taxes to the government body, so a table is needed to store all possible taxes that apply to the Airbnb business.

taxID	tax_amount	tax_type
21	5.00000	Transient Occupancy Tax
22	6.20000	Tourist Development Tax
23	12.40000	Municipal Hotel Tax
24	9.60000	City Lodging Fee
25	2.80000	Occupancy Assessment Tax
26	11.00000	Accommodation Tax
27	18.20000	Lodging Tax
28	16.40000	Resort Tax
29	9.60000	Tourism Improvement District Assessment
30	2.80000	Hospitality Tax
31	17.00000	Room Surcharge
32	9.20000	Tourism Promotion Fee
33	16.40000	Guest Room Assessment
34	4.60000	City Sales Tax
35	9.80000	Local Tourism Levy
36	13.00000	Destination Marketing Fee
37	20.20000	Tourism Infrastructure Fee
38	15.40000	Hotel Occupancy Fee
39	1.60000	Bed Tax
40	17.80000	Local Government Assessment

Figure 17

TABLE: payments_taxes

```
CREATE TABLE `payments_taxes` (
`payments_taxesID` INTEGER(11) NOT NULL
AUTO_INCREMENT,
`paymentID` INTEGER(11) NOT NULL,
`taxID` INTEGER(11) NOT NULL,
PRIMARY KEY (`payments_taxesID`),
CONSTRAINT `fk_paymentid_payments_taxes`
FOREIGN KEY (`paymentID`) REFERENCES
`payments` (`paymentID`),
CONSTRAINT `fk_taxid_payments_taxes`
FOREIGN KEY (`taxID`)
REFERENCES `taxes` (`taxID`) );
```

Test case:

```
SELECT * FROM payments_taxes;
```

Rationale:

A junction table to preserve many-to-many relationship between `payments` and `taxes` tables.

payments_taxesID	paymentID	taxID
1	1	21
2	1	22
3	1	23
4	1	24
5	1	25
6	1	26
7	1	27
8	1	28
9	1	29
10	1	30
11	1	31
12	2	21
13	2	22
14	2	23
15	2	24
16	2	25
17	2	26
18	2	27
19	2	28
20	2	29
21	2	30

Figure 18

TABLE: service_charges

```
CREATE TABLE `service_charges` (
`service_chargesID` INTEGER(5) NOT NULL
AUTO_INCREMENT,
`amount` DECIMAL NOT NULL,
`service_grade` VARCHAR(20) NOT NULL,
PRIMARY KEY (`service_chargesID`) );
```

Test case:

```
SELECT * FROM
service_charges;
```

Rationale:

Airbnb charges customers for the service they provide. Service charges depend on various factors such as region, currency, quality of rental etc. Thus, amount of service, charged is stored, but the actual implementation will depend on the Airbnb business.

service_chargesID	amount	service_grade
1	1.5000	service_grade_1
2	3.4500	service_grade_2
3	10.8900	service_grade_3
4	23.7500	service_grade_4
5	50.2500	service_grade_5
6	9.0000	service_grade_6
7	31.0000	service_grade_7
8	91.0000	service_grade_8
9	46.0000	service_grade_9
10	67.0000	service_grade_10
11	27.0000	service_grade_11
12	95.0000	service_grade_12
13	14.0000	service_grade_13
14	54.0000	service_grade_14
15	80.0000	service_grade_15
16	44.0000	service_grade_16
17	42.0000	service_grade_17
18	22.0000	service_grade_18
19	72.0000	service_grade_19
20	87.0000	service_grade_20

Figure 19

TABLE: coupons

```
CREATE TABLE `coupons` (
`couponID` INTEGER(5) NOT NULL AUTO_INCREMENT,
`bookingID` INTEGER(11) NOT NULL,
`code` VARCHAR(8) NOT NULL,
`discount` DECIMAL NOT NULL,
`creation_date` DATETIME NOT NULL,
`expiry_date` DATETIME NOT NULL,
PRIMARY KEY (`couponID`),
CONSTRAINT `fk_bookingid_coupons`
FOREIGN KEY (`bookingID`)
REFERENCES `booking` (`bookingID`) );
```

Test case:

```
SELECT * FROM coupons;
```

Rationale:

Loyal customers should be rewarded with discount coupons. The database only stores generated coupons and its relevant data with a specific booking. Actual implementation such as generation and distribution of coupons will depend on other systems of Airbnb business.

couponID	bookingID	code	discount	creation_date	expiry_date
1	1	BGSZ2922	20	2023-12-10 02:00:00	2023-12-11 02:00:00
2	4	RSDN8352	20	2023-11-22 06:00:00	2023-11-23 06:00:00
3	5	YKWC9457	30	2023-12-22 03:00:00	2023-12-23 03:00:00
4	6	AHSX7055	30	2023-12-06 04:00:00	2023-12-07 04:00:00
5	7	KOUU3670	30	2024-03-01 02:00:00	2024-03-02 02:00:00
6	8	MLQG0092	30	2023-11-20 05:00:00	2023-11-21 05:00:00
7	9	ZQLS9128	100	2023-12-13 06:00:00	2023-12-14 06:00:00
8	10	PDGA7009	50	2024-01-27 03:00:00	2024-01-28 03:00:00
9	11	PULH3091	100	2024-02-09 02:00:00	2024-02-10 02:00:00
10	12	MJRT3549	30	2024-01-25 01:00:00	2024-01-26 01:00:00
11	13	QUPU6435	30	2023-11-14 03:00:00	2023-11-15 03:00:00
12	14	NLZM3978	20	2023-11-26 06:00:00	2023-11-27 06:00:00
13	15	MLZH1778	30	2023-12-10 05:00:00	2023-12-11 05:00:00
14	16	DQHM4542	50	2023-11-28 04:00:00	2023-11-29 04:00:00
15	17	ZYEJ6323	30	2024-02-18 06:00:00	2024-02-19 06:00:00
16	18	RPRD7406	20	2023-12-06 05:00:00	2023-12-07 05:00:00
17	19	LDER3887	100	2024-01-24 04:00:00	2024-01-25 04:00:00
18	21	NYQG0119	30	2023-12-28 05:00:00	2023-12-29 05:00:00
19	22	LPUD7219	50	2024-01-02 01:00:00	2024-01-03 01:00:00
20	24	DWFB9737	50	2024-01-25 06:00:00	2024-01-26 06:00:00
21	25	IRHL8203	20	2024-01-27 03:00:00	2024-01-28 03:00:00

Figure 20

TABLE: reviews

```
CREATE TABLE `reviews` (
`reviewID` INTEGER(11) NOT NULL
AUTO_INCREMENT,
`listingID` INTEGER(11) NOT NULL,
`guestID` INTEGER(11) NOT NULL,
`timestamp` DATETIME NOT NULL,
`comment` TEXT,
`rating` INTEGER(1),
PRIMARY KEY (`reviewID`),
CONSTRAINT `fk_guestid_reviews`
FOREIGN KEY (`guestID`)
REFERENCES `guest` (`guestID`),
CONSTRAINT `fk_listingid_reviews`
FOREIGN KEY (`listingID`)
REFERENCES `listing` (`listingID`) );
```

Rationale:

Guests can give reviews to a rental listing. Comments, rating starts and timestamps for each review need to be stored in the database.

reviewID	listingID	guestID	timestamp	comment	rating
6	1	5	2024-02-03 15:18:49	Good place.	5
11	1	7	2024-01-05 15:06:49	Best host ever! Highly recommended!	5
22	1	10	2024-02-01 15:44:50	Its Okay.	3
27	1	11	2023-12-24 16:48:02	I loved the place and the host!	5
28	1	11	2024-01-31 18:11:17	Great place which was cozy and comfy.	5
29	1	11	2024-01-02 20:17:47	Good place.	5
36	1	15	2024-01-18 07:04:38	It was a good place, coming back again.	4
12	2	7	2024-01-06 00:19:49	We old timers had a great time together with...	5
37	4	15	2024-01-14 05:53:07	Its Okay.	4
2	5	3	2024-01-16 15:19:52	Lovely Place.	5
14	6	7	2024-01-16 15:52:22	The residence was bright, the night sky view...	5
35	6	15	2024-01-09 03:08:33	Lovely Place.	5
39	6	16	2024-01-29 13:14:38	Thanks the host for great hospitality!	5
13	8	7	2024-01-19 22:17:27	I loved the place and the host!	5
18	8	9	2024-01-26 19:35:38	Thanks the host for great hospitality!	5
1	9	2	2024-02-06 05:10:47	It was a good place, coming back again.	5
10	9	7	2024-02-07 18:34:51	It was a good place, coming back again.	5
25	9	11	2024-01-22 21:36:22	It was Alright.	4
4	10	4	2024-02-11 17:40:06	Lovely Place.	5
17	10	9	2024-01-25 04:45:42	We old timers had a great time together with...	5
16	11	8	2024-01-21 15:31:02	Its Okay.	2

Figure 21

Test case:

```
SELECT * FROM reviews ORDER
BY listingID;
```

TABLE: messages

```
CREATE TABLE `messages` (
`messageID` INTEGER(11) NOT NULL
AUTO_INCREMENT,
`guestID` INTEGER(11) NOT NULL,
`hostID` INTEGER(11) NOT NULL,
`timestamp` DATETIME NOT NULL,
`text` TEXT NOT NULL,
`message_by_host` BOOLEAN NOT NULL,
PRIMARY KEY (`messageID`),
CONSTRAINT `fk_guestid_messages`
FOREIGN KEY (`guestID`)
REFERENCES `guest` (`guestID`),
CONSTRAINT `fk_hostid_messages`
FOREIGN KEY (`hostID`)
REFERENCES `host` (`hostID`) );
```

Test case:

```
SELECT * FROM messages;
```

Rationale:

Chat messages between guests and hosts need to be stored. Since all messages are to be stored in a single table then a special column is used to differ between messages from host or guest. Actual implementation like session management and chat thread differentiation is dependent on other systems deployed by Airbnb.

messageID	guestID	hostID	timestamp	text	message_by_host
6,583	2	1	2023-12-07 21:24:33	Hi, we're settled in, and everything is fantasti...	0
6,584	2	1	2023-12-07 21:27:12	Hi there! I'm so happy to hear that you're c...	1
6,585	2	1	2023-12-07 21:28:49	Thank you! By the way, are there any hiking ...	0
6,586	2	1	2023-12-07 21:31:27	Yes, there's a beautiful nature reserve about...	1
6,587	2	1	2023-12-12 02:38:00	Hello, thank you for all the details about you...	0
6,588	2	1	2023-12-12 06:08:00	Hi! I appreciate your consideration. I unders...	1
6,589	2	2	2023-12-01 09:58:02	Hi, we just got in. The place is lovely, thank y...	0
6,590	2	2	2023-12-01 10:00:39	Hello! I'm thrilled you like it. If there's anythi...	1
6,591	2	2	2023-12-01 10:02:11	Thanks! Is there a public transportation opti...	0
6,592	2	2	2023-12-01 10:05:02	Yes, there's a bus stop just a few blocks aw...	1
6,593	2	2	2023-12-12 03:01:00	Hi, I've been looking at various options and ...	0
6,594	2	2	2023-12-12 07:04:00	Hello! Thank you for letting me know. I'm gl...	1
6,595	3	1	2023-11-15 23:26:14	Hi, we just got in. The place is lovely, thank y...	0
6,596	3	1	2023-11-15 23:28:51	Hello! I'm thrilled you like it. If there's anythi...	1
6,597	3	1	2023-11-15 23:30:23	Thanks! Is there a public transportation opti...	0
6,598	3	1	2023-11-15 23:33:14	Yes, there's a bus stop just a few blocks aw...	1
6,599	3	1	2023-12-26 01:32:00	Hi, I was wondering if you had any recomme...	0
6,600	3	1	2023-12-26 03:01:00	Absolutely! There's a beautiful museum dow...	1
6,601	3	1	2023-12-26 04:49:00	Oh, that sounds great! I've been to a few m...	0
6,602	3	1	2023-12-26 05:52:00	It's been quite pleasant lately, not too hot o...	1
6,603	3	1	2023-12-26 07:00:00	Yeah, weather can really affect plans. I rame...	0

Figure 22

TABLE: language

```
CREATE TABLE `language` (
`languageID` INTEGER(11) NOT NULL
AUTO_INCREMENT,
`code` VARCHAR(5) NOT NULL,
`name` VARCHAR(20) NOT NULL,
PRIMARY KEY (`languageID`) );
```

Test case:

```
SELECT * FROM
`airbnb`.`language`;
```

Rationale:

Users can interact with each other in different languages and also list spoken languages on their profiles. Storing language data can be helpful for Airbnb in analyzing their customer base and producing products for relevant customers.

languageID	code	name
508	aar	Afar
509	abk	Abkhazian
510	ace	Achinese
511	ach	Acoli
512	ada	Adangme
513	ady	Adyghe; Adygei
514	afa	Afro-Asiatic languages
515	afh	Afrihili
516	afr	Afrikaans
517	ain	Ainu
518	aka	Akan
519	akk	Akkadian
520	alb (B)	Albanian
521	ale	Aleut
522	alg	Algonquian languages
523	alt	Southern Altai
524	amh	Amharic
525	ang	English, Old (ca.450-1100)
526	anp	Angika
527	apa	Apache languages
528	ara	Arabic

Figure 23

TABLE: messages_language

```
CREATE TABLE `message_language` (
`ID` INTEGER(11) NOT NULL,
`languageID` INTEGER(11) NOT NULL,
`messageID` INTEGER(11) NOT NULL,
PRIMARY KEY (`ID`),
CONSTRAINT `fk_languageid_messages_languages`
FOREIGN KEY (`languageID`)
REFERENCES `language` (`languageID`),
CONSTRAINT
`fk_messageid_messages_languages`
FOREIGN KEY (`messageID`)
REFERENCES `messages` (`messageID`) );
```

Test case:

```
SELECT * FROM
messages_language;
```

Rationale:

Junction table to represent and store many-to-many relationship between `messages` and `language` tables.

ID	messageID	languageID
1	6,583	636
2	6,584	636
3	6,585	636
4	6,586	636
5	6,587	636
6	6,588	636
7	6,589	636
8	6,590	636
9	6,591	636
10	6,592	636
11	6,593	636
12	6,594	636
13	6,595	636
14	6,596	636
15	6,597	636
16	6,598	636
17	6,599	636
18	6,600	636
19	6,601	636
20	6,602	636
21	6,603	636

Figure 24

TABLE: host_language

```
CREATE TABLE `host_language` (
`ID` INTEGER(11) NOT NULL,
`hostID` INTEGER(11) NOT NULL,
`languageID` INTEGER(11) NOT NULL,
PRIMARY KEY (`ID`),
CONSTRAINT `fk_hostid_host_language` FOREIGN
KEY (`hostID`) REFERENCES `host` (`hostID`),
CONSTRAINT
`fk_languageid_host_language`
FOREIGN KEY (`languageID`) REFERENCES
`language` (`languageID`) );
```

Test case:

```
SELECT * FROM host_language;
```

Rationale:

Similar to `messages_language` table, this table is a junction table between `host` and `language` tables. It can be used to show spoken languages on hosts' profile page.

ID	hostID	languageID
1	1	723
2	2	636
3	3	636
4	4	636
5	5	636
6	6	619
7	7	619
8	8	636
9	9	552
10	10	619
11	11	796
12	12	552
13	13	699
14	14	536
15	15	636
16	16	636
17	17	636
18	18	720
19	19	619
20	20	619

Figure 25

TABLE: ratings_on_guest

```
CREATE TABLE `ratings_on_guest` (
`ratingID` INTEGER(11) NOT NULL
AUTO_INCREMENT,
`guestID` INTEGER(11) NOT NULL,
`rating` INTEGER(1) NOT NULL,
PRIMARY KEY (`ratingID`),
CONSTRAINT `fk_guestid_host_rates_guest`
FOREIGN KEY (`guestID`)
REFERENCES `guest` (`guestID`) );
```

Test case:

```
SELECT * FROM
ratings_on_guest;
```

Rationale:

Hosts can rate a Guest profile which also helps other hosts decide to host a rated guest. This table is used to store ratings from hosts for a guest profile.

ratingID	guestID	rating
1	2	5
2	3	5
3	4	5
4	5	5
5	6	5
6	7	5
7	8	5
8	9	5
9	10	5
10	11	5
11	12	5
12	13	5
13	14	5
14	15	5
15	16	5
16	17	5
17	18	5
18	2	5
19	3	5
20	4	5
21	5	5

Figure 26

TABLE: host_ratings_on_guest

```
CREATE TABLE `host_ratings_on_guest` (
`ID` INTEGER(11) NOT NULL,
`ratingID` INTEGER(11) NOT NULL,
`hostID` INTEGER(11) NOT NULL,
PRIMARY KEY (`ID`),
CONSTRAINT `fk_ratingid_host_ratings_on_guest`
FOREIGN KEY (`ratingID`) REFERENCES
`ratings_on_guest` (`ratingID`),
CONSTRAINT
`fk_hostid_host_ratings_on_guest`
FOREIGN KEY (`hostID`)
REFERENCES `host` (`hostID`) );
```

Rationale:

Junction table between `host` table and `ratings_on_guest` table.

ID	ratingID	hostID
1	1	19
2	18	19
3	35	5
4	52	17
5	69	13
6	86	9
7	103	8
8	120	6
9	137	18
10	154	1
11	171	12
12	188	11
13	205	7
14	222	7
15	239	13
16	2	13
17	19	10
18	36	13
19	53	7
20	70	10
21	87	17

Figure 27

Test case:

```
SELECT * FROM
host_ratings_on_guest;
```

TABLE: ratings_on_host

```
CREATE TABLE `ratings_on_host` (
`ratingID` INTEGER(11) NOT NULL
AUTO_INCREMENT,
`hostID` INTEGER(11) NOT NULL,
`rating` INTEGER(1) NOT NULL,
PRIMARY KEY (`ratingID`),
CONSTRAINT `fk_hostid_guest_rates_host`
FOREIGN KEY (`hostID`)
REFERENCES `host` (`hostID`) );
```

Test case:

```
SELECT * FROM
ratings_on_host;
```

Rationale:

Guests can also rate host profiles. This allows other guests to decide in renting places with highly rated hosts.

ratingID	hostID	rating
1	2	5
2	3	5
3	3	5
4	3	4
5	4	5
6	4	5
7	5	5
8	5	5
9	6	5
10	6	5
11	6	5
12	7	5
13	7	5
14	7	5
15	7	5
16	7	5
17	8	5
18	8	5
19	9	5
20	9	5
21	9	5

Figure 28

TABLE: guest_ratings_on_host

```
CREATE TABLE `guest_ratings_on_host` (
`ID` INTEGER(11) NOT NULL,
`ratingID` INTEGER(11) NOT NULL,
`guestID` INTEGER(11) NOT NULL,
PRIMARY KEY (`ID`),
CONSTRAINT `fk_ratingid_guest_ratings_on_host`
FOREIGN KEY (`ratingID`) REFERENCES
`ratings_on_host` (`ratingID`),
CONSTRAINT
`fk_guestid_guest_ratings_on_host`
FOREIGN KEY (`guestID`)
REFERENCES `guest` (`guestID`) );
```

Test case:

```
SELECT * FROM
guest_ratings_on_host;
```

Rationale:

Junction table between `guest` and `ratings_on_host` tables.

ID	ratingID	guestID
1	1	9
2	2	18
3	3	12
4	4	5
5	5	15
6	6	10
7	7	18
8	8	2
9	9	17
10	10	12
11	11	18
12	12	9
13	13	3
14	14	2
15	15	8
16	16	6
17	17	17
18	18	11
19	19	10
20	20	13
21	21	8

Figure 29

TABLE: user_social_networks

```
CREATE TABLE `user_social_networks` (
`socialID` INTEGER(11) NOT NULL
AUTO_INCREMENT,
`userID` INTEGER(11) NOT NULL,
`social_network_url` TEXT NOT NULL,
`rating_enabled` BOOLEAN NOT NULL,
`created_at` DATETIME NOT NULL,
`updated_at` DATETIME NOT NULL,
PRIMARY KEY (`socialID`),
CONSTRAINT
`fk_userid_user_social_networks`
FOREIGN KEY (`userID`)
REFERENCES `user` (`userID`) );
```

Test case:

```
SELECT * FROM
user_social_networks;
```

Rationale:

Users may link their socials to their profile so the database must store all socials for each user, and, they may update the social links on their profile. Also, some users may disable the feature so a separate column is needed.

socialID	userID	social_network_url	rating_enabled	created_at	updated_at
1	1	www.twitch.com/elenithomas	1	2023-11-13 13:46:32	(NULL)
2	1	www.line.com/elenithomas	1	2023-11-13 13:46:32	(NULL)
3	1	www.linkedin.com/elenithomas	1	2023-11-13 13:46:32	(NULL)
4	1	www.Weibo.com/elenithomas	1	2023-11-13 13:46:32	(NULL)
5	1	www.twitter.com/elenithomas	1	2023-11-13 13:46:32	(NULL)
6	1	www.tiktok.com/elenithomas	1	2023-11-13 13:46:32	(NULL)
7	2	www.facebook.com/dylanhatimi	1	2023-10-20 09:21:22	(NULL)
8	2	www.twitter.com/dylanhatimi	1	2023-10-20 09:21:22	(NULL)
9	2	www.whatsapp.com/dylanhatimi	1	2023-10-20 09:21:22	(NULL)
10	3	www.wechat.com/dylanhatimi	1	2023-09-04 01:48:16	(NULL)
11	3	www.instagram.com/dylanhatimi	1	2023-09-04 01:48:16	(NULL)
12	3	www.telegram.com/dylanhatimi	1	2023-09-04 01:48:16	(NULL)
13	3	www.snapchat.com/dylanhatimi	1	2023-09-04 01:48:16	(NULL)
14	3	www.wechat.com/dylanhatimi	1	2023-09-04 01:48:16	(NULL)
15	3	www.whatsapp.com/dylanhatimi	1	2023-09-04 01:48:16	(NULL)
16	4	www.facebook.com/lenamateo	1	2023-11-30 08:03:56	(NULL)
17	4	www.twitter.com/lenamateo	1	2023-11-30 08:03:56	(NULL)
18	4	www.twitter.com/lenamateo	1	2023-11-30 08:03:56	(NULL)
19	4	www.linkedin.com/lenamateo	1	2023-11-30 08:03:56	(NULL)
20	4	www.snapchat.com/lenamateo	1	2023-11-30 08:03:56	(NULL)
21	4	www.wechat.com/lenamateo	1	2023-11-30 08:03:56	(NULL)

Figure 30

TABLE: wishlist

```
CREATE TABLE `wishlist` (
`wishlistID` INTEGER(11) NOT NULL
AUTO_INCREMENT,
`guestID` INTEGER(11) NOT NULL,
`listingID` INTEGER(11) NOT NULL,
`wishlist_name` VARCHAR(100) NOT NULL,
`listing_info` TEXT,
`url` TEXT NOT NULL,
PRIMARY KEY (`wishlistID`),
CONSTRAINT `fk_guestid_wishlist`
FOREIGN KEY (`guestID`)
REFERENCES `guest` (`guestID`) );
```

Rationale:

Guest users may want to store some listings with them which they would like to visit in the future. A wishlist feature can be added to the website if a corresponding table exists in the database. Users can create multiple wishlists with listings they like and store some notes for themselves. Actual implementation is upto Airbnb.

Test case:

```
SELECT * FROM wishlist;
```

```
1 SELECT * FROM wishlist;
```

wishlist (79r × 6c)

wishlistID	guestID	listingID	wishlist_name	listing_info	url
1	2	17	villages	(NULL)	www.airbnb.com/wishlis
2	2	6	tallest places	(NULL)	www.airbnb.com/wishlis
3	2	16	tallest places	(NULL)	www.airbnb.com/wishlis
4	2	16	villages	(NULL)	www.airbnb.com/wishlis
5	2	3	city places	(NULL)	www.airbnb.com/wishlis
6	2	2	mountains	(NULL)	www.airbnb.com/wishlis
7	3	17	nature	(NULL)	www.airbnb.com/wishlis

Figure 31

INSERT QUERIES

Amenities

```
INSERT INTO `amenities` (`amenitiesID`, `listingID`, `wifi`, `air_conditioning`, `parking`, `food_street`, `shopping_street`, `park`) VALUES
		(1, 1, 1, 1, 1, 0, 0, NULL),
		(2, 2, 1, 1, 0, 0, 0, 0),
		(3, 3, 1, 1, 1, 1, 0, 0),
		(4, 4, 1, 1, 0, 0, 0, 0),
		(5, 5, 1, 1, 0, 0, 0, 0),
		(6, 6, 1, 1, 0, 0, 0, 0),
		(7, 7, 1, 1, 0, 0, 0, 0),
		(8, 8, 1, 1, 0, 0, 0, 0),
		(9, 9, 1, 1, 0, 0, 0, 0),
		(10, 10, 1, 1, 0, 0, 0, 0),
		(11, 11, 1, 1, 0, 0, 0, 0),
		(12, 12, 1, 1, 0, 0, 0, 0),
		(13, 13, 1, 1, 0, 0, 0, 0),
		(14, 14, 1, 1, 0, 0, 0, 0),
		(15, 15, 1, 1, 0, 0, 0, 0),
		(16, 16, 1, 1, 0, 0, 0, 0),
		(17, 17, 1, 1, 0, 0, 0, 0),
		(18, 18, 1, 1, 0, 0, 0, 0),
		(19, 19, 1, 1, 0, 0, 0, 0),
		(20, 20, 1, 1, 0, 0, 0, 0);
```

Availability

```
INSERT INTO `availability` (`availabilityID`, `listingID`, `available_from`, `available_until`) VALUES
        (1, 1, '2023-12-04', '2023-12-14'),
        (2, 2, '2023-12-05', '2023-12-15'),
        (3, 3, '2023-12-08', '2023-12-18'),
        (4, 4, '2023-12-10', '2023-12-20'),
        (5, 5, '2023-12-12', '2023-12-22'),
        (6, 6, '2023-12-15', '2023-12-25'),
        (7, 7, '2023-12-18', '2023-12-28'),
        (8, 8, '2023-12-20', '2023-12-30'),
        (9, 9, '2023-12-23', '2024-01-02'),
        (10, 10, '2023-12-25', '2024-01-04'),
        (11, 11, '2023-12-28', '2024-01-07'),
        (12, 12, '2023-12-30', '2024-01-09'),
        (13, 13, '2024-01-02', '2024-01-12'),
        (14, 14, '2024-01-04', '2024-01-14'),
        (15, 15, '2024-01-07', '2024-01-17'),
        (16, 16, '2024-01-09', '2024-01-19'),
        (17, 17, '2024-01-12', '2024-01-22'),
        (18, 18, '2024-01-14', '2024-01-24'),
        (19, 19, '2024-01-17', '2024-01-27'),
        (20, 20, '2024-01-19', '2024-01-29');
```

Booking

```
INSERT INTO `booking` (`bookingID`, `guestID`, `listingID`, `booking_date`, `booking_from`, `booking_until`, `check_in`, `check_out`, `total_price`, `guest_num`, `status`) VALUES
(1, 2, 9, '2023-12-10 00:00:00', '2024-01-05', '2024-02-04', '2024-01-06 00:19:21', '2024-02-04 10:17:22', 5640, 5, 'booked'),
(2, 3, 18, '2024-01-05 00:00:00', '2024-01-28', '2024-03-13', '2024-01-28 02:19:19', '2024-03-13 04:41:14', 2250, 4, 'cancelled'),
(3, 3, 12, '2023-12-12 00:00:00', '2024-01-13', '2024-02-12', '2024-01-13 23:55:45', '2024-02-12 02:55:02', 1800, 4, 'cancelled'),
(4, 3, 5, '2023-11-22 00:00:00', '2023-12-26', '2024-01-15', '2023-12-26 09:03:55', '2024-01-15 06:40:44', 3560, 4, 'rented'),
(5, 4, 15, '2023-12-22 00:00:00', '2024-01-20', '2024-02-04', '2024-01-20 20:13:19', '2024-02-04 18:39:42', 855, 6, 'rented'),
(6, 4, 10, '2023-12-06 00:00:00', '2024-01-08', '2024-02-07', '2024-01-08 10:07:15', '2024-02-07 19:50:03', 4920, 6, 'booked'),
(7, 5, 18, '2024-03-01 00:00:00', '2024-01-28', '2024-02-27', '2024-01-28 16:04:44', '2024-02-27 22:31:37', 1890, 4, 'rented'),
(8, 5, 1, '2023-11-20 00:00:00', '2023-12-16', '2024-01-30', '2023-12-16 08:30:26', '2024-01-30 14:18:10', 5670, 4, 'rented'),
(9, 6, 17, '2023-12-13 00:00:00', '2024-01-25', '2024-02-24', '2024-01-25 10:24:12', '2024-02-24 17:27:08', 4770, 7, 'rented'),
(10, 6, 12, '2024-01-27 00:00:00', '2024-01-13', '2024-02-02', '2024-01-13 16:33:22', '2024-02-02 15:20:26', 3900, 7, 'booked'),
(11, 6, 18, '2024-02-09 00:00:00', '2024-01-28', '2024-02-17', '2024-01-28 13:17:33', '2024-02-17 22:30:35', 2060, 7, 'booked'),
(12, 7, 9, '2024-01-25 00:00:00', '2024-01-05', '2024-02-04', '2024-01-05 12:51:52', '2024-02-04 03:52:48', 2160, 4, 'booked'),
(13, 7, 1, '2023-11-14 00:00:00', '2023-12-15', '2024-01-04', '2023-12-15 19:13:57', '2024-01-04 01:32:26', 3000, 4, 'rented'),
(14, 7, 2, '2023-11-26 00:00:00', '2023-12-19', '2024-01-03', '2023-12-19 04:43:23', '2024-01-03 03:33:01', 1560, 4, 'rented'),
(15, 7, 8, '2023-12-10 00:00:00', '2024-01-03', '2024-01-18', '2024-01-03 15:13:50', '2024-01-18 06:20:20', 1905, 4, 'rented'),
(16, 7, 6, '2023-11-28 00:00:00', '2023-12-29', '2024-01-13', '2023-12-29 11:08:04', '2024-01-13 05:35:15', 855, 4, 'rented'),
(17, 8, 17, '2024-02-18 00:00:00', '2024-01-25', '2024-02-09', '2024-01-25 08:16:23', '2024-02-09 19:05:09', 1995, 3, 'booked'),
(18, 8, 11, '2023-12-06 00:00:00', '2024-01-10', '2024-01-17', '2024-01-10 02:05:49', '2024-01-17 20:12:04', 511, 3, 'booked'),
(19, 9, 10, '2024-01-24 00:00:00', '2024-01-08', '2024-01-23', '2024-01-08 20:20:32', '2024-01-23 04:10:30', 2565, 3, 'booked'),
(20, 9, 13, '2023-12-10 00:00:00', '2024-01-15', '2024-02-14', '2024-01-15 07:09:32', '2024-02-14 02:31:40', 1650, 3, 'cancelled'),
(21, 9, 8, '2023-12-28 00:00:00', '2024-01-03', '2024-01-23', '2024-01-03 07:30:54', '2024-01-23 11:12:16', 1680, 3, 'rented'),
(22, 9, 16, '2024-01-02 00:00:00', '2024-01-23', '2024-02-22', '2024-01-23 20:07:50', '2024-02-23 00:39:50', 3060, 3, 'rented'),
(23, 9, 12, '2024-01-20 00:00:00', '2024-01-13', '2024-02-27', '2024-01-13 07:06:29', '2024-02-27 20:10:42', 4050, 3, 'missed'),
(24, 10, 18, '2024-01-25 00:00:00', '2024-01-28', '2024-02-17', '2024-01-28 17:56:16', '2024-02-17 21:06:18', 3940, 4, 'rented'),
(25, 10, 17, '2024-01-27 00:00:00', '2024-01-25', '2024-02-09', '2024-01-25 02:37:14', '2024-02-09 19:48:29', 1185, 4, 'booked'),
```

```
(26, 10, 1, '2024-01-04 00:00:00', '2023-12-16', '2024-01-30', '2023-12-16 18:52:52', '2024-01-30 04:24:50', 5130, 4, 'rented'),
(27, 10, 14, '2023-12-13 00:00:00', '2024-01-18', '2024-02-17', '2024-01-18 19:40:35', '2024-02-17 17:36:27', 3840, 4, 'booked'),
(28, 10, 16, '2024-02-07 00:00:00', '2024-01-23', '2024-02-22', '2024-01-23 19:22:21', '2024-02-22 21:41:29', 5220, 4, 'booked'),
(29, 11, 9, '2024-01-08 00:00:00', '2024-01-05', '2024-01-20', '2024-01-05 01:44:29', '2024-01-20 23:22:42', 2835, 7, 'rented'),
(30, 11, 16, '2024-02-03 00:00:00', '2024-01-23', '2024-01-30', '2024-01-23 07:12:37', '2024-01-30 13:41:19', 588, 7, 'rented'),
(31, 11, 1, '2023-12-07 00:00:00', '2023-12-15', '2023-12-22', '2023-12-15 15:12:09', '2023-12-22 21:18:21', 847, 7, 'booked'),
(32, 11, 1, '2023-12-08 00:00:00', '2023-12-15', '2024-01-29', '2023-12-15 11:10:16', '2024-01-29 03:57:18', 6030, 7, 'rented'),
(33, 11, 1, '2024-01-16 00:00:00', '2023-12-16', '2023-12-31', '2023-12-16 11:35:29', '2023-12-31 19:06:24', 1245, 7, 'rented'),
(34, 12, 19, '2023-12-25 00:00:00', '2024-01-30', '2024-03-15', '2024-01-30 16:23:13', '2024-03-15 18:49:06', 7785, 5, 'rented'),
(35, 12, 16, '2024-01-20 00:00:00', '2024-01-23', '2024-02-12', '2024-01-23 23:27:09', '2024-02-12 10:52:01', 1100, 5, 'booked'),
(36, 12, 19, '2024-02-21 00:00:00', '2024-01-30', '2024-02-14', '2024-01-30 01:51:54', '2024-02-14 08:38:56', 1215, 5, 'cancelled'),
(37, 12, 16, '2024-01-12 00:00:00', '2024-01-23', '2024-02-07', '2024-01-23 05:57:50', '2024-02-07 12:04:38', 1545, 5, 'rented'),
(38, 12, 16, '2024-01-09 00:00:00', '2024-01-23', '2024-01-30', '2024-01-23 20:56:14', '2024-01-30 04:52:01', 1064, 5, 'rented'),
(39, 13, 12, '2024-02-12 00:00:00', '2024-01-13', '2024-01-20', '2024-01-13 23:21:59', '2024-01-20 23:05:04', 1330, 6, 'rented'),
(40, 15, 6, '2023-12-17 00:00:00', '2023-12-29', '2024-01-05', '2023-12-29 11:22:56', '2024-01-05 19:12:53', 1022, 5, 'rented'),
(41, 15, 1, '2023-12-31 00:00:00', '2023-12-15', '2024-01-14', '2023-12-15 14:15:05', '2024-01-14 18:24:20', 5160, 5, 'rented'),
(42, 15, 4, '2023-11-29 00:00:00', '2023-12-23', '2024-01-12', '2023-12-23 09:39:06', '2024-01-12 22:38:44', 1320, 5, 'rented'),
(43, 15, 13, '2024-02-08 00:00:00', '2024-01-15', '2024-01-22', '2024-01-15 21:08:09', '2024-01-22 09:05:35', 1295, 5, 'booked'),
(44, 16, 6, '2023-12-04 00:00:00', '2023-12-29', '2024-01-28', '2023-12-29 18:44:27', '2024-01-28 03:16:43', 1500, 3, 'booked'),
(45, 16, 18, '2024-01-28 00:00:00', '2024-01-28', '2024-02-04', '2024-01-28 03:40:54', '2024-02-04 18:08:59', 770, 3, 'rented'),
(46, 16, 3, '2023-11-19 00:00:00', '2023-12-21', '2024-01-10', '2023-12-21 22:14:49', '2024-01-10 22:59:40', 3120, 3, 'missed'),
(47, 16, 18, '2024-01-18 00:00:00', '2024-01-28', '2024-02-27', '2024-01-28 17:06:06', '2024-02-27 08:32:05', 2970, 3, 'rented'),
(48, 16, 19, '2024-01-24 00:00:00', '2024-01-30', '2024-02-14', '2024-01-30 22:49:01', '2024-02-14 07:12:52', 2055, 3, 'rented'),
(49, 17, 14, '2024-02-01 00:00:00', '2024-01-18', '2024-01-25', '2024-01-18 05:05:42', '2024-01-25 10:54:43', 637, 5, 'booked');
```

Coupons

```
INSERT INTO `coupons` (`couponID`, `bookingID`, `code`, `discount`, `creation_date`, `expiry_date`) VALUES
(1, 1, 'BGSZ2922', 20, '2023-12-10 02:00:00', '2023-12-11 02:00:00'),
(2, 4, 'RSDN8352', 20, '2023-11-22 06:00:00', '2023-11-23 06:00:00'),
(3, 5, 'YKWC9457', 30, '2023-12-22 03:00:00', '2023-12-23 03:00:00'),
(4, 6, 'AHSX7055', 30, '2023-12-06 04:00:00', '2023-12-07 04:00:00'),
(5, 7, 'KOUU3670', 30, '2024-03-01 02:00:00', '2024-03-02 02:00:00'),
(6, 8, 'MLQG0092', 30, '2023-11-20 05:00:00', '2023-11-21 05:00:00'),
(7, 9, 'ZQLS9128', 100, '2023-12-13 06:00:00', '2023-12-14 06:00:00'),
(8, 10, 'PDGA7009', 50, '2024-01-27 03:00:00', '2024-01-28 03:00:00'),
(9, 11, 'PULH3091', 100, '2024-02-09 02:00:00', '2024-02-10 02:00:00'),
(10, 12, 'MJRT3549', 30, '2024-01-25 01:00:00', '2024-01-26 01:00:00'),
(11, 13, 'QUPU6435', 30, '2023-11-14 03:00:00', '2023-11-15 03:00:00'),
(12, 14, 'NLZM3978', 20, '2023-11-26 06:00:00', '2023-11-27 06:00:00'),
(13, 15, 'MLZH1778', 30, '2023-12-10 05:00:00', '2023-12-11 05:00:00'),
(14, 16, 'DQHM4542', 50, '2023-11-28 04:00:00', '2023-11-29 04:00:00'),
(15, 17, 'ZYEJ6323', 30, '2024-02-18 06:00:00', '2024-02-19 06:00:00'),
(16, 18, 'RPRD7406', 20, '2023-12-06 05:00:00', '2023-12-07 05:00:00'),
(17, 19, 'LDER3887', 100, '2024-01-24 04:00:00', '2024-01-25 04:00:00'),
(18, 21, 'NYQG0119', 30, '2023-12-28 05:00:00', '2023-12-29 05:00:00'),
(19, 22, 'LPUD7219', 50, '2024-01-02 01:00:00', '2024-01-03 01:00:00'),
(20, 24, 'DWFB9737', 50, '2024-01-25 06:00:00', '2024-01-26 06:00:00');
```

Currency

```
INSERT INTO `currency` (`currencyID`, `name`, `symbol`, `exchange_rate`) VALUES
        (1, 'Cardano', 'ADA', 3.8744821230),
        (2, 'United Arab Emirates Dirham', 'AED', 3.6712605427),
        (3, 'Afghan Afghani', 'AFN', 78.0975506215),
        (4, 'Albanian Lek', 'ALL', 100.0681534738),
        (5, 'Armenian Dram', 'AMD', 384.8825108436),
        (6, 'NL Antillean Guilder', 'ANG', 1.7859402916),
        (7, 'Angolan Kwanza', 'AOA', 825.1369828635),
        (8, 'Argentine Peso', 'ARS', 349.8441351812),
        (9, 'Australian Dollar', 'AUD', 1.5680003013),
        (10, 'Avalanche', 'AVAX', 0.0990323139),
        (11, 'Aruban Florin', 'AWG', 1.7900000000),
        (12, 'Azerbaijani Manat', 'AZN', 1.7000000000),
        (13, 'Bosnia-Herzegovina Convertible Mark', 'BAM', 1.8308402762),
        (14, 'Barbadian Dollar', 'BBD', 2.0000000000),
        (15, 'Bangladeshi Taka', 'BDT', 109.5624138972),
        (16, 'Bulgarian Lev', 'BGN', 1.8218602400),
        (17, 'Bahraini Dinar', 'BHD', 0.3760000000),
        (18, 'Burundian Franc', 'BIF', 2830.9409224292),
        (19, 'Bermudan Dollar', 'BMD', 1.0000000000),
        (20, 'Binance', 'BNB', 0.0045955926);
```

Guest

```
INSERT INTO `guest` (`guestID`, `userID`, `passport_number`, `legal_address`, `id_document`) VALUES
		(2, 21, 'HE0VQDJJ5', 'Suite 646 33510 Rogahn Field, Abshireton, AZ 01208-9163', '29102588967962'),
		(3, 22, 'LZUHH6YGS', 'Parkstr. 21b, Jordanberg, BE 05624', '55348262616537'),
		(4, 23, 'BCE2DYBCV', 'Huerta Rosario 92, Elche, Can 61862', '57681088232861'),
		(5, 24, 'D8CPI63N3', 'Solar Ana Luisa, 9, Telde, Leo 01864', '95261828871503'),
		(6, 25, 'Y0D84WAZZ', 'Esc. 551 Extrarradio Antonio Candelaria 7 Puerta 198, Elche, Ast 25011',
'77647104758315'),
		(7, 26, 'FKC6XIG6N', 'Apt. 771 Jl. Rasuna Said No. 70, Palopo, GO 33672', '77689884600535'),
		(8, 27, 'ODEAEGDXE', 'Via Renato 29, Appartamento 76, Rosalba lido, IS 70745', '29265276991812'),
		(9, 28, 'L9HQX2VHR', 'Incrocio Ross 67, Ricci salentino, RA 47723', '54876332406756'),
		(10, 29, 'BE3KVPP6Q', '1 hoog Tijmesweg 579 III, Oost Sofieberg, OK 4773 HS', '71701764798962'),
		(11, 30, 'HE3HE2MEW', 'Brüder-Bonhoeffer-Str. 7, Corvinscheid, HH 61977', '83253877177886'),
		(12, 31, 'EXODXEV5I', 'Im Kreuzbruch 98c, Neu Kenny, ST 84686', '26795435038778'),
		(13, 32, 'RIUK39IUN', 'Löchergraben 39a, Bad Markhagen, BE 47166', '34622302754095'),
		(14, 33, 'QM4PHUNMV', 'Zimmer 148 Hannah-Höch-Str. 84, Hoffmannfeld, HE 59483', '17691310037845'),
		(15, 34, 'VKQ0OZ1OI', 'al. Frątczak 80283, Działdowo, DŚ 19-158', '12942865981513'),
		(16, 35, '5QLC7SIKP', 'ul. Szeląg 705, Łuków, ŁD 26-836', '52286231787854'),
		(17, 36, '8XNLC4XFQ', 'Apt. 499 830 Ritchie Pass, South Augustaburgh, TN 70589-4086', '70233886625005'),
		(18, 37, 'QDGP7QT3V', 'Suite 327 996 Thomas Summit, East Harrymouth, GA 84428', '74408508387985'),
		(19, 38, 'WOAYK0UB1', '7507 Impasse Du Moulin, 66596 Athis-Mons', '14027842250455'),
		(20, 39, '96H48OI2Q', 'Apt. 890 724 Sherly Islands, Edwardoberg, NC 21618-2395', '66347792998084'),
		(21, 40, 'Z16SIA51S', 'Chun Dong Lu 508hao, City Area - Minxing District, Shanghai', '34085627645616');
```

Guest ratings on host

```
INSERT INTO `guest_ratings_on_host` (`ID`, `ratingID`, `guestID`) VALUES
        (1, 1, 9), (2, 2, 18),(3, 3, 12),(4, 4, 5),(5, 5, 15),(6, 6, 10),(7, 7, 18),(8, 8, 2),(9, 9, 17),
        (10, 10, 12),     (11, 11, 18),     (12, 12, 9),      (13, 13, 3),      (14, 14, 2),
        (15, 15, 8),      (16, 16, 6),      (17, 17, 17),     (18, 18, 11),     (19, 19, 10),
        (20, 20, 13),     (21, 21, 8),      (22, 22, 16),     (23, 23, 12),     (24, 24, 18),
        (25, 25, 17),     (26, 26, 4),      (27, 27, 14),     (28, 28, 16),     (29, 29, 9),
        (30, 30, 16),     (31, 31, 6),      (32, 32, 18),     (33, 33, 12),     (34, 34, 19),
        (35, 35, 16),     (36, 36, 19),     (37, 37, 16),     (38, 38, 16),     (39, 39, 12),
        (40, 40, 6),      (41, 41, 10),     (42, 42, 4),      (43, 43, 13),     (44, 44, 6),
        (45, 45, 18),     (46, 46, 3),      (47, 47, 18),     (48, 48, 19),     (49, 49, 14);
```

Host

```
INSERT INTO `host` (`hostID`, `userID`, `address`) VALUES
        (1, 1, '323-1274, Toyogaoka, Tsukigata-cho Kabato-gun, Hokkaido, Japan\r\n='),
        (2, 2, 'Suite 762 2303 Ariel Prairie, Schummbury, TN 23952, United States'),
        (3, 3, 'Suite 245 4255 Cummings Turnpike, Julianside, WV 89630-2582, United States'),
        (4, 4, 'Apt. 337 Theodor-Heuss-Ring 92b, West Alia, TH 94696, Germany'),
        (5, 5, 'Wiembachallee 7, Berndberg, BE 72528, Germany'),
        (6, 6, 'Bloque Manuel, 4 Puerta 876, Santiago de Compostela, Ara 15476, Mexico'),
        (7, 7, 'Jl. Kartini No. 85, Bantul, SS 24756, Indonesia'),
        (8, 8, 'Apt. 670 Jl. Rasuna Said No. 99, Tanjung Jabung Timur, MA 33528, Indonesia'),
        (9, 9, 'Rambla Luis Miguel, 9, Bilbao, Rio 64215, Argentina'),
        (10, 10, 'Apt. 518 415 Shemeka Garden, Effertzhaven, FL 10443-6644, United States'),
        (11, 11, 'Via Gallo 836, Quarto Olimpia, PE 42018, Italy'),
        (12, 12, '1 hoog Kalikaplantsoen 595, Oud Osgeest, OR 2003 NE'),
        (13, 13, '6729 Owen Mount, Theoton, IA 84715, Australia'),
        (14, 14, '55936 Marvin Plains, Fisherhaven, HI 77329-5140'),
        (15, 15, 'Suite 670 6742 Koelpin Locks, North Denismouth, NJ 76133'),
        (16, 16, '608 Isaias Forks, Miaview, FL 49802'),
        (17, 17, 'Apt. 992 Pfarrer-Klein-Str. 67b, Lianstadt, TH 17206'),
        (18, 18, 'Zimmer 258 Mühlenweg 90b, Bruhnsstadt, SN 42098'),
        (19, 19, 'Apt. 129 Weidenstr. 8, Tischlerberg, HH 86694'),
        (20, 20, 'Quarzstr. 2, Schön Carolinscheid, NI 36707');
```

Host language

```
INSERT INTO `host_language` (`ID`, `hostID`, `languageID`) VALUES
        (1, 1, 723),        (2, 2, 636),        (3, 3, 636),        (4, 4, 636),        (5, 5, 636),
        (6, 6, 619),        (7, 7, 619),        (8, 8, 636),        (9, 9, 552),        (10, 10, 19),
        (11, 11, 796),      (12, 12, 552),      (13, 13, 699),      (14, 14, 536),      (15, 15, 636),
        (16, 16, 636),      (17, 17, 636),      (18, 18, 720),      (19, 19, 619),      (20, 20, 619);
```

Host ratings on guest

```
INSERT INTO `host_ratings_on_guest` (`ID`, `ratingID`, `hostID`) VALUES
        (1, 1, 19),         (2, 18, 19),        (3, 35, 5),         (4, 52, 17),        (5, 69, 13),
        (6, 86, 9),         (7, 103, 8),        (8, 120, 6),        (9, 137, 18),       (10, 154, 1),
        (11, 171, 12),      (12, 188, 11),      (13, 205, 7),       (14, 222, 7),       (15, 239, 13),
        (16, 2, 13),        (17, 19, 10),       (18, 36, 13),       (19, 53, 7),        (20, 70, 10);
```

Images

```
INSERT INTO `images` (`imageID`, `image`, `url`, `file_location`) VALUES
		(54, None, 'www.unsplash.com/99-films-48mTwDzizqE', 'E:images99-films-48mTwDzizqE-unsplash.jpg'),
		(55, None, 'www.unsplash.com/alexandra-gorn-JIUjvqe2ZHg', 'E:imagesalexandra-gorn-JIUjvqe2ZHg-unsplash.jpg'),
		(56, None, 'www.unsplash.com/amira-aboalnaga-O7WjrXiKy_s', 'E:imagesamira-aboalnaga-O7WjrXiKy_s-unsplash.jpg'),
		(57, None, 'www.unsplash.com/andrea-davis-qZTgRKioXcE', 'E:imagesandrea-davis-qZTgRKioXcE-unsplash.jpg'),
		(58, None, 'www.unsplash.com/andy-vult-zwZpdhoTbU0', 'E:imagesandy-vult-zwZpdhoTbU0-unsplash.jpg'),
		(59, None, 'www.unsplash.com/ashley-byrd-yzkTCP4uc9E', 'E:imagesashley-byrd-yzkTCP4uc9E-unsplash.jpg'),
		(60, None, 'www.unsplash.com/barthelemy-de-mazenod-r_zKg2rgc5g', 'E:imagesarthelemy-de-mazenod-r_zKg2rgc5g-unsplash.jpg'),
		(61, None, 'www.unsplash.com/bill-mackie-hK-ZADiFGvk', 'E:imagesill-mackie-hK-ZADiFGvk-unsplash.jpg'),
		(62, None, 'www.unsplash.com/chastity-cortijo-R-w5Q-4Mqm0', 'E:imageschastity-cortijo-R-w5Q-4Mqm0-unsplash.jpg'),
		(63, None, 'www.unsplash.com/christian-koch-D_4R9CcYZOk', 'E:imageschristian-koch-D_4R9CcYZOk-unsplash.jpg'),
		(64, None, 'www.unsplash.com/christopher-jolly-GqbU78bdJFM', 'E:imageschristopher-jolly-GqbU78bdJFM-unsplash.jpg'),
		(65, None, 'www.unsplash.com/collov-home-design-H-1j_s0dhCw', 'E:imagescollov-home-design-H-1j_s0dhCw-unsplash.jpg'),
		(66, None, 'www.unsplash.com/curology-ycEKahEaO5U', 'E:imagescurology-ycEKahEaO5U-unsplash.jpg'),
		(67, None, 'www.unsplash.com/deborah-cortelazzi-gREquCUXQLI', 'E:imagesdeborah-cortelazzi-gREquCUXQLI-unsplash.jpg'),
		(68, None, 'www.unsplash.com/digital-marketing-agency-ntwrk-g39p1kDjvSY', 'E:imagesdigital-marketing-agency-ntwrk-g39p1kDjvSY-nsplash.jpg'),
		(69, None, 'www.unsplash.com/dillon-kydd-XGvwt544g8k', 'E:imagesdillon-kydd-XGvwt544g8k-unsplash.jpg'),
		(70, None, 'www.unsplash.com/drew-coffman-jUOaONoXJQk', 'E:imagesdrew-coffman-jUOaONoXJQk-unsplash.jpg'),
		(71, None, 'www.unsplash.com/eduardo-freire-1UH-uVzTiDU', 'E:imageseduardo-freire-1UH-uVzTiDU-unsplash.jpg'),
		(72, None, 'www.unsplash.com/erik-mclean-rFovKJV0llw', 'E:imageserik-mclean-rFovKJV0llw-unsplash.jpg'),
		(73, None, 'www.unsplash.com/frames-for-your-heart-2d4lAQAlbDA', 'E:imagesframes-for-your-heart-2d4lAQAlbDA-unsplash.jpg'),
		(74, None, 'www.unsplash.com/frames-for-your-heart-mR1CIDduGLc', 'E:imagesframes-for-your-heart-mR1CIDduGLc-unsplash.jpg'),
		(75, None, 'www.unsplash.com/francesca-tosolini-hCU4fimRW-c', 'E:imagesfrancesca-tosolini-hCU4fimRW-c-unsplash.jpg');
```

Language

```
INSERT INTO `language` (`languageID`, `code`, `name`) VALUES
        (508, 'aar', 'Afar'),
        (509, 'abk', 'Abkhazian'),
        (510, 'ace', 'Achinese'),
        (511, 'ach', 'Acoli'),
        (512, 'ada', 'Adangme'),
        (513, 'ady', 'Adyghe; Adygei'),
        (514, 'afa', 'Afro-Asiatic languages'),
        (515, 'afh', 'Afrihili'),
        (516, 'afr', 'Afrikaans'),
        (517, 'ain', 'Ainu'),
        (518, 'aka', 'Akan'),
        (519, 'akk', 'Akkadian'),
        (520, 'alb (B)', 'Albanian'),
        (521, 'ale', 'Aleut'),
        (522, 'alg', 'Algonquian languages'),
        (523, 'alt', 'Southern Altai'),
        (524, 'amh', 'Amharic'),
        (525, 'ang', 'English, Old (ca.450-1100)'),
        (526, 'anp', 'Angika'),
        (527, 'apa', 'Apache languages');
```

Listing

INSERT INTO `listing` (`listingID`, `hostID`, `listing_addressID`, `description`, `bedrooms`, `bathrooms`, `price`) VALUES
(1, 1, 1, 'This charming studio apartment is nestled in the heart of the city. With exposed brick walls and large windows, it offers an abundance of natural light. Perfect for a single professional or a couple seeking a convenient urban lifestyle.', 4, 3, 250.00000),
(2, 2, 2, 'A four-bedroom house in a family-friendly neighborhood. It boasts a large backyard, perfect for kids and pets. The open-concept kitchen and living area make it ideal for entertaining.', 4, 2, 100.00000),
(3, 3, 3, 'This sleek loft apartment offers breathtaking views of the city skyline. With high ceilings, contemporary furnishings, and a rooftop terrace, it\'s an ideal space for young professionals looking for a trendy urban living experience.', 2, 1, 150.00000),
(4, 4, 4, 'A picturesque cottage surrounded by lush greenery and rolling hills. This two-bedroom retreat exudes rustic charm, featuring a stone fireplace and a wraparound porch, perfect for enjoying serene sunsets.', 2, 2, 120.00000),
(5, 5, 5, 'A picturesque cottage surrounded by lush greenery and rolling hills. This two-bedroom retreat exudes rustic charm, featuring a stone fireplace and a wraparound porch, perfect for enjoying serene sunsets.', 3, 3, 220.00000),
(6, 6, 6, 'A cozy bungalow just steps away from the beach. With panoramic ocean views, a private deck, and beach access, this two-bedroom retreat provides a serene escape from the hustle and bustle of city life.', 5, 7, 300.00000),
(7, 7, 7, 'This elegant townhouse, steeped in history, features original hardwood floors, antique fixtures, and a beautifully landscaped garden. Perfect for someone who appreciates classic architecture and timeless design.', 2, 2, 170.00000),
(8, 8, 8, 'A stylish duplex with a modern aesthetic. This property offers a flexible layout, high-end finishes, and a private courtyard, providing an ideal space for both living and working from home.', 3, 3, 250.00000),
(9, 9, 9, ' Located in a historic district, this meticulously restored Victorian flat boasts intricate detailing, stained glass windows, and a cozy fireplace. With three bedrooms and a spacious living area, it\'s perfect for a family.', 4, 5, 195.00000),
(10, 10, 10, 'A unique eco-friendly home surrounded by nature. This energy-efficient property features solar panels, rainwater harvesting, and a serene wooded setting, offering a peaceful and environmentally conscious lifestyle.', 3, 3, 160.00000), (11, 11, 11, 'A luxurious penthouse with floor-to-ceiling windows offering stunning skyline views. This spacious three-bedroom apartment features modern furnishings, a gourmet kitchen, and a private rooftop terrace, perfect for hosting gatherings.', 5, 8, 350.00000),
(12, 12, 12, ' Set on acres of farmland, this renovated farmhouse exudes country charm. With four bedrooms, a wraparound porch, and a barn-turned-entertainment space, it\'s an ideal retreat for those seeking tranquility.', 4, 5, 300.00000),
(13, 13, 13, 'A contemporary townhome in a vibrant neighborhood. This three-story, two-bedroom residence boasts sleek design elements, a rooftop deck, and proximity to local cafes, making it perfect for city living.', 2, 4, 280.00000),
(14, 14, 14, 'A serene lakeside property with panoramic views and a private dock. This three-bedroom house features a sunroom, a fireplace, and ample outdoor space, offering an idyllic getaway from city life.', 3, 3, 240.00000),
(15, 15, 15, 'Located in the heart of the arts district, this loft apartment features exposed beams, industrial accents, and ample natural light. Perfect for creatives, it offers a versatile space for both living and working.', 7, 5, 400.00000),
(16, 16, 16, 'A five-bedroom house in a quiet suburban neighborhood. With a spacious backyard, a play area, and a home office, it\'s tailored for families seeking a comfortable and functional living space.', 5, 4, 230.00000),
(17, 17, 17, 'A classic brownstone with historic charm and modern upgrades. This four-bedroom home features a chef\'s kitchen, a private garden, and proximity to cultural attractions, appealing to those seeking a blend of elegance and convenience.', 4, 4, 245.00000),
(18, 18, 18, 'A studio apartment in an iconic Art Deco building. With vintage details, a cozy layout, and proximity to trendy shops and restaurants, it\'s perfect for a single person seeking character in the heart of the city.', 1, 1, 140.00000),
(19, 19, 19, 'A charming chalet nestled in the mountains, offering breathtaking views. With a stone fireplace, a hot tub, and hiking trails nearby, this two-bedroom retreat is an ideal escape for nature enthusiasts.', 2, 2, 320.00000),
(20, 20, 20, 'A sleek condo overlooking the river, featuring a modern design, high-end appliances, and a balcony for enjoying the tranquil water views. Perfect for professionals seeking a stylish urban retreat.', 1, 1, 150.00000);

Listing Address

```
INSERT INTO `listing_address` (`listing_addressID`, `country`, `state`, `province`, `city`, `sector`, `postcode`, `zipcode`, `area`, `street`, `house_number`, `longitude`, `latitude`) VALUES
		(1, 'Japan', NULL, 'Hokkaido', 'Tsukigata-cho Kabato-gun', NULL, NULL, NULL, NULL, 'Toyogaoka', '323-1274', -56.234897, 23.623485),
		(2, 'USA', 'Nebraska', 'Osgeest', 'Oud', 'Kalikaplantsoen ', NULL, NULL, 'Hoog', '1 hoog', '595', 23.453420, 92.478952),
		(3, 'USA', 'FLorida', 'Miaview', 'Miaview', NULL, NULL, NULL, NULL, 'Isaias Forks', '608', 24.856445, 78.523052),
		(4, 'Australia', 'Iowa', 'Throton', 'Owen Mount', NULL, '84715', NULL, NULL, NULL, '6729', -9.734068, 47.993402),
		(5, 'USA', 'Hawaii', 'Hawaii', 'Fisherhaven', NULL, '77329', 5140, 'Marvin Plains', NULL, '55963', 26.875602, -17.634900),
		(6, 'Germany', 'Hamburg', 'Tischlerberg', 'Tischlerberg', 'Weidenstr. 8', '86694', NULL, NULL, NULL, '129', 48.465306, 43.608366),
		(7, 'Germany', 'Thuringia', 'West Alia', 'Alia', NULL, NULL, NULL, NULL, 'Theodor-Heuss-Ring 92b', '337', 48.992605, 31.747056),
		(8, 'USA', 'Florida', 'Effertzhaven', 'Shemeka', NULL, '10443', 6644, 'Shemeka Garden', '415', '518', 27.935625, 88.234405),
		(9, 'Indonesia', 'Mahkamah Agung', 'Tanjung Jabung Timur', 'Tanjung Jabung Timur', NULL, '33528', NULL, NULL, 'Rasuna Said No. 99', '670', -75.346013, 54.843304),
		(10, 'Germany', 'Thuringia', 'Lianstadt', 'Lianstadt', NULL, '17206', NULL, NULL, 'Pfarrer-Klein-Str. 67b', '992', 48.937654, 36.562073),
		(11, 'Mexico', '', 'Ara', 'Santiago de Compostela', 'Puerta', '15476', NULL, 'Bloque Manuel', NULL, '4', 10.640140, -10.062465),
		(12, 'Indonesia', NULL, 'Bantul', 'Kartini', NULL, '24756', NULL, 'Ji', NULL, '85', -77.471054, 63.261107),
		(13, 'Hungary', 'Nidiea', 'Carolinscheid', 'Schon Carolinscheid', NULL, NULL, NULL, 'Quarzstar', NULL, '2', 52.052184, 28.642509),
		(14, 'Argentina', NULL, 'Rio', 'Bilbao', '9', '64215', NULL, 'Rambla Luis Miguel', NULL, '11', 64.915329, 69.406456),
		(15, 'USA', 'Washington', 'West', 'Julianside', 'Cummings Turnpike', '89630', 4255, 'Suite 245', '', '25', 31.972350, 82.110452),
		(16, 'USA', 'New York', 'North Denismouth', 'Locks', NULL, '76133', 6742, 'Suite 670', NULL, '6', 25.023459, 73.062320),
		(17, 'USA', 'Texas', 'Schummbery', 'Ariel Prairie', NULL, '29352', 2303, 'Suite 762', NULL, '53', 36.925405, 88.092420),
		(18, 'Italy', NULL, 'Peserie', 'Olimipa', NULL, '42018', NULL, 'Quatro', 'Via Gallo', '836', -13.043623, -47.097935),
		(19, 'Germany', NULL, 'Brezdtene', 'Berndberg', NULL, '79425', NULL, NULL, 'Wiembachallee', '7', 54.024421, 49.935012),
		(20, 'Germany', NULL, 'Silldazstadt', 'Bruhnsstadt', NULL, '42040', NULL, 'Muhlenweg', 'Zimmer', '258', 48.194250, 42.523661);
```

Listing Images

```
INSERT INTO `listing_images` (`ID`, `listingID`, `imageID`) VALUES
        (1, 1, 94),     (2, 1, 74),     (3, 1, 68),     (4, 2, 120),    (5, 2, 110),
        (6, 2, 90),     (7, 2, 100),    (8, 3, 85),     (9, 3, 70),     (10, 3, 115),
        (11, 4, 62),    (12, 4, 117),   (13, 4, 75),    (14, 5, 82),    (15, 5, 77),
        (16, 5, 89),    (17, 6, 116),   (18, 6, 55),    (19, 6, 57),    (20, 6, 83),
        (21, 6, 115),   (22, 7, 84),    (23, 7, 90),    (24, 7, 77),    (25, 7, 60),
        (26, 8, 99),    (27, 8, 78),    (28, 8, 87),    (29, 8, 120),   (30, 8, 55),
        (31, 9, 120),   (32, 9, 54),    (33, 9, 80),    (34, 10, 99),   (35, 10, 58),
        (36, 10, 110),  (37, 11, 55),   (38, 11, 121),  (39, 11, 92),   (40, 11, 58),
        (41, 12, 57),   (42, 12, 67),   (43, 12, 101),  (44, 12, 82),   (45, 12, 108),
        (46, 13, 76),   (47, 13, 96),   (48, 13, 57),   (49, 13, 107),  (50, 13, 109),
        (51, 14, 88),   (52, 14, 103),  (53, 14, 97),   (54, 15, 81),   (55, 15, 79),
        (56, 15, 117),  (57, 15, 81),   (58, 16, 118),  (59, 16, 88),   (60, 16, 103),
        (61, 16, 98),   (62, 16, 61);
```

Messages

INSERT INTO `messages` (`messageID`, `guestID`, `hostID`, `timestamp`, `text`, `message_by_host`) VALUES
(6583, 2, 1, '2023-12-07 21:24:33', 'Hi, we\'re settled in, and everything is fantastic. Your place is stunning!', 0),
(6584, 2, 1, '2023-12-07 21:27:12', ' Hi there! I\'m so happy to hear that you\'re comfortable. If you need anything or want suggestions for nearby attractions or restaurants, just let me know.', 1),
(6585, 2, 1, '2023-12-07 21:28:49', 'Thank you! By the way, are there any hiking trails or parks nearby that we should explore?', 0),
(6586, 2, 1, '2023-12-07 21:31:27', ' Yes, there\'s a beautiful nature reserve about a 15-minute drive away with hiking trails and breathtaking views. I\'ll send you more information about it.', 1),
(6587, 2, 1, '2023-12-12 02:38:00', 'Hello, thank you for all the details about your rental. It\'s a beautiful place, but I\'ve had a change in my travel plans and will be staying elsewhere.', 0),
(6588, 2, 1, '2023-12-12 06:08:00', ' Hi! I appreciate your consideration. I understand how plans can change. If you ever find yourself back in this area or if your plans shift again, feel free to check if my place is available. Safe travels!', 1),
(6589, 2, 2, '2023-12-01 09:58:02', 'Hi, we just got in. The place is lovely, thank you!', 0),
(6590, 2, 2, '2023-12-01 10:00:39', ' Hello! I\'m thrilled you like it. If there\'s anything I can assist you with during your stay or if you need any local recommendations, feel free to ask.', 1),
(6591, 2, 2, '2023-12-01 10:02:11', 'Thanks! Is there a public transportation option nearby to get around the city easily?', 0),
(6592, 2, 2, '2023-12-01 10:05:02', ' Yes, there\'s a bus stop just a few blocks away that can take you to downtown in about 20 minutes. I can provide you with a schedule and some tips for getting around.', 1),
(6593, 2, 2, '2023-12-12 03:01:00', 'Hi, I\'ve been looking at various options and while your place is wonderful, I\'ve decided to go with a different style of accommodation that suits my preferences a bit better.', 0),
(6594, 2, 2, '2023-12-12 07:04:00', ' Hello! Thank you for letting me know. I\'m glad you found something that fits your preferences. If your plans ever bring you back this way or if you have any friends visiting, feel free to recommend my
ace. Wishing you a fantastic stay!', 1),
(6595, 3, 1, '2023-11-15 23:26:14', 'Hi, we just got in. The place is lovely, thank you!', 0),
(6596, 3, 1, '2023-11-15 23:28:51', ' Hello! I\'m thrilled you like it. If there\'s anything I can assist you with during your stay or if you need any local recommendations, feel free to ask.', 1),
(6597, 3, 1, '2023-11-15 23:30:23', 'Thanks! Is there a public transportation option nearby to get around the city easily?', 0),
(6598, 3, 1, '2023-11-15 23:33:14', ' Yes, there\'s a bus stop just a few blocks away that can take you to downtown in about 20 minutes. I can provide you with a schedule and some tips for getting around.', 1),
(6599, 3, 1, '2023-12-26 01:32:00', 'Hi, I was wondering if you had any recommendations for local attractions in the area.', 0),
(6600, 3, 1, '2023-12-26 03:01:00', ' Absolutely! There\'s a beautiful museum downtown and a stunning hiking trail nearby.', 1),
(6601, 3, 1, '2023-12-26 04:49:00', 'Oh, that sounds great! I\'ve been to a few museums lately, though. How\'s the weather been around here?', 0),
(6602, 3, 1, '2023-12-26 05:52:00', ' It\'s been quite pleasant lately, not too hot or too cold.', 1),
(6603, 3, 1, '2023-12-26 07:00:00', 'Yeah, weather can really affect plans. I remember one time...', 0),
(6604, 3, 2, '2023-11-18 14:05:10', 'Hi, I hope you\'re doing well! We just checked in, and the place looks amazing. Thank you for the detailed instructions.', 0);

Messages language

```
INSERT INTO `messages_language` (`ID`, `messageID`, `languageID`) VALUES
        (1, 6583, 636),    (2, 6584, 636),    (3, 6585, 636),    (4, 6586, 636),    (5, 6587, 636),
        (6, 6588, 636),    (7, 6589, 636),    (8, 6590, 636),    (9, 6591, 636),    (10, 6592, 636),
        (11, 6593, 636),   (12, 6594, 636),   (13, 6595, 636),   (14, 6596, 636),   (15, 6597, 636),
        (16, 6598, 636),   (17, 6599, 636),   (18, 6600, 636),   (19, 6601, 636),   (20, 6602, 636)
        (21, 6603, 636),   (22, 6604, 636),   (23, 6605, 636),   (24, 6606, 636),   (25, 6607, 636),
        (26, 6608, 636),   (27, 6609, 636),   (28, 6610, 636),   (29, 6611, 636),   (30, 6612, 636),
        (31, 6613, 636),   (32, 6614, 636),   (33, 6615, 636),   (34, 6616, 636),   (35, 6617, 636),
        (36, 6618, 636),   (37, 6619, 636),   (38, 6620, 636),   (39, 6621, 636),   (40, 6622, 636),
        (41, 6623, 636),   (42, 6624, 636),   (43, 6625, 636),   (44, 6626, 636),   (45, 6627, 636),
        (46, 6628, 636),   (47, 6629, 636),   (48, 6630, 636),   (49, 6631, 636),   (50, 6632, 636),
        (51, 6633, 636),   (52, 6634, 636),   (53, 6635, 636),   (54, 6636, 636),   (55, 6637, 636),
        (56, 6638, 636),   (57, 6639, 636),   (58, 6640, 636),   (59, 6641, 636),   (60, 6642, 636),
        (61, 6643, 636);
```

Payments

```
INSERT INTO `payments` (`paymentID`, `guestID`, `service_chargesID`, `currencyID`, `bookingID`, `payment_date`, `payment_method`, `amount`) VALUES
(1, 2, 2, 1, 1, '2023-12-10 03:49:53', 'Apple Pay', 21858),
(2, 3, 3, 2, 2, '2024-01-05 17:05:54', 'Credit Card', 8348),
(3, 3, 4, 3, 3, '2023-12-12 02:58:24', 'Paypal', 140693),
(4, 3, 4, 4, 4, '2023-11-22 12:10:43', 'Debit Card', 358619),
(5, 4, 5, 5, 5, '2023-12-22 23:55:20', 'Paypal', 333266),
(6, 4, 3, 6, 6, '2023-12-06 03:15:41', 'Paypal', 8806),
(7, 5, 3, 7, 7, '2024-03-01 02:41:26', 'Credit Card', 1562356),
(8, 5, 4, 8, 8, '2023-11-20 21:09:28', 'Debit Card', 1991925),
(9, 6, 4, 9, 9, '2023-12-13 21:05:35', 'Paypal', 7517),
(10, 6, 2, 11, 10, '2024-01-27 23:27:10', 'Paypal', 7071),
(11, 6, 3, 12, 11, '2024-02-09 23:01:59', 'Credit Card', 3587),
(12, 7, 3, 13, 12, '2024-01-25 17:57:17', 'Paypal', 4047),
(13, 7, 2, 14, 13, '2023-11-14 01:07:04', 'Debit Card', 6003),
(14, 7, 3, 15, 14, '2023-11-26 16:50:06', 'Apple Pay', 176423),
(15, 7, 1, 16, 15, '2023-12-10 08:16:13', 'Paypal', 3477),
(16, 7, 1, 17, 16, '2023-11-28 21:23:33', 'Debit Card', 322),
(17, 8, 3, 18, 17, '2024-02-18 23:41:08', 'Debit Card', 5789982),
(18, 8, 5, 19, 18, '2023-12-07 00:19:22', 'Debit Card', 522),
(19, 9, 3, 21, 19, '2024-01-24 10:31:27', 'Debit Card', 3569),
(20, 9, 1, 22, 20, '2023-12-10 10:46:53', 'Alipay', 11462);
```

Payments taxes

```
INSERT INTO `payments_taxes` (`payments_taxesID`, `paymentID`, `taxID`) VALUES
        (1, 1, 21),     (2, 1, 22),     (3, 1, 23),     (4, 1, 24),     (5, 1, 25),
        (6, 1, 26),     (7, 1, 27),     (8, 1, 28),     (9, 1, 29),     (10, 1, 30),
        (11, 1, 31),    (12, 2, 21),    (13, 2, 22),    (14, 2, 23),    (15, 2, 24),
        (16, 2, 25),    (17, 2, 26),    (18, 2, 27),    (19, 2, 28),    (20, 2, 29),
        (21, 2, 30),    (22, 2, 31),    (23, 3, 21),    (24, 3, 22),    (25, 3, 23),
        (26, 3, 24),    (27, 3, 25),    (28, 3, 26),    (29, 3, 27),    (30, 3, 28),
        (31, 3, 29),    (32, 3, 30),    (33, 3, 31),    (34, 3, 32),    (35, 4, 21),
        (36, 4, 22),    (37, 4, 23),    (38, 4, 24),    (39, 4, 25),    (40, 4, 26),
        (41, 4, 27),    (42, 4, 28),    (43, 4, 29),    (44, 4, 30),    (45, 4, 31),
        (46, 4, 32),    (47, 4, 33),    (48, 4, 34),    (49, 4, 35),    (50, 4, 36),
        (51, 4, 37),    (52, 5, 21),    (53, 5, 22),    (54, 5, 23),    (55, 5, 24),
        (56, 5, 25),    (57, 5, 26),    (58, 5, 27),    (59, 5, 28),    (60, 5, 29);
```

Property Category

```
INSERT INTO `property_category` (`categoryID`, `listingID`, `budget`, `standard`, `luxury`) VALUES
        (1, 1, 0, 1, 0),
        (2, 2, 1, 0, 0),
        (3, 3, 0, 1, 0),
        (4, 4, 1, 0, 0),
        (5, 5, 0, 1, 0),
        (6, 6, 0, 0, 1),
        (7, 7, 0, 1, 0),
        (8, 8, 0, 0, 1),
        (9, 9, 0, 0, 1),
        (10, 10, 0, 1, 0),
        (11, 11, 0, 0, 1),
        (12, 12, 0, 0, 1),
        (13, 13, 0, 0, 1),
        (14, 14, 0, 0, 1),
        (15, 15, 0, 0, 1),
        (16, 16, 0, 0, 1),
        (17, 17, 0, 0, 1),
        (18, 18, 0, 1, 0),
        (19, 19, 0, 0, 1),
        (20, 20, 1, 0, 0);
```

Property type

```
INSERT INTO `property_type` (`typeID`, `listingID`, `appartment`, `house`, `vacation_rental`) VALUES
        (1, 1, 1, 0, 0),
        (2, 2, 0, 1, 0),
        (3, 3, 1, 0, 0),
        (4, 4, 0, 0, 1),
        (5, 5, 0, 0, 1),
        (6, 6, 0, 1, 0),
        (7, 7, 0, 0, 1),
        (8, 8, 0, 0, 1),
        (9, 9, 0, 0, 1),
        (10, 10, 0, 1, 0),
        (11, 11, 0, 0, 1),
        (12, 12, 0, 0, 1),
        (13, 13, 0, 1, 0),
        (14, 14, 0, 0, 1),
        (15, 15, 1, 0, 0),
        (16, 16, 0, 1, 0),
        (17, 17, 0, 1, 0),
        (18, 18, 1, 0, 0),
        (19, 19, 1, 0, 0),
        (20, 20, 1, 0, 0);
```

Ratings on guest

```
INSERT INTO `ratings_on_guest` (`ratingID`, `guestID`, `rating`) VALUES
        (1, 2, 5),
        (2, 3, 5),
        (3, 4, 5),
        (4, 5, 5),
        (5, 6, 5),
        (6, 7, 5),
        (7, 8, 5),
        (8, 9, 5),
        (9, 10, 5),
        (10, 11, 5),
        (11, 12, 5),
        (12, 13, 5),
        (13, 14, 5),
        (14, 15, 5),
        (15, 16, 5),
        (16, 17, 5),
        (17, 18, 5),
        (18, 2, 5),
        (19, 3, 5),
        (20, 4, 5);
```

Ratings on host

```
INSERT INTO `ratings_on_host` (`ratingID`, `hostID`, `rating`) VALUES
(1, 2, 5),  (2, 3, 5), (3, 3, 5), (4, 3, 4), (5, 4, 5), (6, 4, 5), (7, 5, 5), (8, 5, 5), (9, 6, 5), (10, 6,
5),  (11, 6, 5),  (12, 7, 5),  (13, 7, 5),  (14, 7, 5),  (15, 7, 5),
     (16, 7, 5),  (17, 8, 5),  (18, 8, 5),  (19, 9, 5),  (20, 9, 5),
     (21, 9, 5),  (22, 9, 5),  (23, 9, 4),  (24, 10, 5),  (25, 10, 5),
     (26, 10, 5),  (27, 10, 5),  (28, 10, 4),  (29, 11, 5),  (30, 11, 5),
     (31, 11, 5),  (32, 11, 5),  (33, 11, 5),  (34, 12, 5),  (35, 12, 5),
     (36, 12, 5),  (37, 12, 5),  (38, 12, 5),  (39, 13, 5),  (40, 15, 5),
     (41, 15, 5),  (42, 15, 4),  (43, 15, 5),  (44, 16, 5),  (45, 16, 5),
     (46, 16, 5),  (47, 16, 5),  (48, 16, 5),  (49, 17, 5);
```

Reviews

```
INSERT INTO `reviews` (`reviewID`, `listingID`, `guestID`, `timestamp`, `comment`, `rating`) VALUES
        (1, 9, 2, '2024-02-06 05:10:47', 'It was a good place, coming back again.', 5),
        (2, 5, 3, '2024-01-16 15:19:52', 'Lovely Place.', 5),
        (3, 15, 4, '2024-02-08 00:52:20', 'Nice stay.', 5),
        (4, 10, 4, '2024-02-11 17:40:06', 'Lovely Place.', 5),
        (5, 18, 5, '2024-02-29 18:34:39', 'We old timers had a great time together with the host. Very hospitable. Great people.', 5),
        (6, 1, 5, '2024-02-03 15:18:49', 'Good place.', 5),
        (7, 17, 6, '2024-02-28 03:44:21', 'Good place.', 5),
        (8, 12, 6, '2024-02-05 17:23:21', 'Good.', 4),
        (9, 18, 6, '2024-02-20 03:39:33', 'Our Family loved the visit! We recommend it to anyone!', 5),
        (10, 9, 7, '2024-02-07 18:34:51', 'It was a good place, coming back again.', 5),
        (11, 1, 7, '2024-01-05 15:06:49', 'Best host ever! Highly recommended!', 5),
        (12, 2, 7, '2024-01-06 00:19:49', 'We old timers had a great time together with the host. Very hospitable. Great people.', 5),
        (13, 8, 7, '2024-01-19 22:17:27', 'I loved the place and the host!', 5),
        (14, 6, 7, '2024-01-16 15:52:22', 'The residence was bright, the night sky views were amazing.', 5),
        (15, 17, 8, '2024-02-11 20:09:43', 'It was Alright.', 4),
        (16, 11, 8, '2024-01-21 15:31:02', 'Its Okay.', 2),
        (17, 10, 9, '2024-01-25 04:45:42', 'We old timers had a great time together with the host. Very hospitable. Great people.', 5),
        (18, 8, 9, '2024-01-26 19:35:38', 'Thanks the host for great hospitality!', 5),
        (19, 16, 9, '2024-02-24 12:24:50', 'It was a good place, coming back again.', 5),
        (20, 18, 10, '2024-02-19 00:28:20', 'Good.', 4),
        (21, 17, 10, '2024-02-11 08:36:57', 'It was Alright.', 3),
        (22, 1, 10, '2024-02-01 15:44:50', 'Its Okay.', 3),
        (23, 14, 10, '2024-02-18 23:49:44', 'It was great! Thanks to host for hospitality.', 5),
        (24, 16, 10, '2024-02-25 20:18:56', 'Good place.', 4),
        (25, 9, 11, '2024-01-22 21:36:22', 'It was Alright.', 4);
```

Room type

```
INSERT INTO `room_type` (`roomID`, `listingID`, `small`, `medium`, `large`) VALUES
        (1, 1, 1, 0, 0),      (2, 2, 0, 1, 0),      (3, 3, 1, 0, 0),      (4, 4, 0, 0, 1),      (5, 5, 0, 0, 1),
        (6, 6, 0, 1, 0),      (7, 7, 0, 0, 1),      (8, 8, 0, 0, 1),      (9, 9, 0, 0, 1),      (10, 10, 0, 1, 0),
        (11, 11, 0, 0, 1),    (12, 12, 0, 0, 1),    (13, 13, 0, 1, 0),    (14, 14, 0, 0, 1),    (15, 15, 1, 0, 0),
        (16, 16, 0, 1, 0),    (17, 17, 0, 1, 0),    (18, 18, 1, 0, 0),    (19, 19, 0, 0, 1),    (20, 20, 1, 0, 0);
```

Service charges

```
INSERT INTO `service_charges` (`service_chargesID`, `amount`, `service_grade`) VALUES
        (1, 1.5000, 'service_grade_1'),        (2, 3.4500, 'service_grade_2'),
        (3, 10.8900, 'service_grade_3'),       (4, 23.7500, 'service_grade_4'),
        (5, 50.2500, 'service_grade_5'),       (6, 9.0000, 'service_grade_6'),
        (7, 31.0000, 'service_grade_7'),       (8, 91.0000, 'service_grade_8'),
        (9, 46.0000, 'service_grade_9'),       (10, 67.0000, 'service_grade_10'),
        (11, 27.0000, 'service_grade_11'),     (12, 95.0000, 'service_grade_12'),
        (13, 14.0000, 'service_grade_13'),     (14, 54.0000, 'service_grade_14'),
        (15, 80.0000, 'service_grade_15'),     (16, 44.0000, 'service_grade_16'),
        (17, 42.0000, 'service_grade_17'),     (18, 22.0000, 'service_grade_18'),
        (19, 72.0000, 'service_grade_19'),     (20, 87.0000, 'service_grade_20');
```

Taxes

```
INSERT INTO `taxes` (`taxID`, `tax_amount`, `tax_type`) VALUES
        (21, 5.00000, 'Transient Occupancy Tax'),      (22, 6.20000, 'Tourist Development Tax'),
        (23, 12.40000, 'Municipal Hotel Tax'),         (24, 9.60000, 'City Lodging Fee'),
        (25, 2.80000, 'Occupancy Assessment Tax'),  (26, 11.00000, 'Accommodation Tax'),
        (27, 18.20000, 'Lodging Tax'),         (28, 16.40000, 'Resort Tax'),
        (29, 9.60000, 'Tourism Improvement District Assessment'),
        (30, 2.80000, 'Hospitality Tax'),
        (31, 17.00000, 'Room Surcharge'),   (32, 9.20000, 'Tourism Promotion Fee'),
        (33, 16.40000, 'Guest Room Assessment'),    (34, 4.60000, 'City Sales Tax'),
        (35, 9.80000, 'Local Tourism Levy'), (36, 13.00000, 'Destination Marketing Fee'),
        (37, 20.20000, 'Tourism Infrastructure Fee'), (38, 15.40000, 'Hotel Occupancy Fee'),
        (39, 1.60000, 'Bed Tax'),     (40, 17.80000, 'Local Government Assessment');
```

User

```
INSERT INTO `user` (`userID`, `name`, `email`, `phone_number`, `password`, `date_created`) VALUES
(1, 'Kuruma Ikusai', 'kurumakun@japan.com', '03534643523', 'vrejkgnljvdfwe312553', '2023-10-06 05:25:57'),
(2, 'Adam B. Junior', 'adamjunior@gmail.com', '04126436347', 'tetete34S.', '2023-08-10 03:22:11'),
(3, 'Anna Hamza', 'annahamza9@gmail.com', '05705466793', 'lhCTqABtQJjl5UP', '2023-10-16 22:50:08'),
(4, 'Emma Omar', 'emmaomar4@outlook.com', '02965292380', 'e8su47w1cfxpHyO', '2023-08-05 20:10:18'),
(5, 'Jeronimo Mateo', 'jeronimomateo6@gmail.com', '04542877133', 'FbHqVG3khGBCd1G', '2023-09-11 10:04:48'),
(6, 'Lena Mateo', 'lenamateo7@gmail.com', '08636902364', '2nrrnb3444waLWw7nE', '2023-10-21 09:17:17'),
(7, 'Omar Hatimi', 'omarhatimi7@hotmail.com', '06161195341', 'QntZ6siz8', '2023-10-22 22:01:53'),
(8, 'Hussein Agustin', 'husseinagustin8@gmail.com', '08032261939', 'jSvMD83fJrz0qGn', '2023-09-08 11:26:48'),
(9, 'Hasnaa Selim', 'hasnaaselim0@hotmail.com', '06071849909', 'ftM4vcQe0BwA7h6', '2023-09-30 11:06:07'),
(10, 'Mila Nathan', 'milanathan6@microsoft.com', '02585352003', 'u9dl7Jc8bM', '2023-10-29 13:38:00'),
(11, 'Miguel Angel Benjamin', 'miguelangelbenjamin3@hotmail.com', '08141809877', 'J9PCuZbnhOtzSv8', '2023-10-07 16:41:21'),
(12, 'Sara Omar', 'saraomar2@microsoft.com', '09012730365', 'u1uYNXD4ER6X', '2023-08-10 17:33:11'),
(13, 'Viktoria Noah', 'viktorianoah2@outlook.com', '04087149004', 'Jy4GYqylum0gpR', '2023-10-17 13:57:56'),
(14, 'Mia Oliver', 'miaoliver4@yahoo.com', '05040346441', 'Tbgaed2Wed95rak', '2023-09-25 10:14:16'),
(15, 'Felipe Thomas', 'felipethomas5@gmail.com', '04552269639', 'JZp555RXCqCqXeUS', '2023-08-01 07:15:20'),
(16, 'Jacob Abdel-Rahman', 'jacobabdel-rahman7@gmail.com', '01376473475', 'HM7cy1Q2aUb6JAo', '2023-09-29 15:00:18'),
(17, 'Shaimaa Saeed', 'shaimaasaeed4@yahoo.com', '07646640697', '1J3eCwI8wrrYLB', '2023-10-17 16:37:10'),
(18, 'Lea Hamza', 'leahamza5@hotmail.com', '01926040235', '3FKFxbwI5KClJfW', '2023-10-22 08:35:41'),
(19, 'Thiago Ibrahim', 'thiagoibrahim9@yahoo.com', '08320815283', '0yeVDyBTmC4kv', '2023-10-24 08:31:30'),
(20, 'Juan Bilal', 'juanbilal3@yahoo.com', '05039234208', 'aQKBs7d7b3OmTUT', '2023-10-15 08:07:51'),
(21, 'Melisa Nathan', 'melisanathan7@yahoo.com', '04987081669', 'z39ZWdmsALZ0V', '2023-08-30 09:51:24'),
(22, 'Lucas Benjamin', 'lucasbenjamin0@microsoft.com', '01448744059', 'jukGKtp1Jf', '2023-08-22 12:02:33'),
(23, 'Martin Benjamin', 'martinbenjamin8@yahoo.com', '04397775532', 'runaPOZP4ho9M', '2023-08-13 15:01:27'),
(24, 'Kevin Tareq', 'kevintareq0@yahoo.com', '07005677558', 'WUS88IrQKYyEtE', '2023-08-24 03:26:01'),
(25, 'Jose Luis Ali', 'joseluisali7@yahoo.com', '09080421870', '71zcoG0tJfj8', '2023-08-21 22:52:28'),
(26, 'Gamila Ali', 'gamilaali9@hotmail.com', '04262540309', 'YLX9vHvFEqJrZ', '2023-08-17 02:04:17'),
(27, 'Emma Liam', 'emmaliam2@hotmail.com', '06582513010', 'yKO3b9VbR', '2023-09-23 18:20:34'),
(28, 'Marcos Taha', 'marcostaha9@gmail.com', '02374493261', 'ZbSHtXcf6EFsA', '2023-09-01 04:19:06'),
(29, 'Mila Selim', 'milaselim3@microsoft.com', '03143035689', '9rhN0GbxGO1NAI', '2023-10-09 01:24:30'),
(30, 'Eleni Thomas', 'elenithomas8@yahoo.com', '01853829015', 'Jns7BjLYYCnVJ', '2023-09-06 18:05:24'),
(31, 'Maria Edouard', 'mariaedouard1@hotmail.com', '05992618990', 'WyD9dHzWTGMgI', '2023-08-28 23:42:26'),
(32, 'Noah Oliver', 'noaholiver5@microsoft.com', '02710208299', '2d9r5nK2bekV', '2023-10-22 07:56:59'),
(33, 'Maximiliano Hatimi', 'maximilianohatimi3@hotmail.com', '01157786526', 'YIHBnrlt6Oesc', '2023-08-03 15:31:30'),
(34, 'Liam Liam', 'liamliam4@gmail.com', '08286927494', 'es25DZ8jIj4', '2023-08-08 23:46:01'),
(35, 'Emilia Benjamin', 'emiliabenjamin5@outlook.com', '02747422012', 'skTThXJGBZqOXpu', '2023-09-10 15:12:11'),
(36, 'Dylan Hatimi', 'dylanhatimi7@yahoo.com', '05110830441', 'aTOydPXpWGR8t', '2023-10-08 07:11:00'),
(37, 'Lucia Mateo', 'luciamateo2@hotmail.com', '01884539063', 'dLM47hqvJfGmd', '2023-10-27 01:34:58'),
(38, 'Emma Leo', 'emmaleo2@outlook.com', '02338015294', 'kdxZAoK7Uk', '2023-09-24 22:33:24'),
(39, 'Qasim Querishi', 'qasimq11@yahoo.com', '04981346669', 'z39ZWdmsALZ0V', '2023-08-10 23:07:39'),
(40, 'Donnie Yen', 'donnieyen@microsoft.com', '011448745641', 'jukGKtp1Jf', '2023-08-06 01:35:25');
```

User social networks

```
INSERT INTO `user_social_networks` (`socialID`, `userID`, `social_network_url`, `rating_enabled`, `created_at`, `updated_at`) VALUES
(1, 1, 'www.twitch.com/elenithomas', 1, '2023-11-13 13:46:32', NULL),
(2, 1, 'www.line.com/elenithomas', 1, '2023-11-13 13:46:32', NULL),
(3, 1, 'www.linkedin.com/elenithomas', 1, '2023-11-13 13:46:32', NULL),
(4, 1, 'www.Weibo.com/elenithomas', 1, '2023-11-13 13:46:32', NULL),
(5, 1, 'www.twitter.com/elenithomas', 1, '2023-11-13 13:46:32', NULL),
(6, 1, 'www.tiktok.com/elenithomas', 1, '2023-11-13 13:46:32', NULL),
(7, 2, 'www.facebook.com/dylanhatimi', 1, '2023-10-20 09:21:22', NULL),
(8, 2, 'www.twitter.com/dylanhatimi', 1, '2023-10-20 09:21:22', NULL),
(9, 2, 'www.whatsapp.com/dylanhatimi', 1, '2023-10-20 09:21:22', NULL),
(10, 3, 'www.wechat.com/dylanhatimi', 1, '2023-09-04 01:48:16', NULL),
(11, 3, 'www.instagram.com/dylanhatimi', 1, '2023-09-04 01:48:16', NULL),
(12, 3, 'www.telegram.com/dylanhatimi', 1, '2023-09-04 01:48:16', NULL),
(13, 3, 'www.snapchat.com/dylanhatimi', 1, '2023-09-04 01:48:16', NULL),
(14, 3, 'www.wechat.com/dylanhatimi', 1, '2023-09-04 01:48:16', NULL),
(15, 3, 'www.whatsapp.com/dylanhatimi', 1, '2023-09-04 01:48:16', NULL),
(16, 4, 'www.facebook.com/lenamateo', 1, '2023-11-30 08:03:56', NULL),
(17, 4, 'www.twitter.com/lenamateo', 1, '2023-11-30 08:03:56', NULL),
(18, 4, 'www.twitter.com/lenamateo', 1, '2023-11-30 08:03:56', NULL),
(19, 4, 'www.linkedin.com/lenamateo', 1, '2023-11-30 08:03:56', NULL),
(20, 4, 'www.snapchat.com/lenamateo', 1, '2023-11-30 08:03:56', NULL);
```

Wishlist

```
INSERT INTO `wishlist` (`wishlistID`, `guestID`, `listingID`, `wishlist_name`, `listing_info`, `url`) VALUES
        (1, 2, 17, 'villages', NULL, 'www.airbnb.com/wishlist/user21'),
        (2, 2, 6, 'tallest places', NULL, 'www.airbnb.com/wishlist/user21'),
        (3, 2, 16, 'tallest places', NULL, 'www.airbnb.com/wishlist/user21'),
        (4, 2, 16, 'villages', NULL, 'www.airbnb.com/wishlist/user21'),
        (5, 2, 3, 'city places', NULL, 'www.airbnb.com/wishlist/user21'),
        (6, 2, 2, 'mountains', NULL, 'www.airbnb.com/wishlist/user21'),
        (7, 3, 17, 'nature', NULL, 'www.airbnb.com/wishlist/user22'),
        (8, 3, 14, 'cool apartments', NULL, 'www.airbnb.com/wishlist/user22'),
        (9, 3, 7, 'tallest places', NULL, 'www.airbnb.com/wishlist/user22'),
        (10, 3, 14, 'cool apartments', NULL, 'www.airbnb.com/wishlist/user22'),
        (11, 4, 19, 'tallest places', NULL, 'www.airbnb.com/wishlist/user23'),
        (12, 4, 16, 'villages', NULL, 'www.airbnb.com/wishlist/user23'),
        (13, 4, 2, 'mountains', NULL, 'www.airbnb.com/wishlist/user23'),
        (14, 4, 11, 'city places', NULL, 'www.airbnb.com/wishlist/user23'),
        (15, 4, 9, 'mountains', NULL, 'www.airbnb.com/wishlist/user23'),
        (16, 5, 12, 'villages', NULL, 'www.airbnb.com/wishlist/user24'),
        (17, 5, 17, 'mountains', NULL, 'www.airbnb.com/wishlist/user24'),
        (18, 5, 9, 'villages', NULL, 'www.airbnb.com/wishlist/user24'),
        (19, 5, 6, 'nature', NULL, 'www.airbnb.com/wishlist/user24'),
        (20, 5, 12, 'mountains', NULL, 'www.airbnb.com/wishlist/user24');
```

Test Cases

Simple SQL

What was the most preferred method of payment?

```
SELECT payment_method, COUNT(payment_method)
AS `usage`
FROM payments
GROUP BY payment_method
ORDER BY `usage` DESC;
```

We see that PayPal and Credit card were the most preferred methods of payment.

payments (8r × 2c)

payment_method	usage
Paypal	14
Debit Card	13
Apple Pay	6
Credit Card	6
Google Pay	5
Alipay	2
Bank Account	2
Sofort Überweisung	1

Figure 32

Normal SQL

What was the discount percent with coupons on each payment?

```
SELECT p.amount, c.discount,  cu.name AS
'currency',
((p.amount/(p.amount - c.discount))-1)*100 AS
'% discount'
FROM payments p, coupons c, currency cu
WHERE p.bookingID = c.bookingID
AND p.currencyID = cu.currencyID;
```

The result shows payments, applied discounts and discount percentages in respective currencies.

Result #1 (43r × 4c)

amount	discount	currency	% discount
21,858	20	Cardano	0.0916
358,619	20	Albanian Lek	0.0056
333,266	30	Armenian Dram	0.0090
8,806	30	NL Antillean Guilder	0.3418
1,562,356	30	Angolan Kwanza	0.0019
1,991,925	30	Argentine Peso	0.0015
7,517	100	Australian Dollar	1.3483
7,071	50	Aruban Florin	0.7121
3,587	100	Azerbaijani Manat	2.8678
4,047	30	Bosnia-Herzegovina Convertible Mark	0.7468
6,003	30	Barbadian Dollar	0.5023
176,423	20	Bangladeshi Taka	0.0113
3,477	30	Bulgarian Lev	0.8703
322	50	Bahraini Dinar	18.3824

Figure 33

Moderate SQL

Describe the spread of average days spent by guests staying in each category of property.

```
SELECT  COUNT(*), luxury, standard, budget,
ROUND(AVG(DATEDIFF(booking_from,
booking_date))) AS 'avg booking days',
ROUND(AVG(DATEDIFF(booking_until,
booking_from))) AS 'avg stay days'
FROM booking INNER JOIN property_category
USING (listingID)
WHERE DATEDIFF(booking_until,booking_date) > 0
GROUP BY budget, standard, luxury;
```

Result describes: many guests stay in luxury properties for around 3 weeks, meanwhile some guests stay about a month in standard properties and few guests spend around half a month in budget properties. Bookings of all properties are made two to three weeks before.

property_category (3r × 6c)

COUNT(*)	luxury	standard	budget	avg booking days	avg stay days
24	1	0	0	15	23
14	0	1	0	11	27
2	0	0	1	24	18

Figure 34

Complex SQL

How much was the monthly earning of Airbnb deducting taxes and payables?

```
SELECT `year-month`, SUM(amount) AS 'revenue', FORMAT(SUM(tax),1) AS 'tax', SUM(`service
charges`) AS 'services', FORMAT(SUM(amount)-SUM(tax)+SUM(`service charges`),1) AS 'income',
`currency`
FROM (
SELECT DATE_FORMAT(payment_date , '%Y %m') AS 'year-month',
ROUND(payments.amount/exchange_rate) AS 'amount',
CEIL(service_charges.amount/exchange_rate) AS 'service charges',
SUM(t.tax_amount) AS 'tax',
'USD' AS `currency` FROM payments
INNER JOIN booking USING(bookingID)
INNER JOIN currency USING (currencyID)
INNER JOIN service_charges USING(service_chargesID)
INNER JOIN payments_taxes USING (paymentID)
INNER JOIN taxes t USING(taxID)
GROUP BY payments.paymentID
) AS subquery
GROUP BY `year-month` ORDER BY `year-month`;
```

Result #1 (5r × 6c)

year-month	revenue	tax	services	income	currency
2023 11	19,208	1,171.2	226	18,262.8	USD
2023 12	50,145	2,576.8	92	47,660.2	USD
2024 01	42,315	2,552.8	118	39,880.2	USD
2024 02	14,570	1,395.0	141	13,316.0	USD
2024 03	1,893	201.2	1	1,692.8	USD

Figure 35

SQL JOIN

Lets find out how many hosts own rentals in each country.

```
SELECT user.userID, `name`, country,
COUNT(ls.listing_addressID) AS `rentals owned`
FROM user
INNER JOIN host on user.userID = host.userID
INNER JOIN listing l ON l.hostID = host.hostID
INNER JOIN listing_address ls ON
ls.listing_addressID = l.listing_addressID
GROUP BY country, host.hostID
ORDER BY `rentals owned` DESC, country;
```

The result shows us that each host only owns a single rental in a given country. If a host owned multiple rentals in many countries, that would have showed here in the result.

Result #1 (20r × 4c)

userID	name	country	rentals owned
14	Mia Oliver	Argentina	1
4	Emma Omar	Australia	1
7	Omar Hatimi	Germany	1
19	Thiago Ibrahim	Germany	1
10	Mila Nathan	Germany	1
6	Lena Mateo	Germany	1
20	Juan Bilal	Germany	1
13	Viktoria Noah	Hungary	1
9	Hasnaa Selim	Indonesia	1
12	Sara Omar	Indonesia	1
18	Lea Hamza	Italy	1
1	Kuruma Ikusai	Japan	1
11	Miguel Angel Benjamin	Mexico	1
16	Jacob Abdel-Rahman	USA	1

Figure 36

SQL UNION

Airbnb wants to create a consolidated list of both hosts and guests with their basic details.

```
SELECT h.userID, name, email, `password`,
'Host' AS role
FROM host h
INNER JOIN user u ON h.userID = u.userID
UNION
SELECT g.userID, name, email, `password`,
'Guest' AS role
FROM guest g
INNER JOIN user u ON g.userID = u.userID;
```

Here we get a combined table identifying users as hosts or guests with their respective information.

listing (40r × 5c)

userID	name	email	password	role
15	Felipe Thomas	felipethomas5@gmail.com	JZp555RXCqCqXeUS	Host
16	Jacob Abdel-Rahman	jacobabdel-rahman7@gmail.com	HM7cy1Q2aUb6JAo	Host
17	Shaimaa Saeed	shaimaasaeed4@yahoo.com	1J3eCwI8wrrYLB	Host
18	Lea Hamza	leahamza5@hotmail.com	3FKFxbwI5KClJfW	Host
19	Thiago Ibrahim	thiagoibrahim9@yahoo.com	0yeVDyBTmC4kv	Host
20	Juan Bilal	juanbilal3@yahoo.com	aQKBs7d7b3OmTUT	Host
21	Melisa Nathan	melisanathan7@yahoo.com	z39ZWdmsALZ0V	Guest
22	Lucas Benjamin	lucasbenjamin0@microsoft.com	jukGKtp1Jf	Guest
23	Martin Benjamin	martinbenjamin8@yahoo.com	runaPOZP4ho9M	Guest
24	Kevin Tareq	kevintareq0@yahoo.com	WUS88IrQKYyEtE	Guest
25	Jose Luis Ali	joseluisali7@yahoo.com	71zcoG0tJfj8	Guest
26	Gamila Ali	gamilaali9@hotmail.com	YLX9vHvFEqJrZ	Guest
27	Emma Liam	emmaliam2@hotmail.com	yKO3b9VbR	Guest
28	Marcos Taha	marcostaha9@gmail.com	ZbSHtXcf6EFsA	Guest
29	Mila Salim	milasalim2@microsoft.com	0rbN0GbxGO1NAl	Guest

Figure 37

Remarks

Potential Improvements

- Instead of two extra tables for guests and hosts, the `user` table could have an extra column to identify users as host or guest in a single table.
- Ratings of hosts and guests could be stored in one table instead of separate tables.
- Existing columns of some tables can be used as Primary Keys instead of system-defined Primary Keys (e.g. `` `currency`.`symbol` ``).
- Instead of using Surrogate Keys for junction tables, a Combined Primary Key could be made from the existing Foreign Keys.
- Some tables could be removed if defining the minimum cardinality relationship between entities suffices the need, or if catering for rare occurring events is not required (e.g. two apartments in the same building upload same images for their listing, creating a m:m relationship, hence requiring a junction table between `listing` and `images` tables).

Limitations

- Existing structure of the database schema is not efficient enough, since data is spread over more tables then what is sufficient.
- Implementing the summation of `host` and `guest` tables into `user` table can mean structural changes across the database since the database schema follows a hierarchical structure and tables in lower hierarchy depend on tables in upper hierarchy.
- Data stored in existing structure would be lost, or would require extra effort to be moved to an appropriate site and then reinserted within the new structure with the correct format in the database.

Summary

Airbnb will have users that can be divided into two groups, 'hosts' and 'guests'. Hosts will make account on website and list their places to be rented with pictures, addresses and other respective requirements. Guests will also make account and provide necessary information such as name, contact info and valid identification documents for security and other purposes. Guests will book listings and make payments in their available currencies. Taxes, service charges and other types of costs will be included in the bills of guests by Airbnb. Guests will also receive coupons and discount offers from Airbnb. Moreover, guests can give reviews to a listing of a host and; hosts and guests can chat with each other on messaging service provided by Airbnb before bookings or payments are made or at any other times. Hosts and Guests can give ratings to each others' profiles as well as list languages they can speak for ease of use and accessibility. Since guests would like to save potential listings for future visits so wish-list feature is provided by Airbnb. Social media accounts of users will be listed so users can connect with each other using various channels.

List of Figures

- Figure 10. | Source: Author
- Figure 11. | Source: Author
- Figure 12. | Source: Author
- Figure 13. | Source: Author
- Figure 14. | Source: Author
- Figure 15. | Source: Author
- Figure 16. | Source: Author
- Figure 17. | Source: Author
- Figure 18. | Source: Author
- Figure 19. | Source: Author
- Figure 20. | Source: Author
- Figure 21. | Source: Author

- Figure 22. | Source: Author
- Figure 23. | Source: Author
- Figure 24. | Source: Author
- Figure 25. | Source: Author
- Figure 26. | Source: Author
- Figure 27. | Source: Author
- Figure 28. | Source: Author
- Figure 29. | Source: Author
- Figure 30. | Source: Author
- Figure 31. | Source: Author
- Figure 32. | Source: Author
- Figure 33. | Source: Author

- Figure 34. | Source: Author
- Figure 35. | Source: Author
- Figure 36. | Source: Author
- Figure 37. | Source: Author

Phase 3 work

Finalization Phase

Abstract:

A database was created for Airbnb business. The business has two types of users. Hosts and guest. Each category can perform a number of actions and create different types of data that need to be stored in relation to the category. A table is used to store all users that sign up on Airbnb website. Two separate tables are used to categorize users as hosts and guests. For these two tables, there are a number of different tables created that relate back to these tables and some of these tables also have relationships with each other. For hosts, their information needs to be stored in tables such as: personal host information, rental information like type of rental, number of rooms, available services, pictures and location etcetera. Similarly, for guests, information like how many bookings they make, how many payments they make, in what currency, service charges, taxes and their social accounts etcetera. Then, some user generated data like message conversations between hosts and guests, their reviews and ratings, their profile ratings etcetera.

Initially, the design consisted of about 23 tables but the design implementation was inadequate. This was because cardinality between different entities was not matching up to the realistic requirement. For example, ratings of hosts and guests were to be stored in a single table but both hosts and guests can give multiple ratings so storing that in one table meant that there would be a use of primary keys multiple times. This would be error prone so a better approach was to store guests' and hosts' reviews separately with their own table and primary keys. And using a junction table we can link back and forth between who gave reviews to whom and how much. Similar approach was taken to properly define relationships between payments and taxes table pair and messages and languages table pair. This was a number of junction tables were introduced to correctly define the required cardinality.

The database stores everything in its internal structure in a relational format. Some of the tables have NULL values allowed for their attributes, even though from the perspective of the database this is not good, but in the realistic application this makes sense because, for instance, in the `booking` table the attributes `check-in` and `check-out` will not be filled until the guest actually checks in the stay site and then checks out on his final day, so not allowing NULL values in the database will not work. These attribute values will be later updated by the system and stored in the database. Similarly, for the table `listing` the foreign key attribute `listing_addressID` is allowed NULL because, depending on how the Airbnb website works for the host, the listing will be created first and then later the listing address will be stored for that listing, so if the `listing_addressID` attribute is not allowed to be NULL then the database would not store the listing at all, which would not be acceptable for the application. Hence, other tables have similar cases that are solved by setting attributes as NULL.

Some key features of the database are the segregation of tables into two main hierarchies in the ERD sense. This should aid the developer querying the database make sense of how the schema is logically built. As for the database sense, there is enough data to get answers for many different questions so very complex queries can be run. For example, a developer can perform simple sentiment analysis by querying results that show counts of specific sentimental words – like ‘Good’, ‘Great’ or ‘okay’, ‘bad’ etc – in reviews for all the rentals stored in the database. Which can be further segregated into sentiment by country or type of rental etcetera.

Metadata:

Table	Rows	Size
amenities	20	32.0 KiB
availability	20	32.0 KiB
booking	49	48.0 KiB
coupons	43	32.0 KiB
currency	179	16.0 KiB
guest	21	48.0 KiB
guest_rating_on_host	49	48.0 KiB
host	17	64.0 KiB
host_language	20	48.0 KiB
host_ratings_on_guest	255	48.0 KiB
images	70	80.0 KiB
language	393	48.0 KiB
listing	21	64.0 KiB
listing_address	20	16.0 KiB
listing_images	80	48.0 KiB
messages	795	1.9 MiB
messages_language	999	96.0 KiB
payments	49	96.0 KiB
payments_taxes	753	80.0 KiB
property_category	20	32.0 KiB
property_type	20	32.0 KiB
ratings_on_guest	255	48.0 KiB
ratings_on_host	49	48.0 KiB
reviews	43	48.0 KiB
room_type	20	32.0 KiB
service_charges	20	16.0 KiB
taxes	20	32.0 KiB
user	38	16.0 KiB
user_social_networks	141	32.0 KiB
wishlsit	79	48.0 KiB
Total		
30	4558	3.2 MiB