PROJECT: BUILD A DATA MART IN SQL

AIRBNB

DEVELOPMENT PHASE

CHRISTOPHER MASUKUME

Table of Contents

01

Introduction

Abstract, workspace & Schema Creation

Abstract

To implement the Airbnb database, I had to design and create a database that can effectively handle the Airbnb use case. This started with developing an ER Diagram (Entity Relationship Diagram) to visualize the entities, attributes and relationships involved.

Based on the ER Diagram, SQL statements were used to create the tables which were carefully designed to store information such as properties, bookings, users and other related data. To ensure functionality, test cases were developed which covered various scenarios property booking and user registration and also data retrieval. To execute these test cases I was able to ensure that the creation process and reproduction is possible.

Screenshots of the DBMS were captured to provide a visual representation of the implementations, and the screenshots showed actual data that is stored in the entities and the relationships between these entities. These are documented as part of the implementation process to ensure clarity and facilitate understanding of the implemented database structure. Throughout the implemented process feedback and hints were research from online tutorials and discussion forums to ensure accuracy and avoid making errors. These sources helped me understand the depth of database implementation and creation.

In conclusion, the implementation of the Airbnb DBMS required proper designs, SQL statements and test case development. The current database will provide a solid foundation for managing and meeting the requirement outlined in this use case.

Workstation

For the Airbnb use case I used MySQL workbench as my Database Management System and to design my ER-Diagram. The DBMS used is a Community version which is an open source and freely downloadable. It is largely supported by a big community of developers, and it comes under a GPL License.

To set up the work environment one is required to install MySQL server before the workbench application.

Workstation

For data appropriation I used a software called dbForge Studio for MySQL to create dummy data that is used within the database for the Airbnb use case.

dbForge Studio for MySQL

Schema Creation

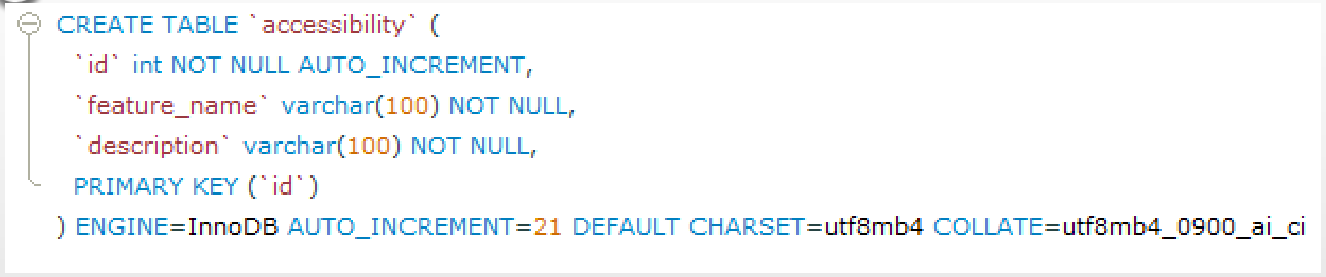
We use a schema called “airbnb” which is a logical representation of a database that describes the structural definition or description of entire database.

02

Table Creation

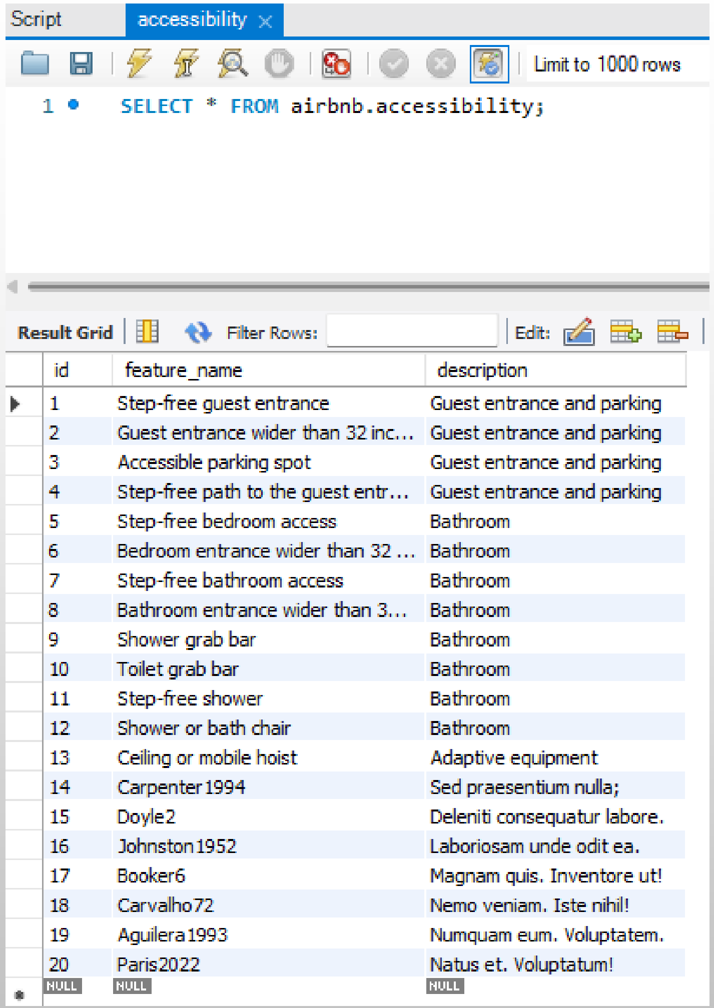
Creation of tables together with their SQL Codes

• ACCESSIBILITY



```
CREATE TABLE `accessibility` (
  `id` int NOT NULL AUTO_INCREMENT,
  `feature_name` varchar(100) NOT NULL,
  `description` varchar(100) NOT NULL,
  PRIMARY KEY (`id`)
) ENGINE=InnoDB AUTO_INCREMENT=21 DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4_0900_ai_ci
```

Create table

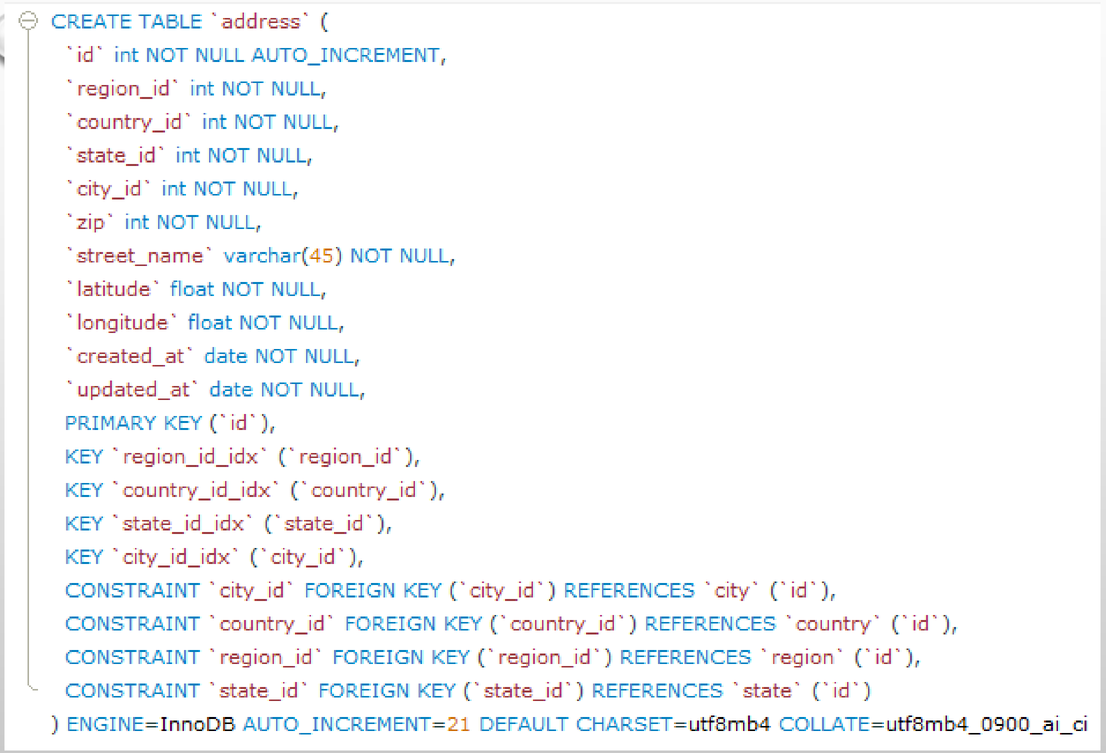


```
SELECT * FROM airbnb.accessibility;
```

id	feature_name	description
1	Step-free guest entrance	Guest entrance and parking
2	Guest entrance wider than 32 inc...	Guest entrance and parking
3	Accessible parking spot	Guest entrance and parking
4	Step-free path to the guest entr...	Guest entrance and parking
5	Step-free bedroom access	Bathroom
6	Bedroom entrance wider than 32 ...	Bathroom
7	Step-free bathroom access	Bathroom
8	Bathroom entrance wider than 3...	Bathroom
9	Shower grab bar	Bathroom
10	Toilet grab bar	Bathroom
11	Step-free shower	Bathroom
12	Shower or bath chair	Bathroom
13	Ceiling or mobile hoist	Adaptive equipment
14	Carpenter1994	Sed praesentium nulla;
15	Doyle2	Deleniti consequatur labore.
16	Johnston1952	Laboriosam unde odit ea.
17	Booker6	Magnam quis. Inventore ut!
18	Carvalho72	Nemo veniam. Iste nihil!
19	Aguilera1993	Numquam eum. Voluptatem.
20	Paris2022	Natus et. Voluptatum!
NULL	NULL	NULL

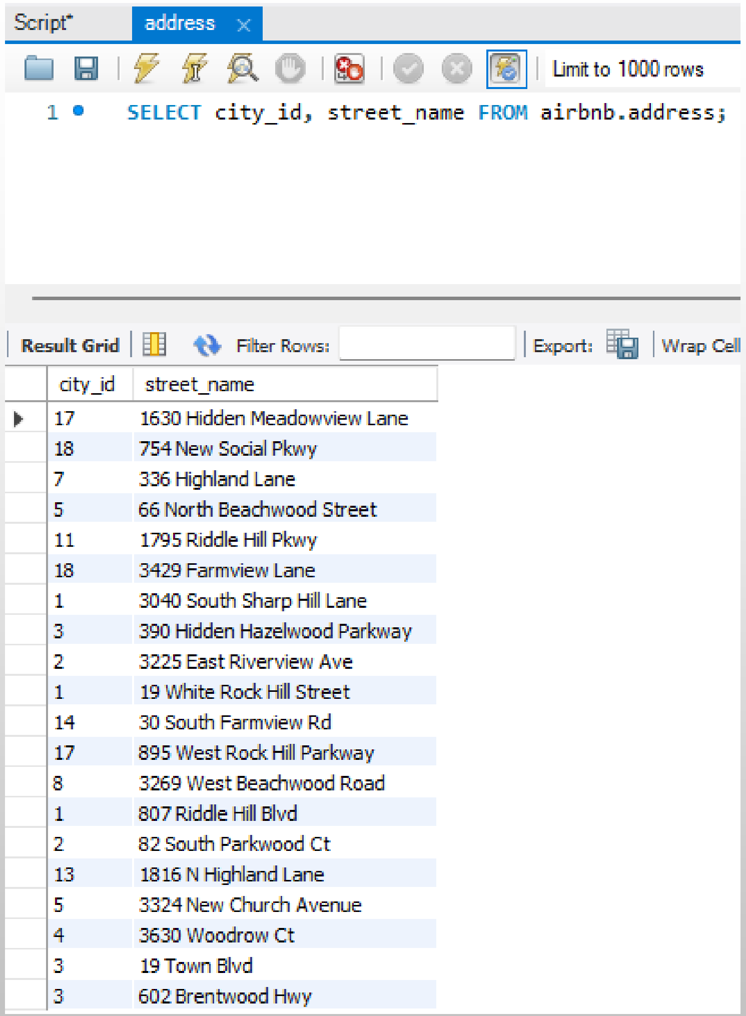
SELECT & FROM function is used to select specific rows and columns from the table that are of interest.

• ADDRESS



```
CREATE TABLE `address` (
  `id` int NOT NULL AUTO_INCREMENT,
  `region_id` int NOT NULL,
  `country_id` int NOT NULL,
  `state_id` int NOT NULL,
  `city_id` int NOT NULL,
  `zip` int NOT NULL,
  `street_name` varchar(45) NOT NULL,
  `latitude` float NOT NULL,
  `longitude` float NOT NULL,
  `created_at` date NOT NULL,
  `updated_at` date NOT NULL,
  PRIMARY KEY (`id`),
  KEY `region_id_idx` (`region_id`),
  KEY `country_id_idx` (`country_id`),
  KEY `state_id_idx` (`state_id`),
  KEY `city_id_idx` (`city_id`),
  CONSTRAINT `city_id` FOREIGN KEY (`city_id`) REFERENCES `city` (`id`),
  CONSTRAINT `country_id` FOREIGN KEY (`country_id`) REFERENCES `country` (`id`),
  CONSTRAINT `region_id` FOREIGN KEY (`region_id`) REFERENCES `region` (`id`),
  CONSTRAINT `state_id` FOREIGN KEY (`state_id`) REFERENCES `state` (`id`)
) ENGINE=InnoDB AUTO_INCREMENT=21 DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4_0900_ai_ci
```

Create table

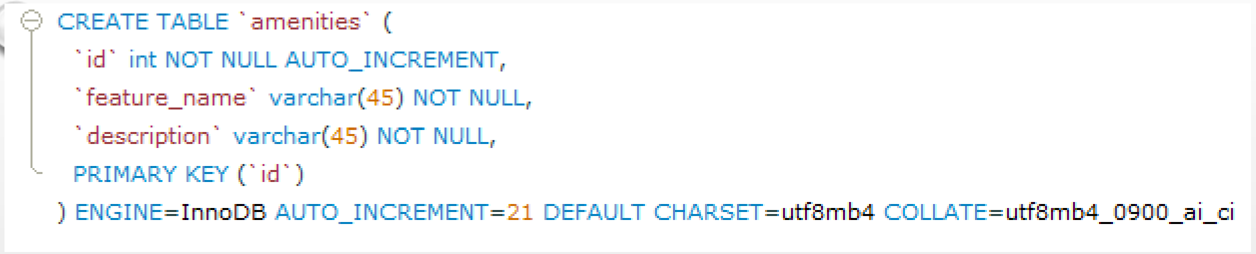


```
SELECT city_id, street_name FROM airbnb.address;
```

city_id	street_name
17	1630 Hidden Meadowview Lane
18	754 New Social Pkwy
7	336 Highland Lane
5	66 North Beachwood Street
11	1795 Riddle Hill Pkwy
18	3429 Farmview Lane
1	3040 South Sharp Hill Lane
3	390 Hidden Hazelwood Parkway
2	3225 East Riverview Ave
1	19 White Rock Hill Street
14	30 South Farmview Rd
17	895 West Rock Hill Parkway
8	3269 West Beachwood Road
1	807 Riddle Hill Blvd
2	82 South Parkwood Ct
13	1816 N Highland Lane
5	3324 New Church Avenue
4	3630 Woodrow Ct
3	19 Town Blvd
3	602 Brentwood Hwy

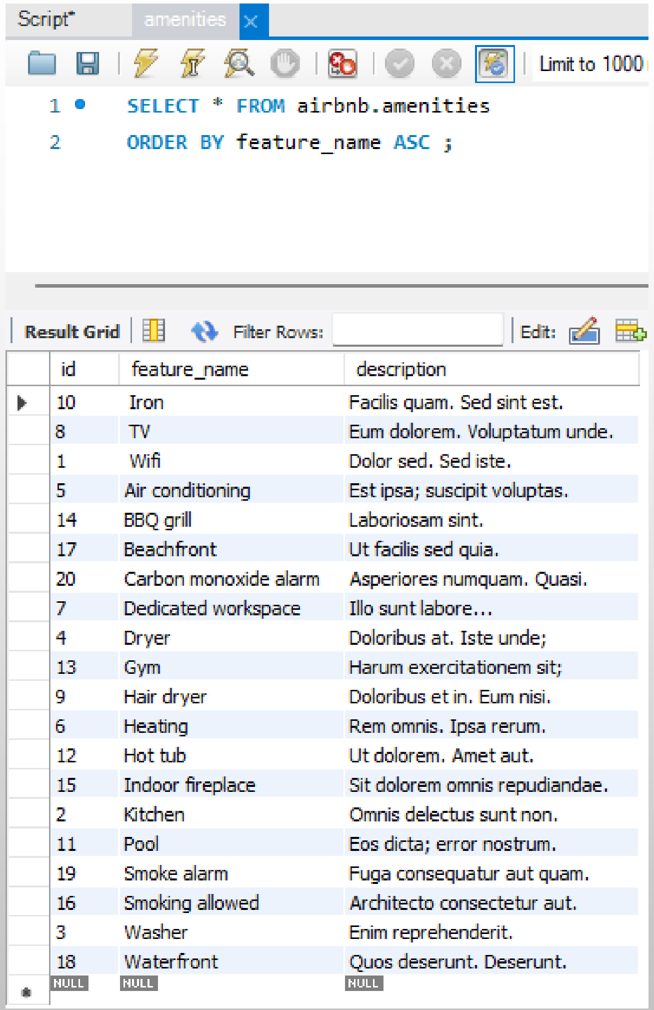
From the SELECT function columns that are to be selected are separated by "," in case it is more than one column.

• AMENITIES



```
CREATE TABLE `amenities` (
  `id` int NOT NULL AUTO_INCREMENT,
  `feature_name` varchar(45) NOT NULL,
  `description` varchar(45) NOT NULL,
  PRIMARY KEY (`id`)
) ENGINE=InnoDB AUTO_INCREMENT=21 DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4_0900_ai_ci
```

Create table

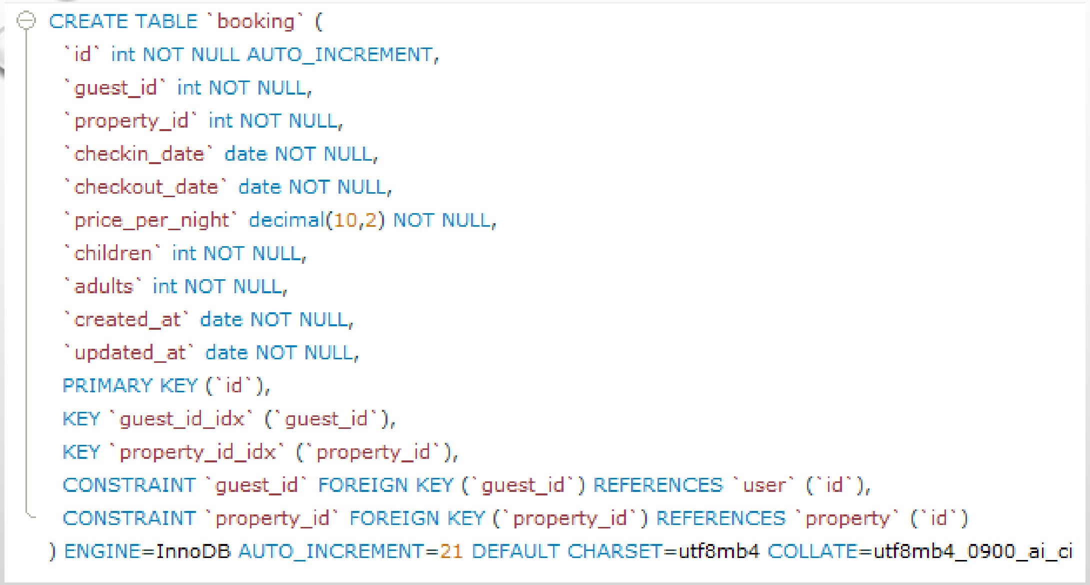


```
SELECT * FROM airbnb.amenities
ORDER BY feature_name ASC ;
```

id	feature_name	description
10	Iron	Facilis quam. Sed sint est.
8	TV	Eum dolorem. Voluptatum unde.
1	Wifi	Dolor sed. Sed iste.
5	Air conditioning	Est ipsa; suscipit voluptas.
14	BBQ grill	Laboriosam sint.
17	Beachfront	Ut facilis sed quia.
20	Carbon monoxide alarm	Asperiores numquam. Quasi.
7	Dedicated workspace	Illo sunt labore...
4	Dryer	Doloribus at. Iste unde;
13	Gym	Harum exercitationem sit;
9	Hair dryer	Doloribus et in. Eum nisi.
6	Heating	Rem omnis. Ipsa rerum.
12	Hot tub	Ut dolorem. Amet aut.
15	Indoor fireplace	Sit dolorem omnis repudiandae.
2	Kitchen	Omnis delectus sunt non.
11	Pool	Eos dicta; error nostrum.
19	Smoke alarm	Fuga consequatur aut quam.
16	Smoking allowed	Architecto consectetur aut.
3	Washer	Enim reprehenderit.
18	Waterfront	Quos deserunt. Deserunt.
NULL	NULL	NULL

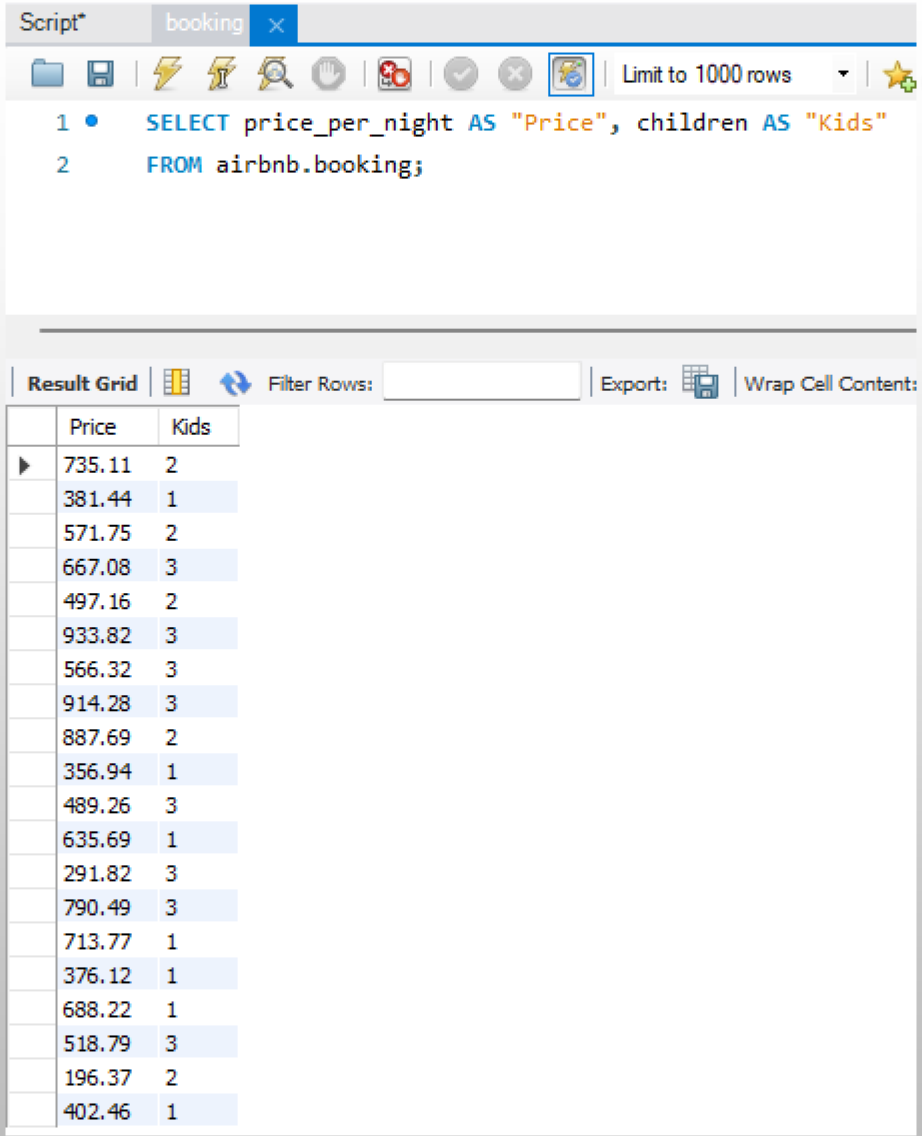
We can use function ORDER BY to query results of a column to arrange them with Ascending order.

- **BOOKING**



```
CREATE TABLE `booking` (
  `id` int NOT NULL AUTO_INCREMENT,
  `guest_id` int NOT NULL,
  `property_id` int NOT NULL,
  `checkin_date` date NOT NULL,
  `checkout_date` date NOT NULL,
  `price_per_night` decimal(10,2) NOT NULL,
  `children` int NOT NULL,
  `adults` int NOT NULL,
  `created_at` date NOT NULL,
  `updated_at` date NOT NULL,
  PRIMARY KEY (`id`),
  KEY `guest_id_idx` (`guest_id`),
  KEY `property_id_idx` (`property_id`),
  CONSTRAINT `guest_id` FOREIGN KEY (`guest_id`) REFERENCES `user` (`id`),
  CONSTRAINT `property_id` FOREIGN KEY (`property_id`) REFERENCES `property` (`id`)
) ENGINE=InnoDB AUTO_INCREMENT=21 DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4_0900_ai_ci
```

Create table

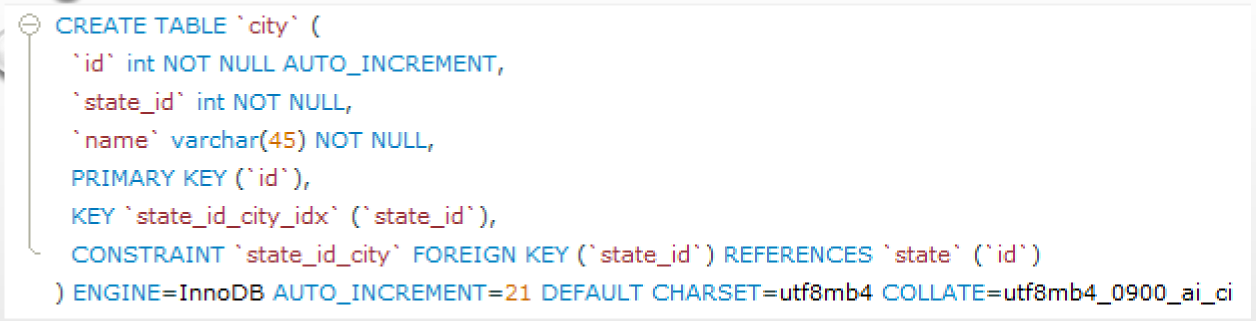


```
SELECT price_per_night AS "Price", children AS "Kids"
FROM airbnb.booking;
```

Price	Kids
735.11	2
381.44	1
571.75	2
667.08	3
497.16	2
933.82	3
566.32	3
914.28	3
887.69	2
356.94	1
489.26	3
635.69	1
291.82	3
790.49	3
713.77	1
376.12	1
688.22	1
518.79	3
196.37	2
402.46	1

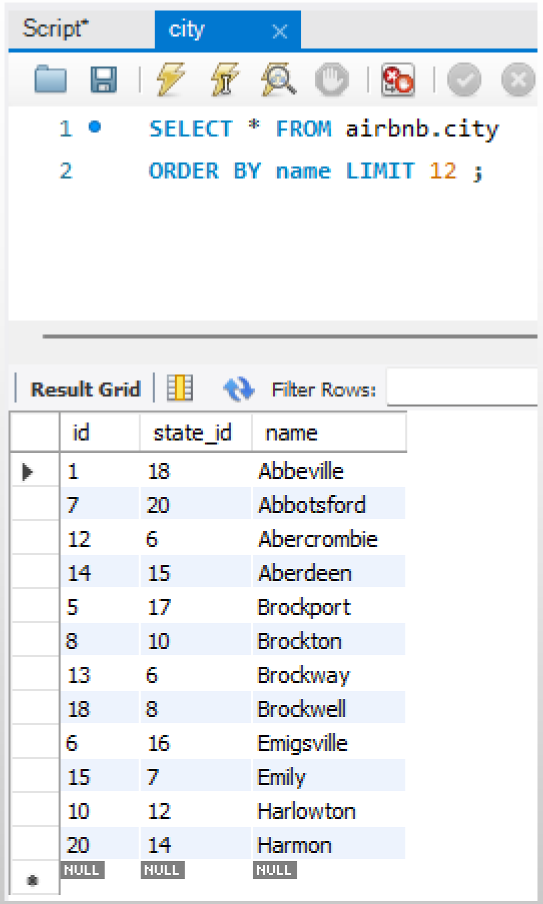
We can use the AS functions to display our columns as the names we prefer.

- **CITY**



```
CREATE TABLE `city` (
  `id` int NOT NULL AUTO_INCREMENT,
  `state_id` int NOT NULL,
  `name` varchar(45) NOT NULL,
  PRIMARY KEY (`id`),
  KEY `state_id_city_idx` (`state_id`),
  CONSTRAINT `state_id_city` FOREIGN KEY (`state_id`) REFERENCES `state` (`id`)
) ENGINE=InnoDB AUTO_INCREMENT=21 DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4_0900_ai_ci
```

Create table

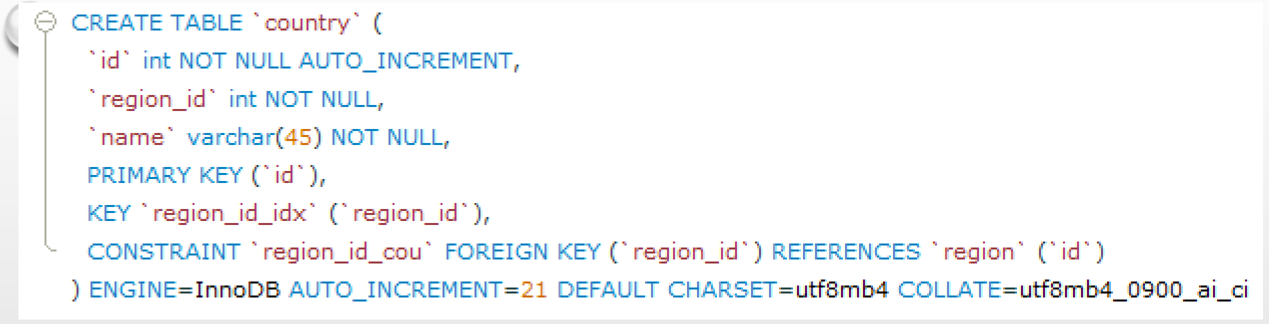


```
SELECT * FROM airbnb.city
ORDER BY name LIMIT 12 ;
```

id	state_id	name
1	18	Abbeville
7	20	Abbotsford
12	6	Abercrombie
14	15	Aberdeen
5	17	Brockport
8	10	Brockton
13	6	Brockway
18	8	Brockwell
6	16	Emigsville
15	7	Emily
10	12	Harlowton
20	14	Harmon
NULL	NULL	NULL

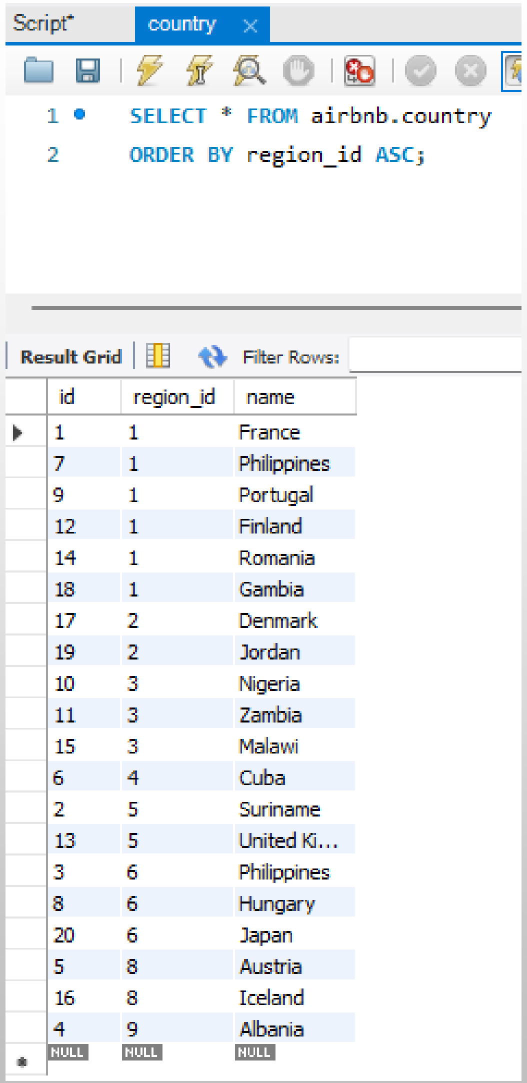
We can set the limit of rows to be displayed after sorting them by ascending order.

• COUNTRY



```
CREATE TABLE `country` (
  `id` int NOT NULL AUTO_INCREMENT,
  `region_id` int NOT NULL,
  `name` varchar(45) NOT NULL,
  PRIMARY KEY (`id`),
  KEY `region_id_idx` (`region_id`),
  CONSTRAINT `region_id_cou` FOREIGN KEY (`region_id`) REFERENCES `region` (`id`)
) ENGINE=InnoDB AUTO_INCREMENT=21 DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4_0900_ai_ci
```

Create table

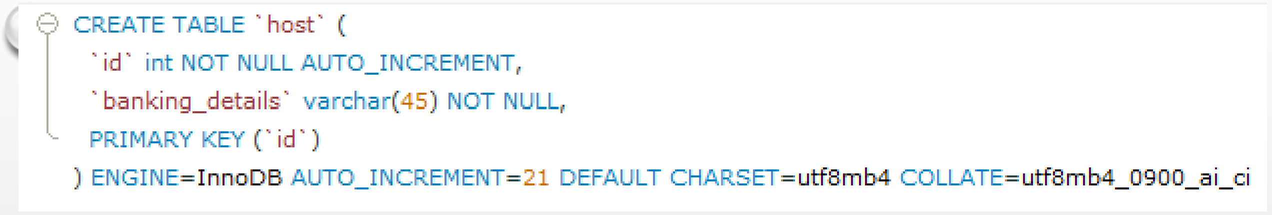


```
SELECT * FROM airbnb.country
ORDER BY region_id ASC;
```

id	region_id	name
1	1	France
7	1	Philippines
9	1	Portugal
12	1	Finland
14	1	Romania
18	1	Gambia
17	2	Denmark
19	2	Jordan
10	3	Nigeria
11	3	Zambia
15	3	Malawi
6	4	Cuba
2	5	Suriname
13	5	United Ki...
3	6	Philippines
8	6	Hungary
20	6	Japan
5	8	Austria
16	8	Iceland
4	9	Albania
NULL	NULL	NULL

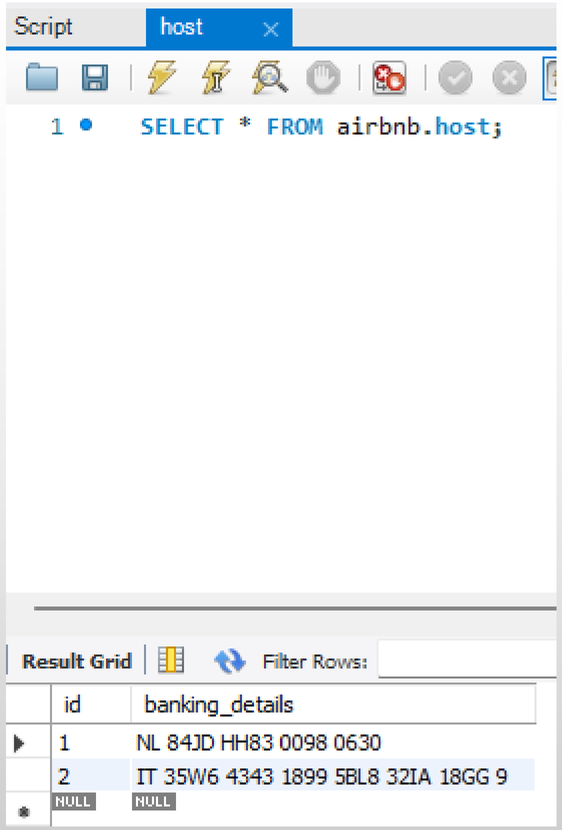
Names of countries that are available on the address entity.

- **HOST**



```
CREATE TABLE `host` (
  `id` int NOT NULL AUTO_INCREMENT,
  `banking_details` varchar(45) NOT NULL,
  PRIMARY KEY (`id`)
) ENGINE=InnoDB AUTO_INCREMENT=21 DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4_0900_ai_ci
```

Create table

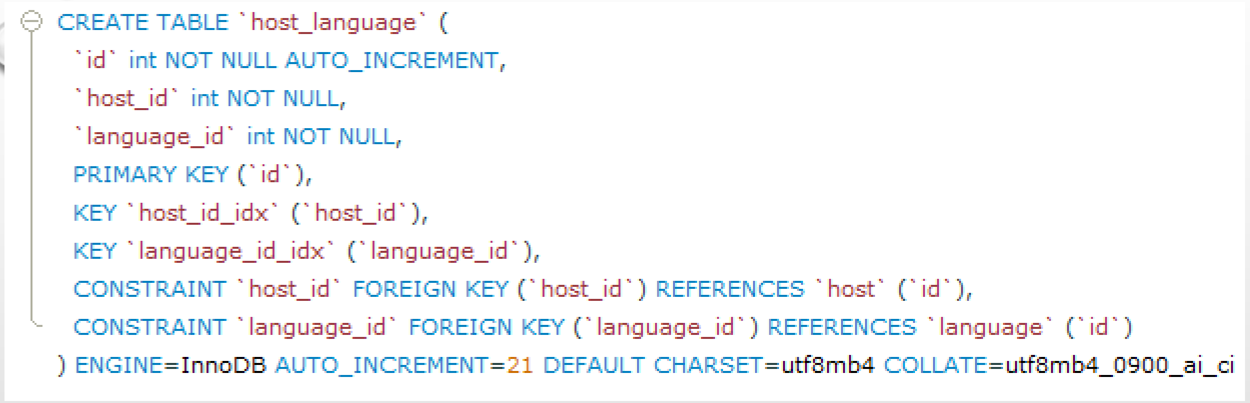


```
SELECT * FROM airbnb.host;
```

```
   id    banking_details
▶  1     NL 84JD HH83 0098 0630
   2     IT 35W6 4343 1899 5BL8 32IA 18GG 9
*  NULL  NULL
```

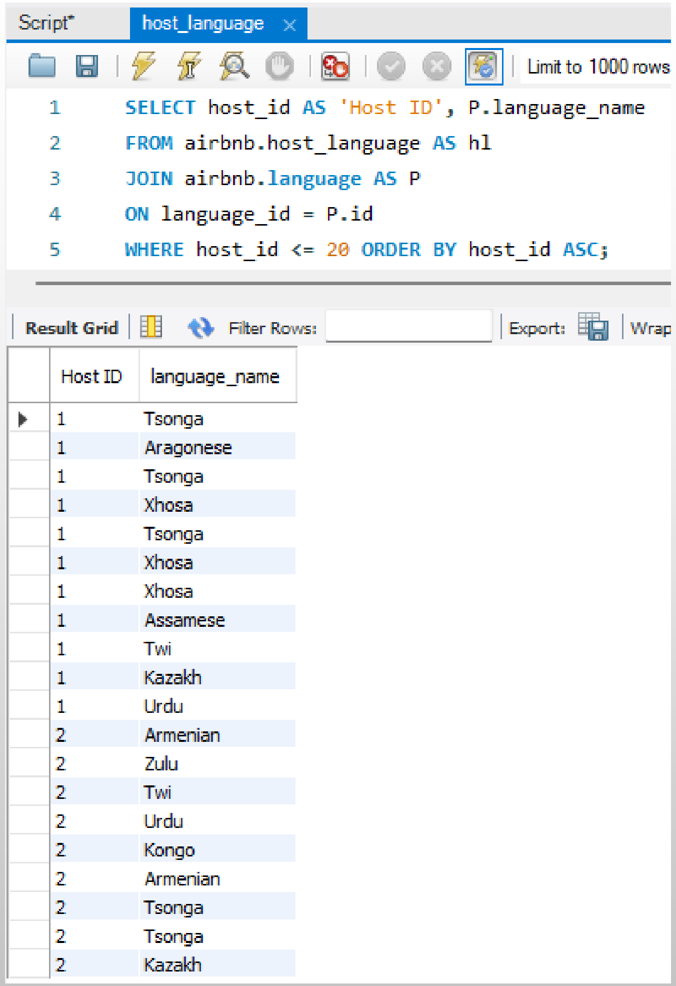
The results of the hosts of the properties.

• HOST LANGUAGE



```
CREATE TABLE `host_language` (
  `id` int NOT NULL AUTO_INCREMENT,
  `host_id` int NOT NULL,
  `language_id` int NOT NULL,
  PRIMARY KEY (`id`),
  KEY `host_id_idx` (`host_id`),
  KEY `language_id_idx` (`language_id`),
  CONSTRAINT `host_id` FOREIGN KEY (`host_id`) REFERENCES `host` (`id`),
  CONSTRAINT `language_id` FOREIGN KEY (`language_id`) REFERENCES `language` (`id`)
) ENGINE=InnoDB AUTO_INCREMENT=21 DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4_0900_ai_ci
```

Create table

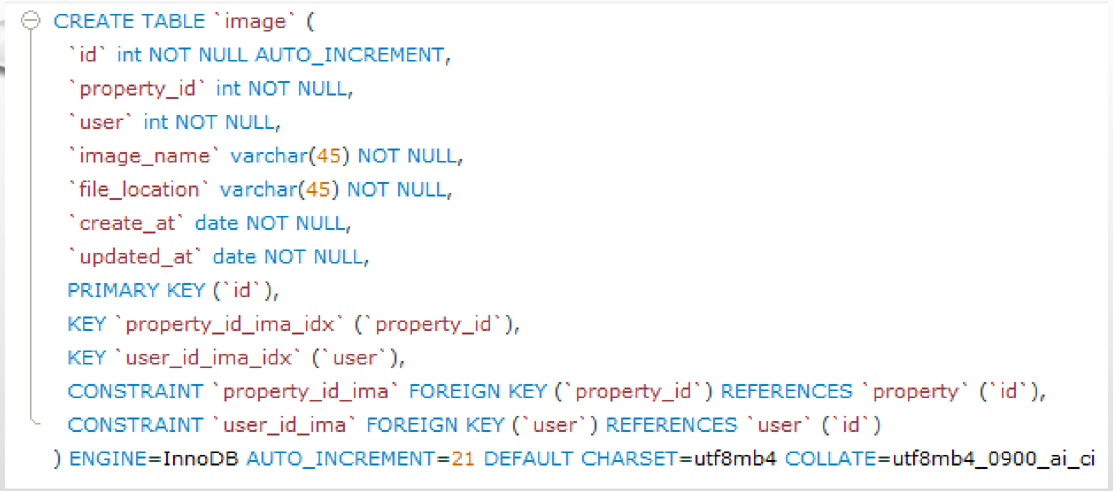


```
SELECT host_id AS 'Host ID', P.language_name
FROM airbnb.host_language AS hl
JOIN airbnb.language AS P
ON language_id = P.id
WHERE host_id <= 20 ORDER BY host_id ASC;
```

Host ID	language_name
1	Tsonga
1	Aragonese
1	Tsonga
1	Xhosa
1	Tsonga
1	Xhosa
1	Xhosa
1	Assamese
1	Twi
1	Kazakh
1	Urdu
2	Armenian
2	Zulu
2	Twi
2	Urdu
2	Kongo
2	Armenian
2	Tsonga
2	Tsonga
2	Kazakh

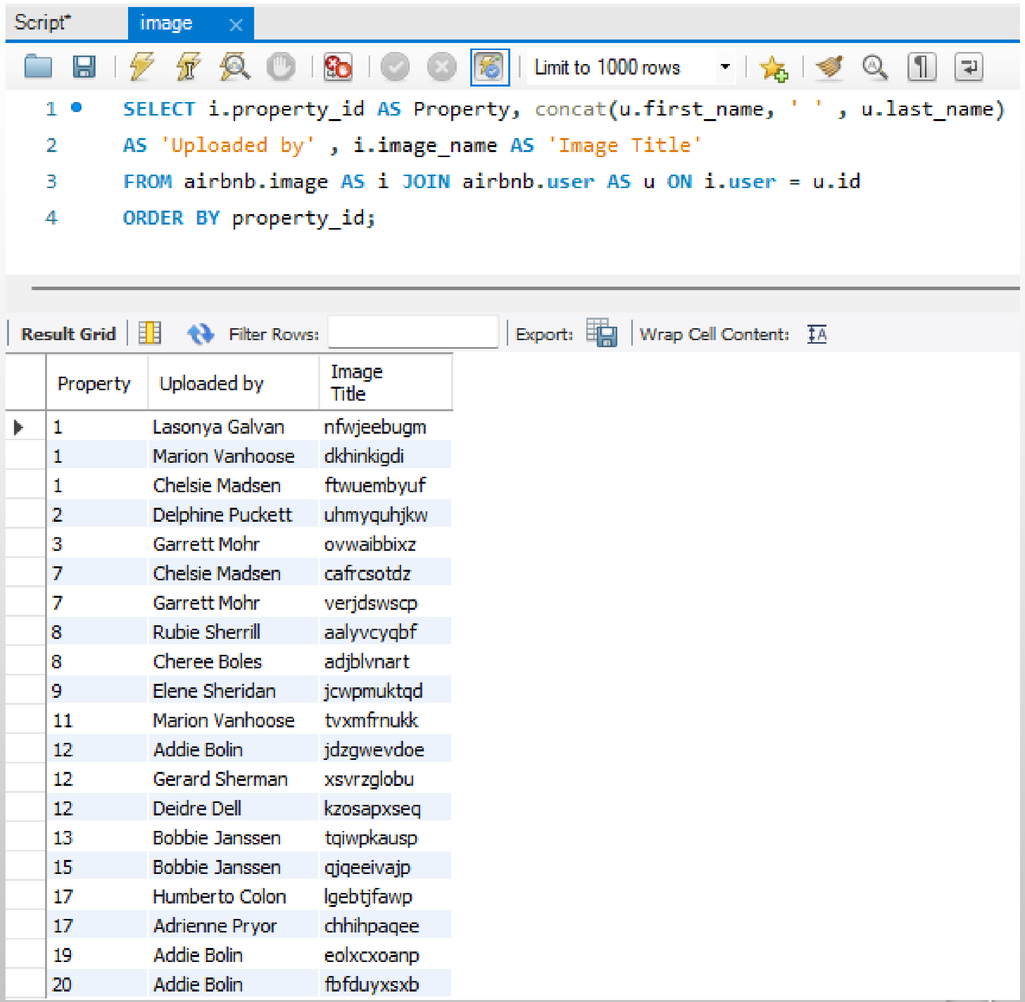
The table for the language is used to join with the host language so that we can display which host speaks which languages.

- **IMAGE**



```
CREATE TABLE `image` (
  `id` int NOT NULL AUTO_INCREMENT,
  `property_id` int NOT NULL,
  `user` int NOT NULL,
  `image_name` varchar(45) NOT NULL,
  `file_location` varchar(45) NOT NULL,
  `create_at` date NOT NULL,
  `updated_at` date NOT NULL,
  PRIMARY KEY (`id`),
  KEY `property_id_ima_idx` (`property_id`),
  KEY `user_id_ima_idx` (`user`),
  CONSTRAINT `property_id_ima` FOREIGN KEY (`property_id`) REFERENCES `property` (`id`),
  CONSTRAINT `user_id_ima` FOREIGN KEY (`user`) REFERENCES `user` (`id`)
) ENGINE=InnoDB AUTO_INCREMENT=21 DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4_0900_ai_ci
```

Create table

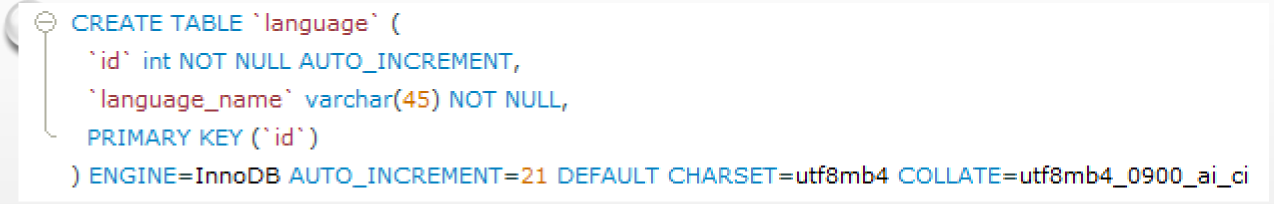


```
SELECT i.property_id AS Property, concat(u.first_name, ' ' , u.last_name)
AS 'Uploaded by' , i.image_name AS 'Image Title'
FROM airbnb.image AS i JOIN airbnb.user AS u ON i.user = u.id
ORDER BY property_id;
```

Property	Uploaded by	Image Title
1	Lasonya Galvan	nfwjeebugm
1	Marion Vanhoose	dkhinkigdi
1	Chelsie Madsen	ftwuembyuf
2	Delphine Puckett	uhmyquhjkw
3	Garrett Mohr	ovwaibbixz
7	Chelsie Madsen	cafrcsotdz
7	Garrett Mohr	verjdswscp
8	Rubie Sherrill	aalyvcyqbf
8	Cheree Boles	adjblvnart
9	Elene Sheridan	jcwpmuktqd
11	Marion Vanhoose	tvxmfrnukk
12	Addie Bolin	jdzgwevdoe
12	Gerard Sherman	xsvrzglobu
12	Deidre Dell	kzosapxseq
13	Bobbie Janssen	tqiwpkausp
15	Bobbie Janssen	qjqeeivajp
17	Humberto Colon	lgebtjfawp
17	Adrienne Pryor	chhihpaqee
19	Addie Bolin	eolxcxoanp
20	Addie Bolin	fbfduyxsxb

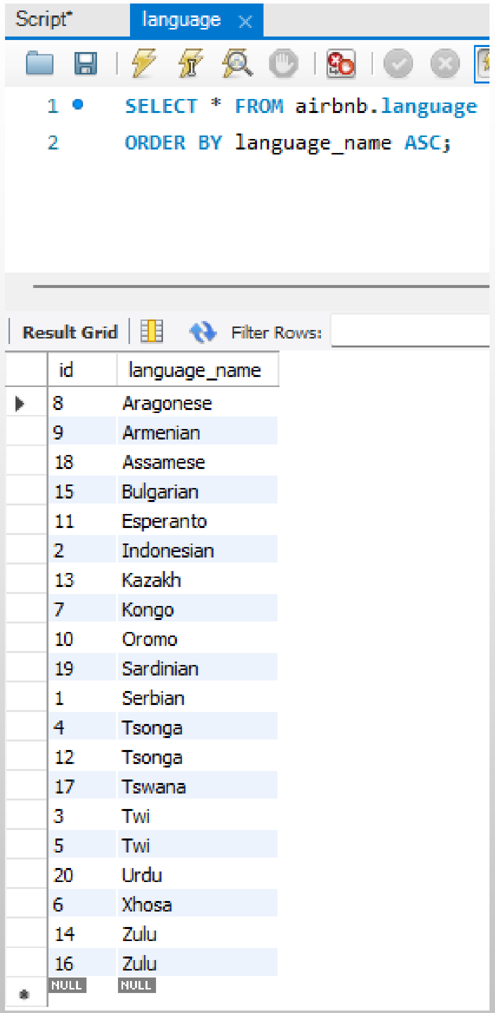
We can use the JOIN function to join a related table (user) to extract the names and use the Concatenate function to merge the first and last name.

• LANGUAGE



```
CREATE TABLE `language` (
  `id` int NOT NULL AUTO_INCREMENT,
  `language_name` varchar(45) NOT NULL,
  PRIMARY KEY (`id`)
) ENGINE=InnoDB AUTO_INCREMENT=21 DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4_0900_ai_ci
```

Create table

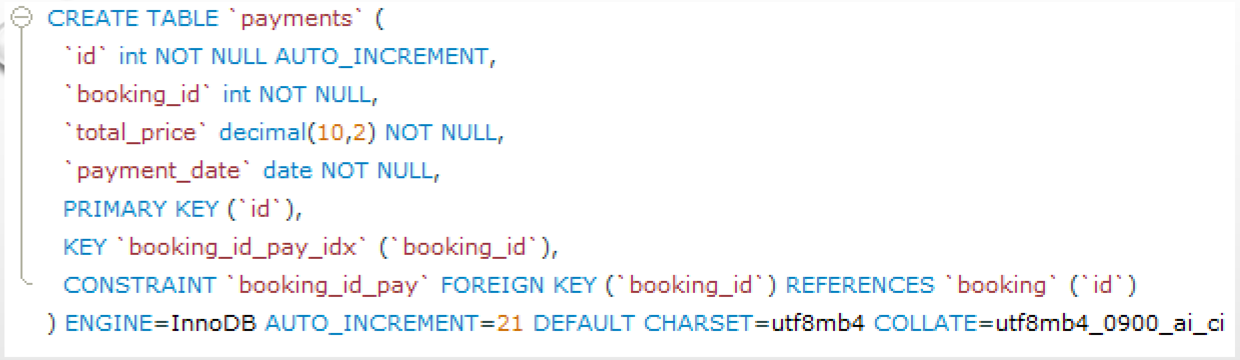


```
SELECT * FROM airbnb.language
ORDER BY language_name ASC;
```

id	language_name
8	Aragonese
9	Armenian
18	Assamese
15	Bulgarian
11	Esperanto
2	Indonesian
13	Kazakh
7	Kongo
10	Oromo
19	Sardinian
1	Serbian
4	Tsonga
12	Tsonga
17	Tswana
3	Twi
5	Twi
20	Urdu
6	Xhosa
14	Zulu
16	Zulu
NULL	NULL

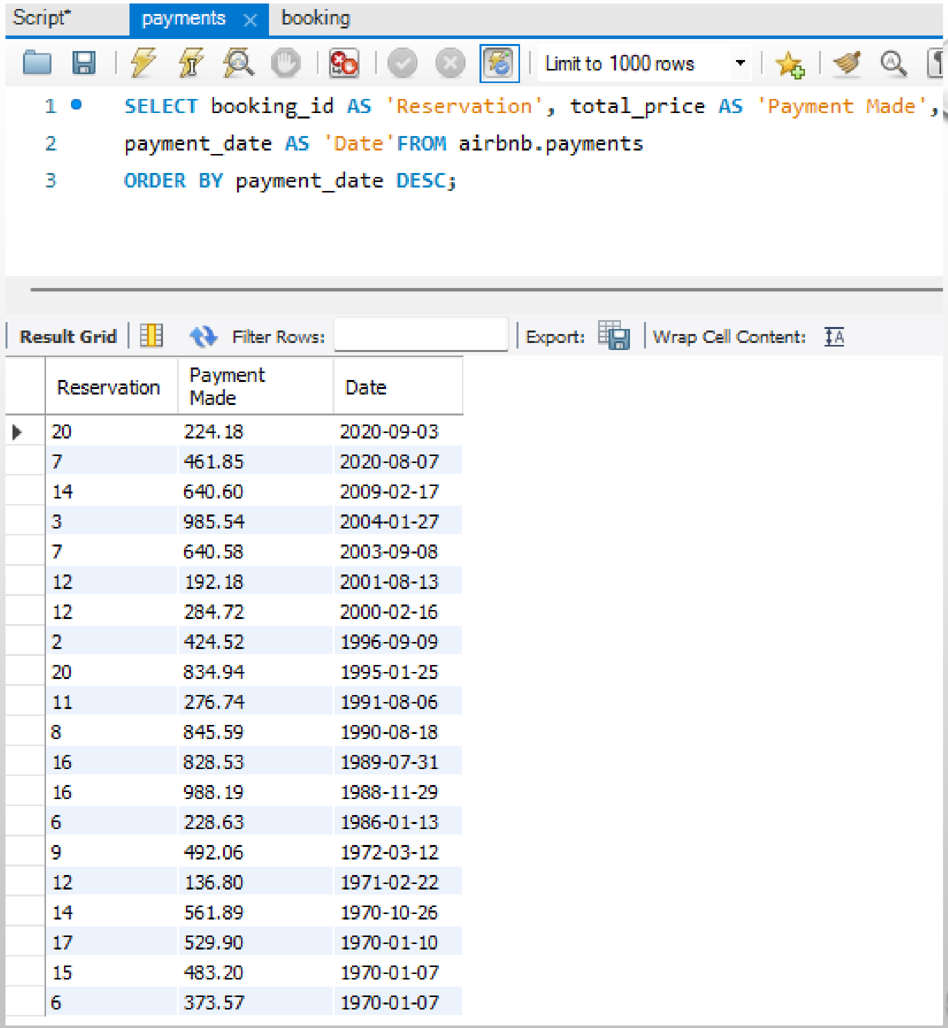
We can display the languages that are spoken by the Hosts then arrange them in ascending order for easy readability.

- **PAYMENT**



```
CREATE TABLE `payments` (
  `id` int NOT NULL AUTO_INCREMENT,
  `booking_id` int NOT NULL,
  `total_price` decimal(10,2) NOT NULL,
  `payment_date` date NOT NULL,
  PRIMARY KEY (`id`),
  KEY `booking_id_pay_idx` (`booking_id`),
  CONSTRAINT `booking_id_pay` FOREIGN KEY (`booking_id`) REFERENCES `booking` (`id`)
) ENGINE=InnoDB AUTO_INCREMENT=21 DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4_0900_ai_ci
```

Create table

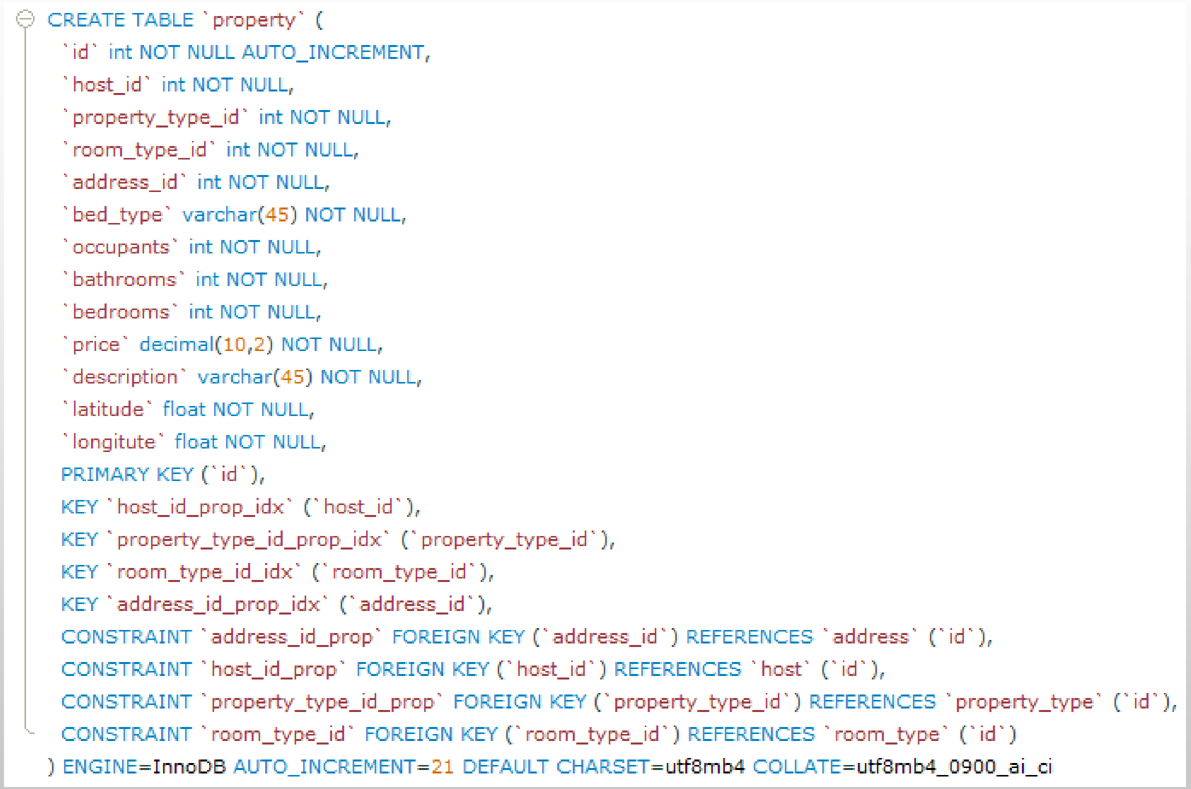


```
SELECT booking_id AS 'Reservation', total_price AS 'Payment Made',
payment_date AS 'Date'FROM airbnb.payments
ORDER BY payment_date DESC;
```

Reservation	Payment Made	Date
20	224.18	2020-09-03
7	461.85	2020-08-07
14	640.60	2009-02-17
3	985.54	2004-01-27
7	640.58	2003-09-08
12	192.18	2001-08-13
12	284.72	2000-02-16
2	424.52	1996-09-09
20	834.94	1995-01-25
11	276.74	1991-08-06
8	845.59	1990-08-18
16	828.53	1989-07-31
16	988.19	1988-11-29
6	228.63	1986-01-13
9	492.06	1972-03-12
12	136.80	1971-02-22
14	561.89	1970-10-26
17	529.90	1970-01-10
15	483.20	1970-01-07
6	373.57	1970-01-07

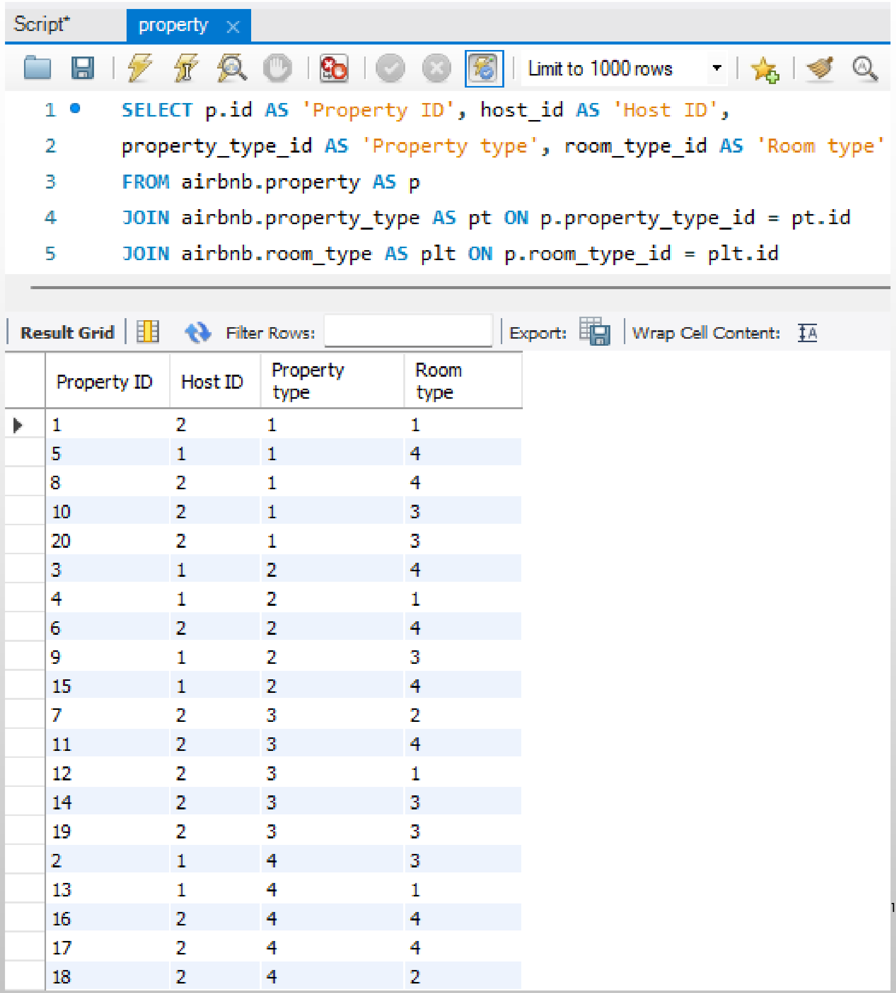
For easy record accessing via date of payment we can use the ORDER BY function to sort the results by date starting with the latest payment date.

- **PROPERTY**



```
CREATE TABLE `property` (
  `id` int NOT NULL AUTO_INCREMENT,
  `host_id` int NOT NULL,
  `property_type_id` int NOT NULL,
  `room_type_id` int NOT NULL,
  `address_id` int NOT NULL,
  `bed_type` varchar(45) NOT NULL,
  `occupants` int NOT NULL,
  `bathrooms` int NOT NULL,
  `bedrooms` int NOT NULL,
  `price` decimal(10,2) NOT NULL,
  `description` varchar(45) NOT NULL,
  `latitude` float NOT NULL,
  `longitude` float NOT NULL,
  PRIMARY KEY (`id`),
  KEY `host_id_prop_idx` (`host_id`),
  KEY `property_type_id_prop_idx` (`property_type_id`),
  KEY `room_type_id_idx` (`room_type_id`),
  KEY `address_id_prop_idx` (`address_id`),
  CONSTRAINT `address_id_prop` FOREIGN KEY (`address_id`) REFERENCES `address` (`id`),
  CONSTRAINT `host_id_prop` FOREIGN KEY (`host_id`) REFERENCES `host` (`id`),
  CONSTRAINT `property_type_id_prop` FOREIGN KEY (`property_type_id`) REFERENCES `property_type` (`id`),
  CONSTRAINT `room_type_id` FOREIGN KEY (`room_type_id`) REFERENCES `room_type` (`id`)
) ENGINE=InnoDB AUTO_INCREMENT=21 DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4_0900_ai_ci
```

Create table

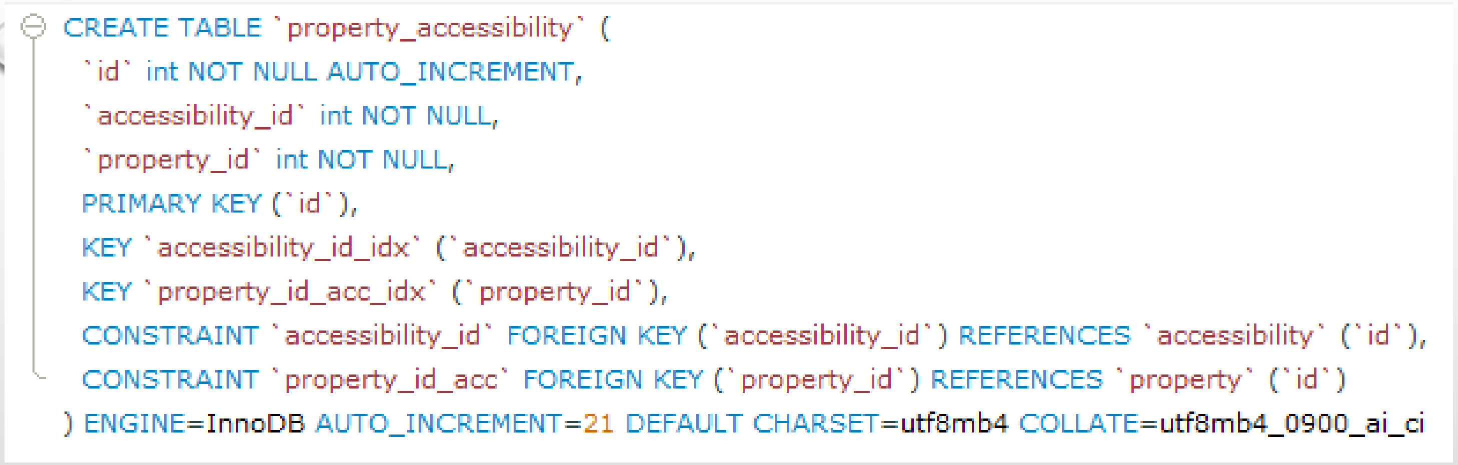


```sql
SELECT p.id AS 'Property ID', host_id AS 'Host ID',
property_type_id AS 'Property type', room_type_id AS 'Room type'
FROM airbnb.property AS p
JOIN airbnb.property_type AS pt ON p.property_type_id = pt.id
JOIN airbnb.room_type AS plt ON p.room_type_id = plt.id
```

Property ID	Host ID	Property type	Room type
1	2	1	1
5	1	1	4
8	2	1	4
10	2	1	3
20	2	1	3
3	1	2	4
4	1	2	1
6	2	2	4
9	1	2	3
15	1	2	4
7	2	3	2
11	2	3	4
12	2	3	1
14	2	3	3
19	2	3	3
2	1	4	3
13	1	4	1
16	2	4	4
17	2	4	4
18	2	4	2

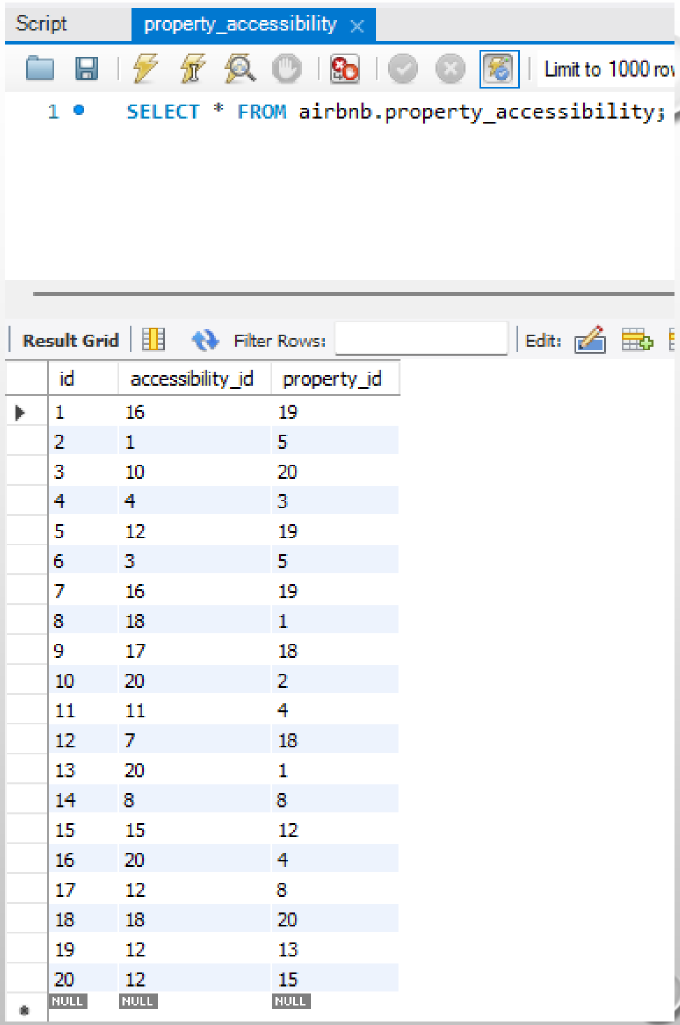
Property is filter to show results of the host and what conditions or what kind of properties do they have.

• PAYMENT ACCESSIBILITY



```sql
CREATE TABLE `property_accessibility` (
  `id` int NOT NULL AUTO_INCREMENT,
  `accessibility_id` int NOT NULL,
  `property_id` int NOT NULL,
  PRIMARY KEY (`id`),
  KEY `accessibility_id_idx` (`accessibility_id`),
  KEY `property_id_acc_idx` (`property_id`),
  CONSTRAINT `accessibility_id` FOREIGN KEY (`accessibility_id`) REFERENCES `accessibility` (`id`),
  CONSTRAINT `property_id_acc` FOREIGN KEY (`property_id`) REFERENCES `property` (`id`)
) ENGINE=InnoDB AUTO_INCREMENT=21 DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4_0900_ai_ci
```

Create table

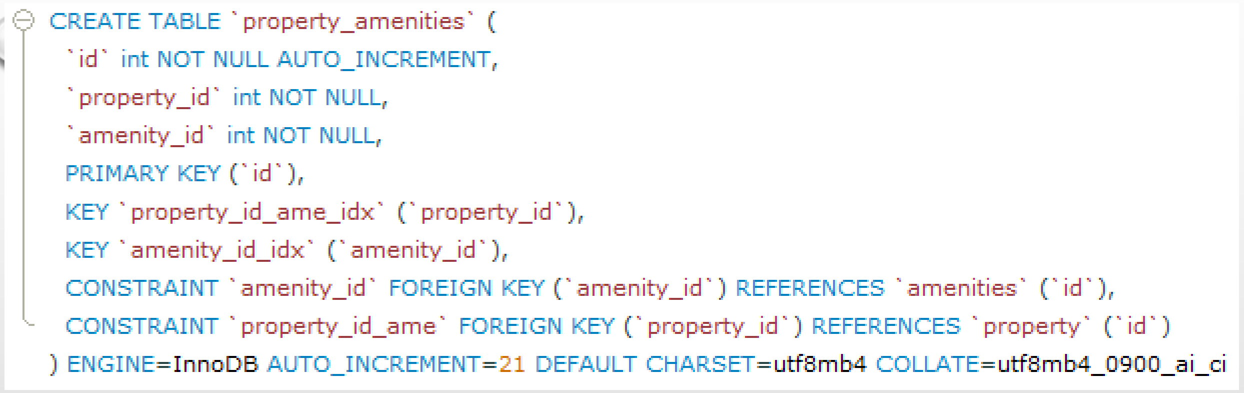


```sql
SELECT * FROM airbnb.property_accessibility;
```

id	accessibility_id	property_id
1	16	19
2	1	5
3	10	20
4	4	3
5	12	19
6	3	5
7	16	19
8	18	1
9	17	18
10	20	2
11	11	4
12	7	18
13	20	1
14	8	8
15	15	12
16	20	4
17	12	8
18	18	20
19	12	13
20	12	15
NULL	NULL	NULL

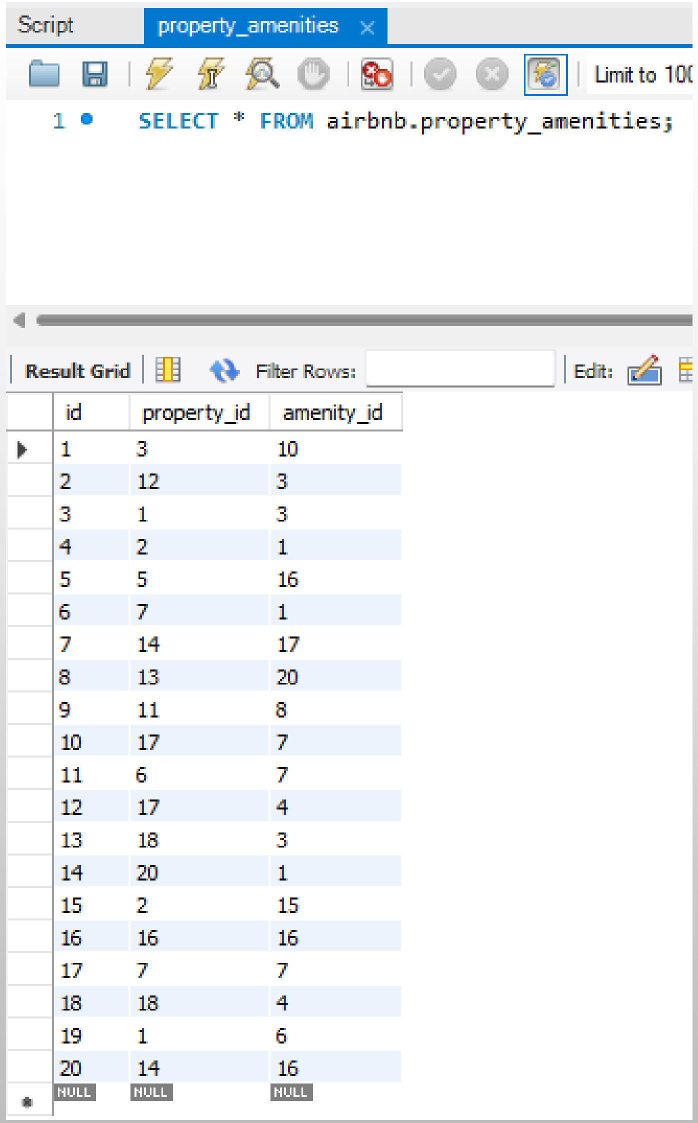
The property and what accessibility it does have so that we can know also what guests are drawn to.

- **PROPERTY AMENITIES**



```
CREATE TABLE `property_amenities` (
  `id` int NOT NULL AUTO_INCREMENT,
  `property_id` int NOT NULL,
  `amenity_id` int NOT NULL,
  PRIMARY KEY (`id`),
  KEY `property_id_ame_idx` (`property_id`),
  KEY `amenity_id_idx` (`amenity_id`),
  CONSTRAINT `amenity_id` FOREIGN KEY (`amenity_id`) REFERENCES `amenities` (`id`),
  CONSTRAINT `property_id_ame` FOREIGN KEY (`property_id`) REFERENCES `property` (`id`)
) ENGINE=InnoDB AUTO_INCREMENT=21 DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4_0900_ai_ci
```

Create table

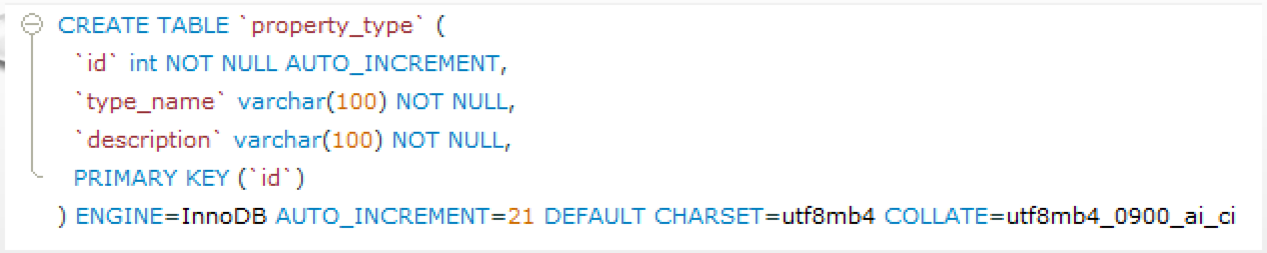


```
SELECT * FROM airbnb.property_amenities;
```

id	property_id	amenity_id
1	3	10
2	12	3
3	1	3
4	2	1
5	5	16
6	7	1
7	14	17
8	13	20
9	11	8
10	17	7
11	6	7
12	17	4
13	18	3
14	20	1
15	2	15
16	16	16
17	7	7
18	18	4
19	1	6
20	14	16
NULL	NULL	NULL

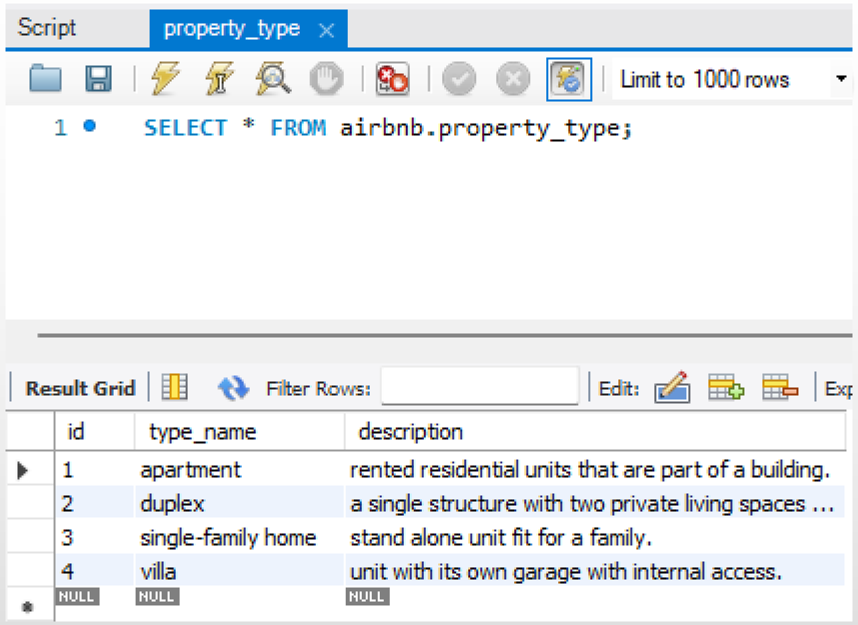
Each property is associated with an amenity that is displayed as results.

• PROPERTY TYPE



```
CREATE TABLE `property_type` (
  `id` int NOT NULL AUTO_INCREMENT,
  `type_name` varchar(100) NOT NULL,
  `description` varchar(100) NOT NULL,
  PRIMARY KEY (`id`)
) ENGINE=InnoDB AUTO_INCREMENT=21 DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4_0900_ai_ci
```

Create table

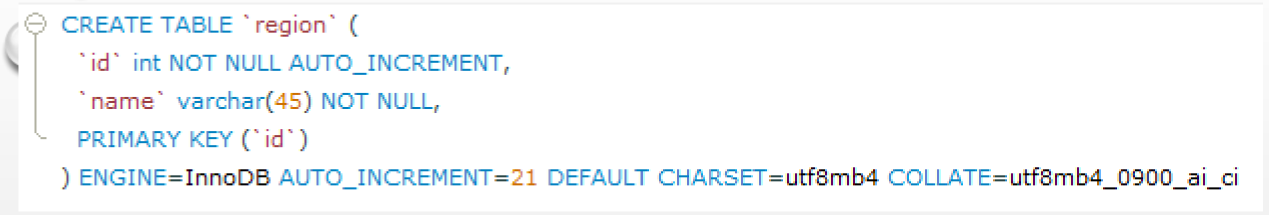


```
SELECT * FROM airbnb.property_type;
```

id	type_name	description
1	apartment	rented residential units that are part of a building.
2	duplex	a single structure with two private living spaces ...
3	single-family home	stand alone unit fit for a family.
4	villa	unit with its own garage with internal access.
NULL	NULL	NULL

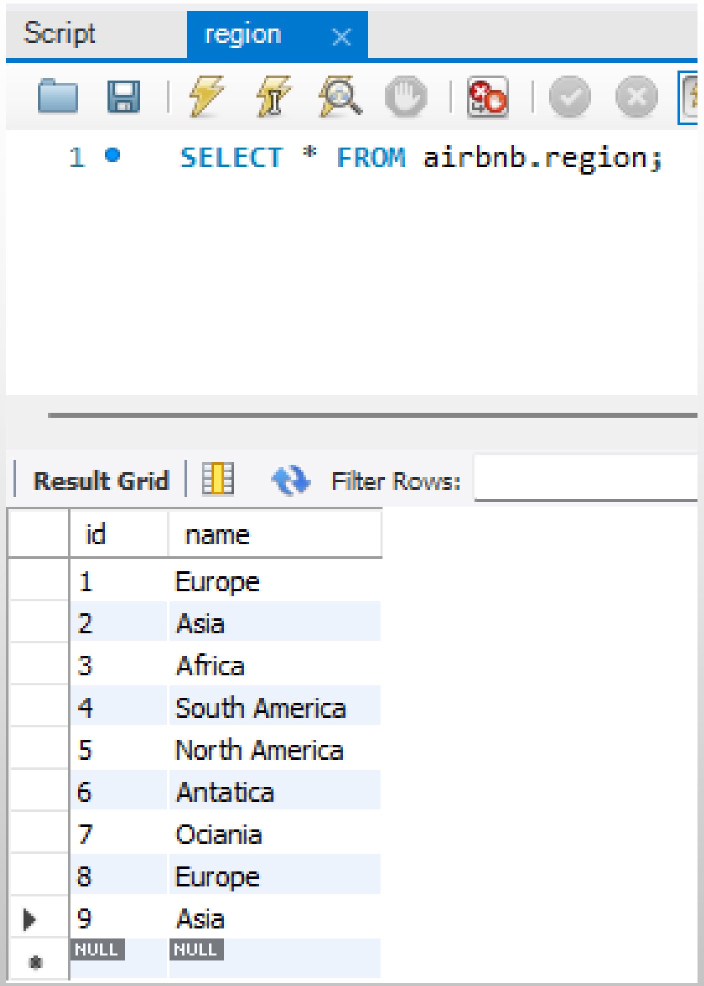
Results show the type of property offered by the host.

- **REGION**



```
CREATE TABLE `region` (
  `id` int NOT NULL AUTO_INCREMENT,
  `name` varchar(45) NOT NULL,
  PRIMARY KEY (`id`)
) ENGINE=InnoDB AUTO_INCREMENT=21 DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4_0900_ai_ci
```

Create table

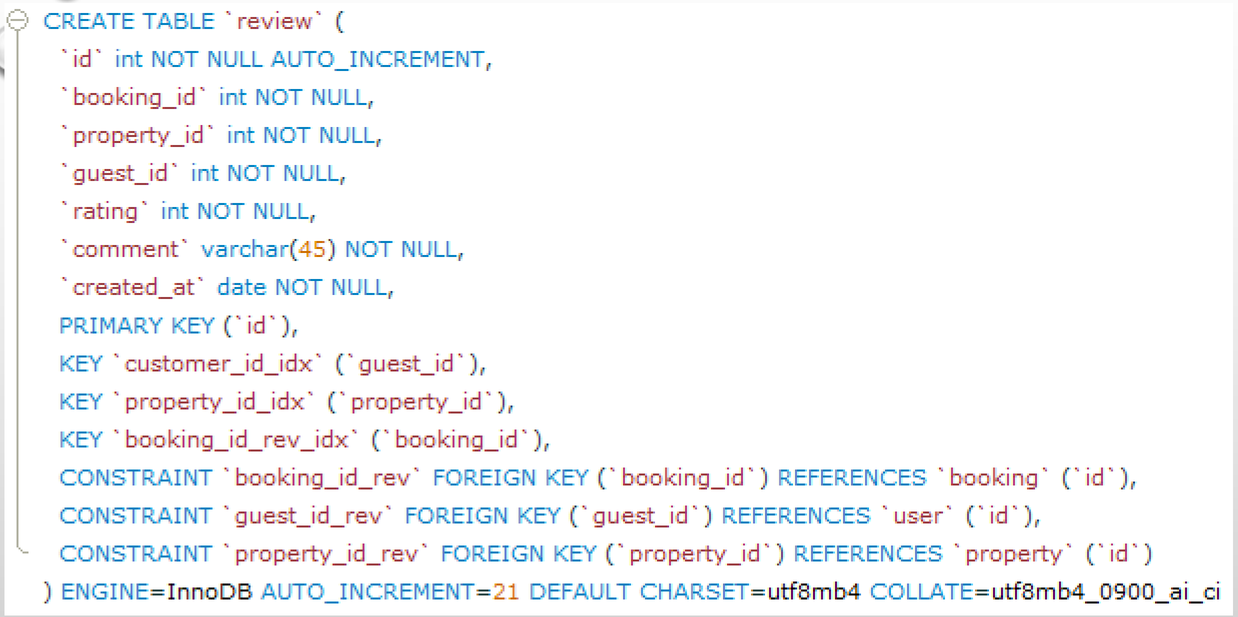


```
SELECT * FROM airbnb.region;
```

id	name
1	Europe
2	Asia
3	Africa
4	South America
5	North America
6	Antatica
7	Ociania
8	Europe
9	Asia
NULL	NULL

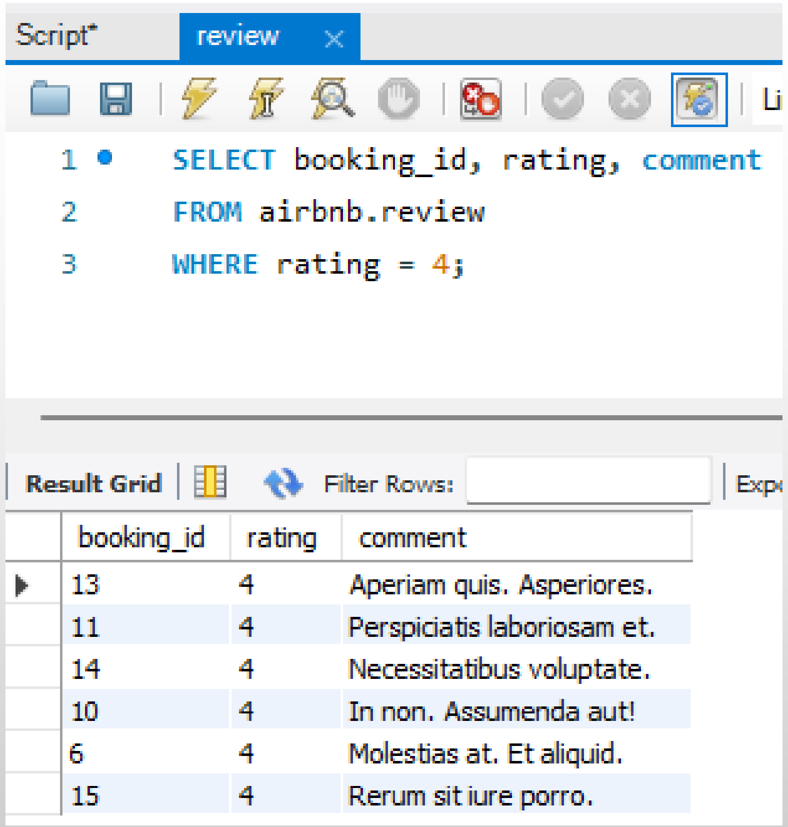
A selection for the different regions that properties can be booked by guests.

• REVIEW



```
CREATE TABLE `review` (
  `id` int NOT NULL AUTO_INCREMENT,
  `booking_id` int NOT NULL,
  `property_id` int NOT NULL,
  `guest_id` int NOT NULL,
  `rating` int NOT NULL,
  `comment` varchar(45) NOT NULL,
  `created_at` date NOT NULL,
  PRIMARY KEY (`id`),
  KEY `customer_id_idx` (`guest_id`),
  KEY `property_id_idx` (`property_id`),
  KEY `booking_id_rev_idx` (`booking_id`),
  CONSTRAINT `booking_id_rev` FOREIGN KEY (`booking_id`) REFERENCES `booking` (`id`),
  CONSTRAINT `guest_id_rev` FOREIGN KEY (`guest_id`) REFERENCES `user` (`id`),
  CONSTRAINT `property_id_rev` FOREIGN KEY (`property_id`) REFERENCES `property` (`id`)
) ENGINE=InnoDB AUTO_INCREMENT=21 DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4_0900_ai_ci
```

Create table

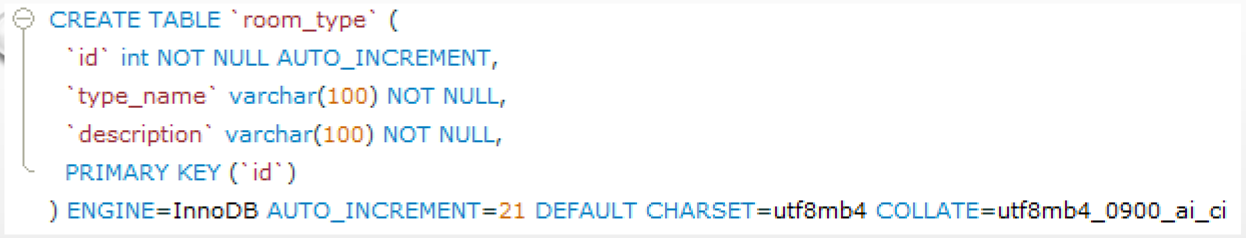


```
SELECT booking_id, rating, comment
FROM airbnb.review
WHERE rating = 4;
```

booking_id	rating	comment
13	4	Aperiam quis. Asperiores.
11	4	Perspiciatis laboriosam et.
14	4	Necessitatibus voluptate.
10	4	In non. Assumenda aut!
6	4	Molestias at. Et aliquid.
15	4	Rerum sit iure porro.

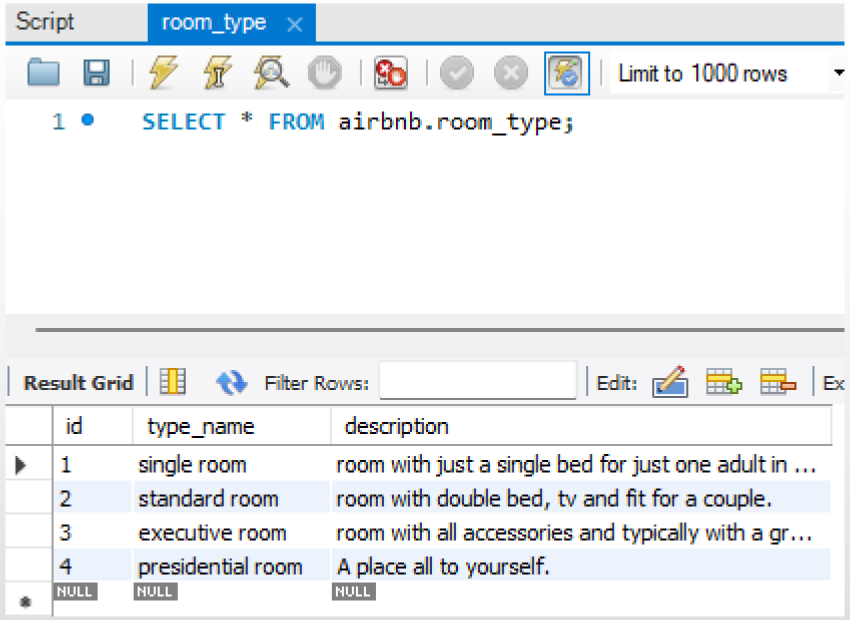
Using the WHERE statement we can select only display only rows that have a rating of 4.

• ROOM TYPE



```
CREATE TABLE `room_type` (
  `id` int NOT NULL AUTO_INCREMENT,
  `type_name` varchar(100) NOT NULL,
  `description` varchar(100) NOT NULL,
  PRIMARY KEY (`id`)
) ENGINE=InnoDB AUTO_INCREMENT=21 DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4_0900_ai_ci
```

Create table

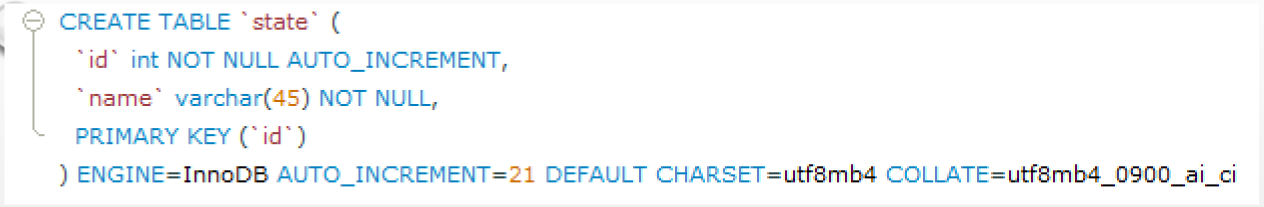


```
SELECT * FROM airbnb.room_type;
```

id	type_name	description
1	single room	room with just a single bed for just one adult in ...
2	standard room	room with double bed, tv and fit for a couple.
3	executive room	room with all accessories and typically with a gr...
4	presidential room	A place all to yourself.
NULL	NULL	NULL

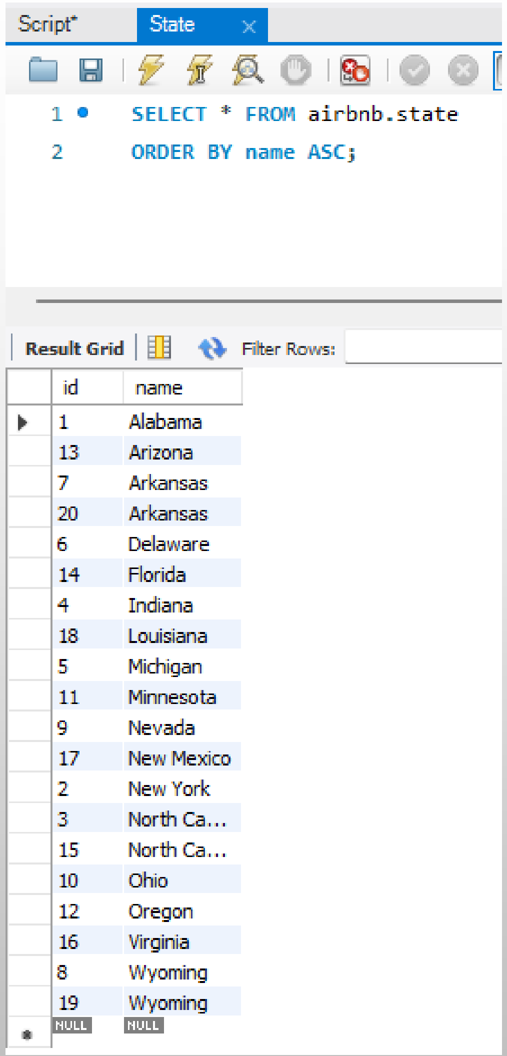
This table is supposed to go alongside with the property entity so that when guest view their preferences they are provided with enough details on the property.

• STATE



```
CREATE TABLE `state` (
  `id` int NOT NULL AUTO_INCREMENT,
  `name` varchar(45) NOT NULL,
  PRIMARY KEY (`id`)
) ENGINE=InnoDB AUTO_INCREMENT=21 DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4_0900_ai_ci
```

Create table

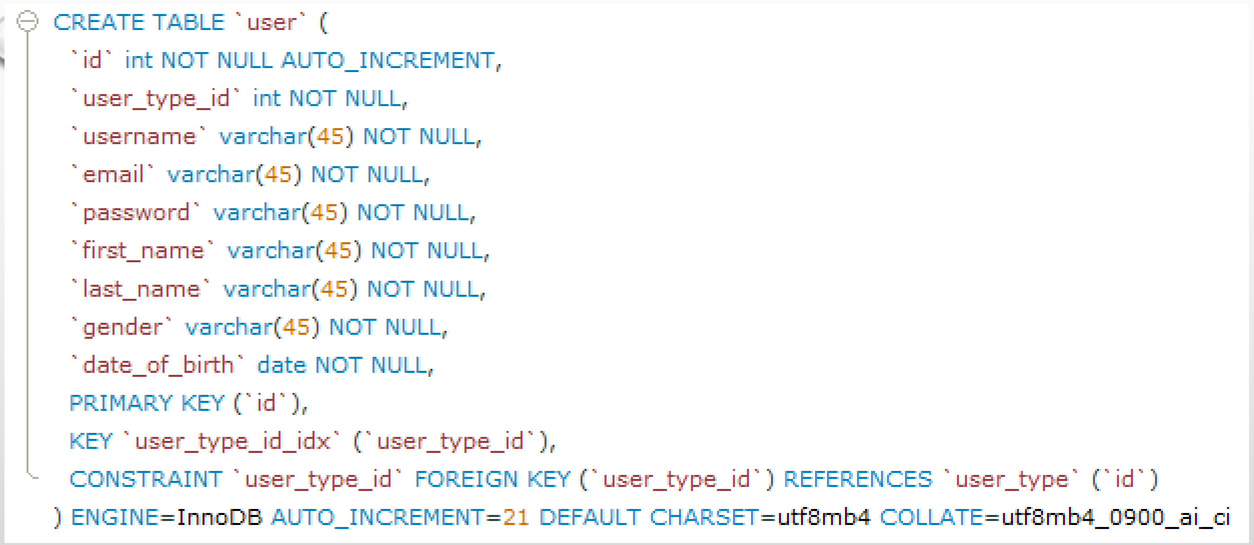


```
SELECT * FROM airbnb.state
ORDER BY name ASC;
```

id	name
1	Alabama
13	Arizona
7	Arkansas
20	Arkansas
6	Delaware
14	Florida
4	Indiana
18	Louisiana
5	Michigan
11	Minnesota
9	Nevada
17	New Mexico
2	New York
3	North Ca...
15	North Ca...
10	Ohio
12	Oregon
16	Virginia
8	Wyoming
19	Wyoming
NULL	NULL

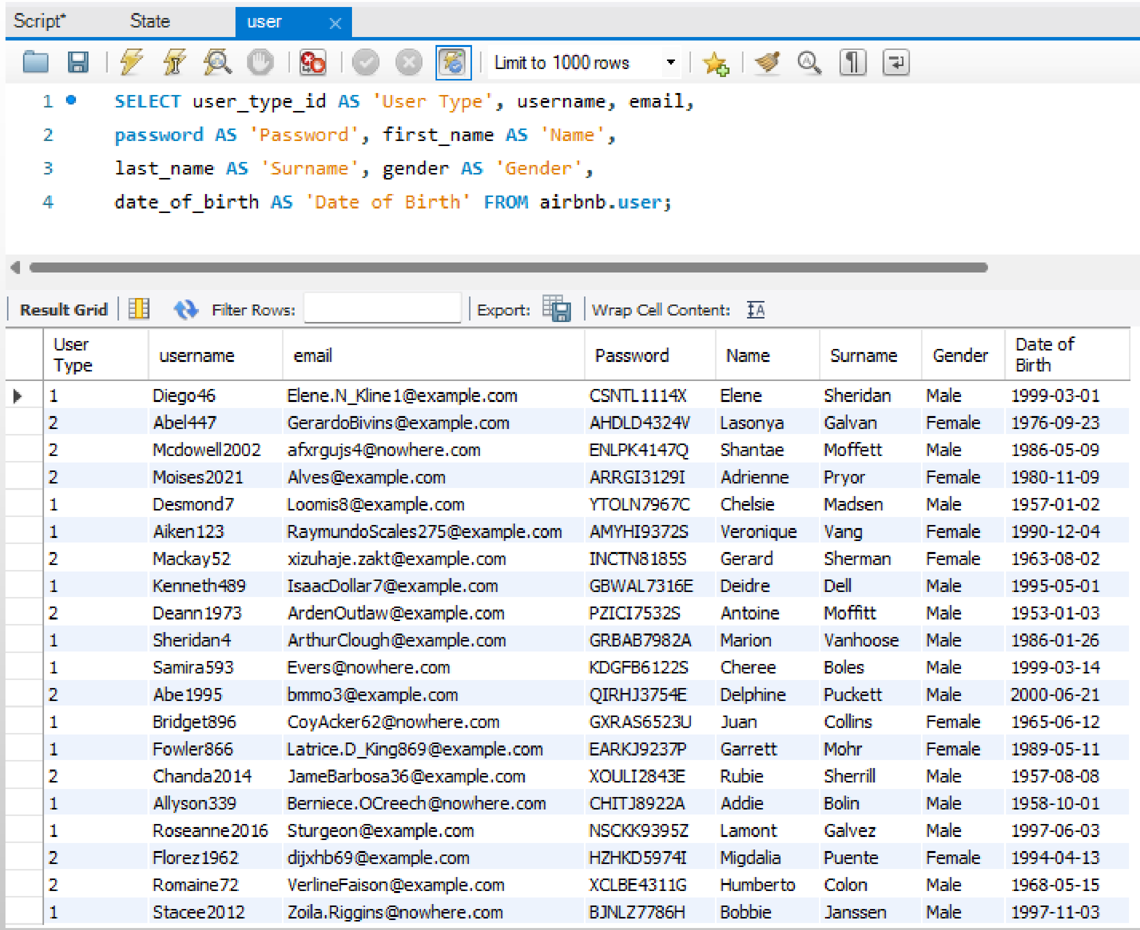
The Order by function is used to arrange the states by a specified order.

- **USER**



```
CREATE TABLE `user` (
  `id` int NOT NULL AUTO_INCREMENT,
  `user_type_id` int NOT NULL,
  `username` varchar(45) NOT NULL,
  `email` varchar(45) NOT NULL,
  `password` varchar(45) NOT NULL,
  `first_name` varchar(45) NOT NULL,
  `last_name` varchar(45) NOT NULL,
  `gender` varchar(45) NOT NULL,
  `date_of_birth` date NOT NULL,
  PRIMARY KEY (`id`),
  KEY `user_type_id_idx` (`user_type_id`),
  CONSTRAINT `user_type_id` FOREIGN KEY (`user_type_id`) REFERENCES `user_type` (`id`)
) ENGINE=InnoDB AUTO_INCREMENT=21 DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4_0900_ai_ci
```

Create table

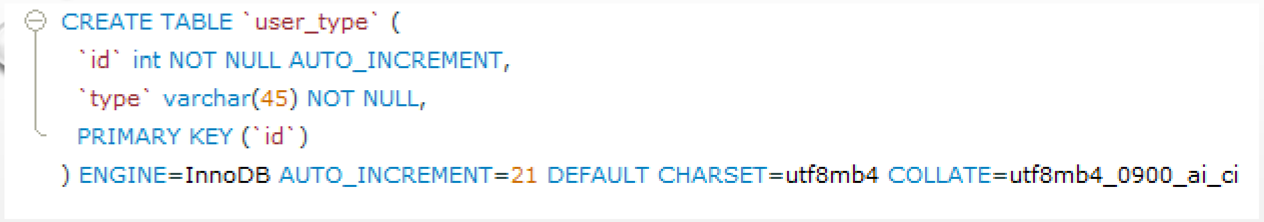


```
SELECT user_type_id AS 'User Type', username, email,
password AS 'Password', first_name AS 'Name',
last_name AS 'Surname', gender AS 'Gender',
date_of_birth AS 'Date of Birth' FROM airbnb.user;
```

User Type	username	email	Password	Name	Surname	Gender	Date of Birth
1	Diego46	Elene.N_Kline1@example.com	CSNTL1114X	Elene	Sheridan	Male	1999-03-01
2	Abel447	GerardoBivins@example.com	AHDLD4324V	Lasonya	Galvan	Female	1976-09-23
2	Mcdowell2002	afxrgujs4@nowhere.com	ENLPK4147Q	Shantae	Moffett	Male	1986-05-09
2	Moises2021	Alves@example.com	ARRGI3129I	Adrienne	Pryor	Female	1980-11-09
1	Desmond7	Loomis8@example.com	YTOLN7967C	Chelsie	Madsen	Male	1957-01-02
1	Aiken123	RaymundoScales275@example.com	AMYHI9372S	Veronique	Vang	Female	1990-12-04
2	Mackay52	xizuhaje.zakt@example.com	INCTN8185S	Gerard	Sherman	Female	1963-08-02
1	Kenneth489	IsaacDollar7@example.com	GBWAL7316E	Deidre	Dell	Male	1995-05-01
2	Deann1973	ArdenOutlaw@example.com	PZICI7532S	Antoine	Moffitt	Male	1953-01-03
1	Sheridan4	ArthurClough@example.com	GRBAB7982A	Marion	Vanhoose	Male	1986-01-26
1	Samira593	Evers@nowhere.com	KDGFB6122S	Cheree	Boles	Male	1999-03-14
2	Abe1995	bmmo3@example.com	QIRHJ3754E	Delphine	Puckett	Male	2000-06-21
1	Bridget896	CoyAcker62@nowhere.com	GXRAS6523U	Juan	Collins	Female	1965-06-12
1	Fowler866	Latrice.D_King869@example.com	EARKJ9237P	Garrett	Mohr	Female	1989-05-11
2	Chanda2014	JameBarbosa36@example.com	XOULI2843E	Rubie	Sherrill	Male	1957-08-08
1	Allyson339	Berniece.OCreech@nowhere.com	CHITJ8922A	Addie	Bolin	Male	1958-10-01
1	Roseanne2016	Sturgeon@example.com	NSCKK9395Z	Lamont	Galvez	Male	1997-06-03
2	Florez1962	dijxhb69@example.com	HZHKD5974I	Migdalia	Puente	Female	1994-04-13
2	Romaine72	VerlineFaison@example.com	XCLBE4311G	Humberto	Colon	Male	1968-05-15
1	Stacee2012	Zoila.Riggins@nowhere.com	BJNLZ7786H	Bobbie	Janssen	Male	1997-11-03

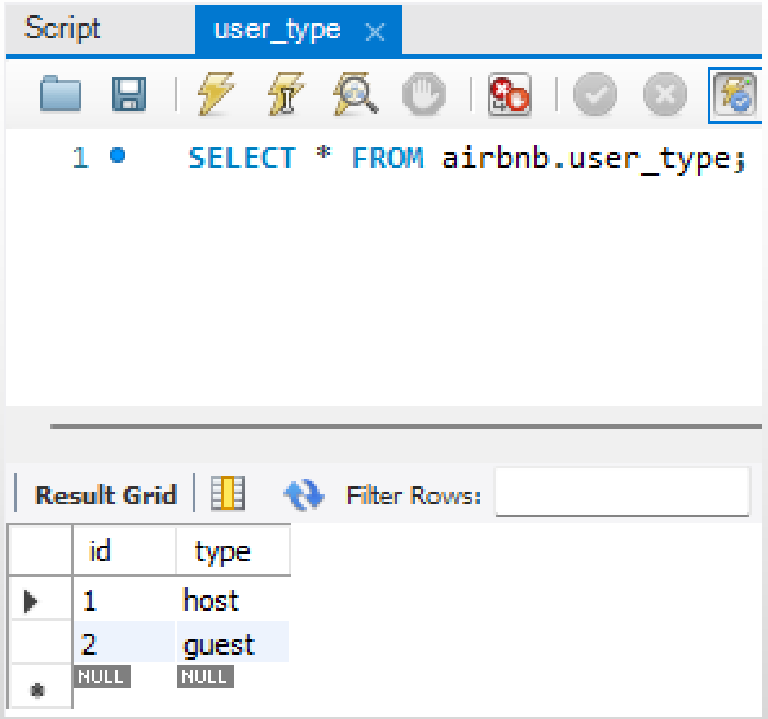
Function AS helps display columns as preferred names for easy readability when using the database.

- **USER TYPE**



```
CREATE TABLE `user_type` (
  `id` int NOT NULL AUTO_INCREMENT,
  `type` varchar(45) NOT NULL,
  PRIMARY KEY (`id`)
) ENGINE=InnoDB AUTO_INCREMENT=21 DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4_0900_ai_ci
```

Create table



```
SELECT * FROM airbnb.user_type;
```

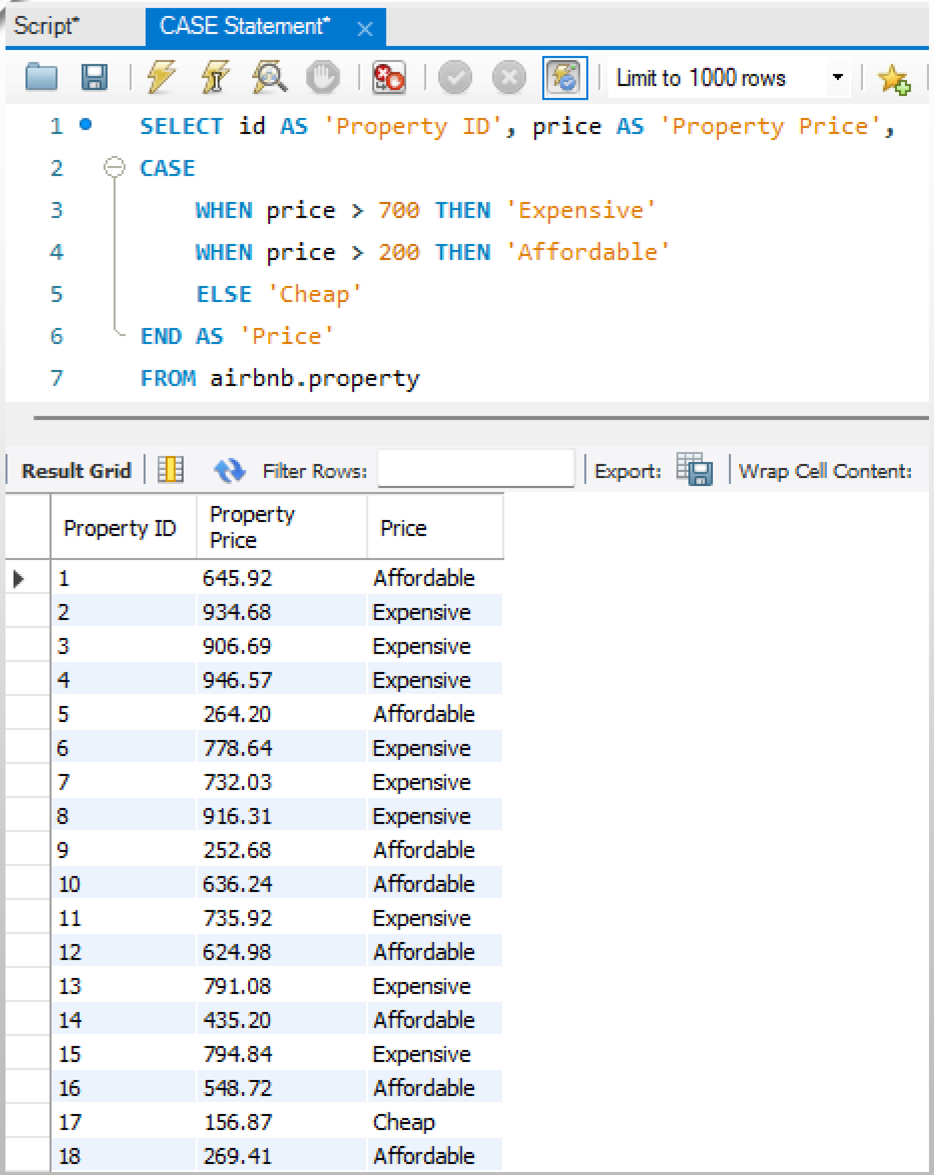
id	type
1	host
2	guest
NULL	NULL

Results of this query displays the two types of users for the Airbnb resources.

03

TEST CASES

- **CASE STATEMENT**

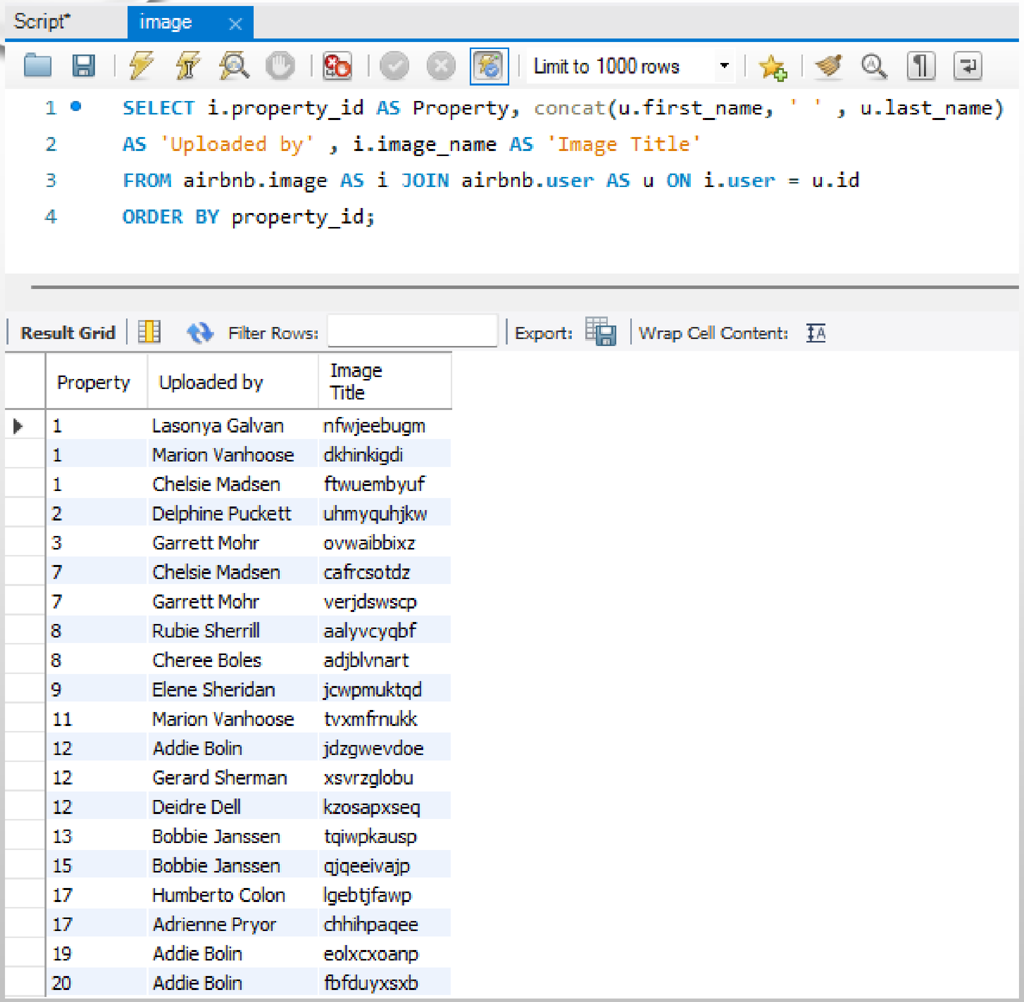


```sql
SELECT id AS 'Property ID', price AS 'Property Price',
CASE
    WHEN price > 700 THEN 'Expensive'
    WHEN price > 200 THEN 'Affordable'
    ELSE 'Cheap'
END AS 'Price'
FROM airbnb.property
```

Property ID	Property Price	Price
1	645.92	Affordable
2	934.68	Expensive
3	906.69	Expensive
4	946.57	Expensive
5	264.20	Affordable
6	778.64	Expensive
7	732.03	Expensive
8	916.31	Expensive
9	252.68	Affordable
10	636.24	Affordable
11	735.92	Expensive
12	624.98	Affordable
13	791.08	Expensive
14	435.20	Affordable
15	794.84	Expensive
16	548.72	Affordable
17	156.87	Cheap
18	269.41	Affordable

A CASE is a statement that operates if-then-else type of logical queries. In this test case, I used property table to describe the price column to show affordability of each property to each guest.

- **FUNCTIONS**

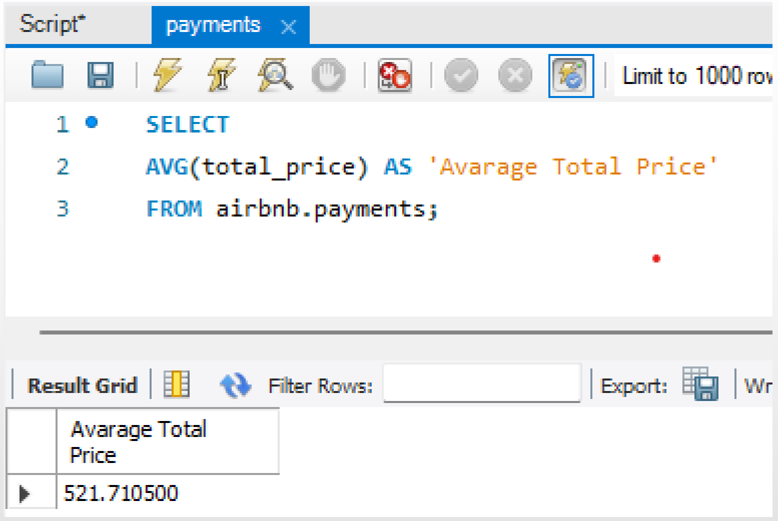


```
SELECT i.property_id AS Property, concat(u.first_name, ' ' , u.last_name)
AS 'Uploaded by' , i.image_name AS 'Image Title'
FROM airbnb.image AS i JOIN airbnb.user AS u ON i.user = u.id
ORDER BY property_id;
```

Property	Uploaded by	Image Title
1	Lasonya Galvan	nfwjeebugm
1	Marion Vanhoose	dkhinkigdi
1	Chelsie Madsen	ftwuembyuf
2	Delphine Puckett	uhmyquhjkw
3	Garrett Mohr	ovwaibbixz
7	Chelsie Madsen	cafrcsotdz
7	Garrett Mohr	verjdswscp
8	Rubie Sherrill	aalyvcyqbf
8	Cheree Boles	adjblvnart
9	Elene Sheridan	jcwpmuktqd
11	Marion Vanhoose	tvxmfrnukk
12	Addie Bolin	jdzgwevdoe
12	Gerard Sherman	xsvrzglobu
12	Deidre Dell	kzosapxseq
13	Bobbie Janssen	tqiwpkausp
15	Bobbie Janssen	qjqeeivajp
17	Humberto Colon	lgebtjfawp
17	Adrienne Pryor	chhihpaqee
19	Addie Bolin	eolxcxoanp
20	Addie Bolin	fbfduyxsxb

JOIN & CONCATENATE FUNCTION

Here the Concatenate Function has been used to take values from two columns and then put then together in one. JOIN is also used to reference the table that the Concatenate function is going to be used on to.

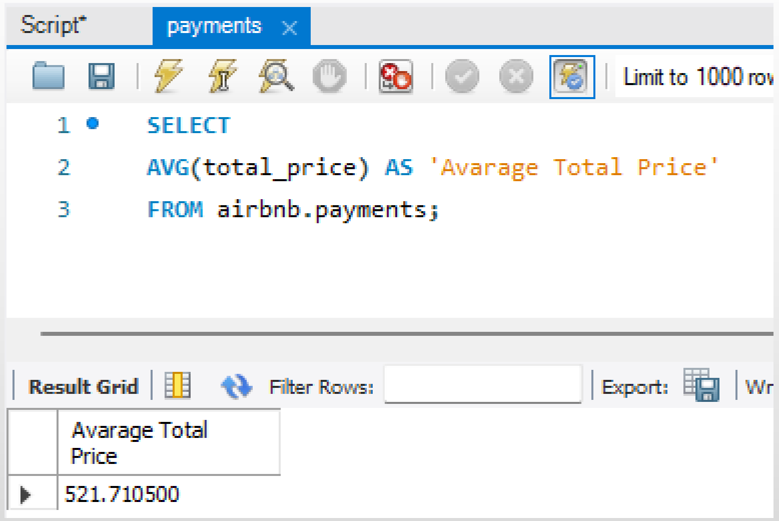


```
SELECT
AVG(total_price) AS 'Avarage Total Price'
FROM airbnb.payments;
```

Avarage Total Price
521.710500

AVERAGE FUNCTION

- **FUNCTIONS**



```
SELECT
AVG(total_price) AS 'Avarage Total Price'
FROM airbnb.payments;
```

```
Avarage Total Price
521.710500
```

Here the Concatenate Function has been used to take values from two columns and then put them together in one. JOIN is also used to reference the table that the Concatenate function is going to be used on to.

THANK YOU