ANALYSIS OF AIRBNB DATA USING SQL

SUBMITTED BY

ISHANI MAHAJAN(502304198)

VARUN SINGH(502304213)

Business Process and Scope Overview

·What is this system all about?

•This system is a Database Management System (DBMS) using SQL for analyzing Airbnb data.
It involves of a structured database to store and manage information related to Airbnb listings, hosts, reviews, and other relevant data. The primary goal is to perform analysis and gain insights from this data for various purposes.

LISTING | **REVIEW** | **HOSTS** | **BOOKING** | **AVAILABILITY**

·Why we need this system?

- The system is needed to harness the wealth of data .
- By organizing this data in a systematic and accessible manner, the organization can derive valuable insights into reviews, pricing strategies, host performance, and other critical aspects of the hospitality industry.
- This information is crucial for making informed business decisions.

·What is the scope of this system?

•Data Collection and Storage: Gathering Airbnb data from reliable sources and organizing it into a structured SQL database.
•Data Cleaning and Preprocessing: Ensuring that the data is accurate, complete, and ready for analysis.
•Database Design: Defining the schema, tables, and relationships to efficiently store and manage the data.

Data Analysis: Writing SQL queries to explore and analyse the data for various insights and trends.
•**Host Performance Analysis**: Understanding the preferences and behaviour of Airbnb users, which host have high number of listings. Rating comparison of hosts.
•**Market Analysis**: In which cities are Airbnb properties most prevalent, and what type of properties are popular in each location.

- **Pricing Strategies**: How does property price vary by city and property type.

•**Recommendations and Insights**: Providing actionable recommendations based on the analysis conducted.
•**Sentiment Analysis**: Assessing guest satisfaction through sentiment analysis of reviews.

Can you identify different customer segments based on their booking preferences, reviews, and property type preferences?

Do guests tend to leave more positive reviews when they stay for longer periods?

In which cities are Airbnb properties most prevalent, and what types of properties are popular in each location?

What is the average rating for each host, and how does it correlate with their response time and superhost status?

•How does this system work?

This system is a database-driven application designed to manage information about property listings, hosts, availability, reviews, and bookings.

It allows users to perform operations like retrieving data, modifying records, and defining database structure.

The system enforces data integrity, incorporates business logic, and may include error handling. It can be used for tasks like property rental and booking management.

·How will this system help the organization?

This system enhances customer experience, optimizes resources, and supports data-driven decision-making, providing the organization with a competitive edge and better business outcomes.

·Who are the key stakeholders in this system?

Key stakeholders include hosts, administrators, reviewers, data analysts, marketing, IT support, and product development teams. They collectively contribute to the platform's success and functionality.

•WHAT WE ARE TRYING TO EVALUATE?

We are trying to evaluate the performance, user satisfaction, and efficiency of the Airbnb-like platform.

This involves assessing factors like booking rates, customer reviews, host satisfaction, system uptime, revenue generation, and user feedback. The evaluation aims to understand how well the platform meets its goals and identifies areas for improvement.

- **Entities and their attributes in the provided system:**

1. **Entities:**
- **Host**
- **Listing**
- **Availability**
- **Review**
- **Booking**

2.Attributes:

Host:

- host_id (Primary Key)
- host_name
- host_location
- host_since
- host_response_time
- host_response_rate
- superhost

- **Listing:**
- listing_id (Primary Key)
- property_type
- room_type
- accommodates
- bedrooms
- bathrooms
- price
- city
- neighborhood
- rating
- host_id (Foreign Key)
- availability_id (Foreign Key)
- review_id (Foreign Key)

Availability:

- availability_id (Primary Key)
- listing_id (Foreign Key)
- available_date
- minimum_nights
- maximum_nights

Review:

- review_id (Primary Key)
- listing_id (Foreign Key)
- reviewer_id
- review_date
- review_text
- review_rating

Booking:

- booking_id (Primary Key)
- listing_id (Foreign Key)
- Reviewer_id
- checkin_date
- checkout_date

The relationships between the entities in the provided system:

Host:

- One-to-Many relationship with **Listing**: A host can have multiple listings.
- One-to-Many relationship with **Review**: A host can receive multiple reviews.

Listing:

- Many-to-One relationship with **Host**: Many listings belong to one host.
- Many-to-One relationship with **Availability**: Many listings can have multiple availability dates.
- Many-to-One relationship with **Review**: Many reviews can be associated with one listing.

Availability:

- Many-to-One relationship with **Listing**: Many availability entries belong to one listing.

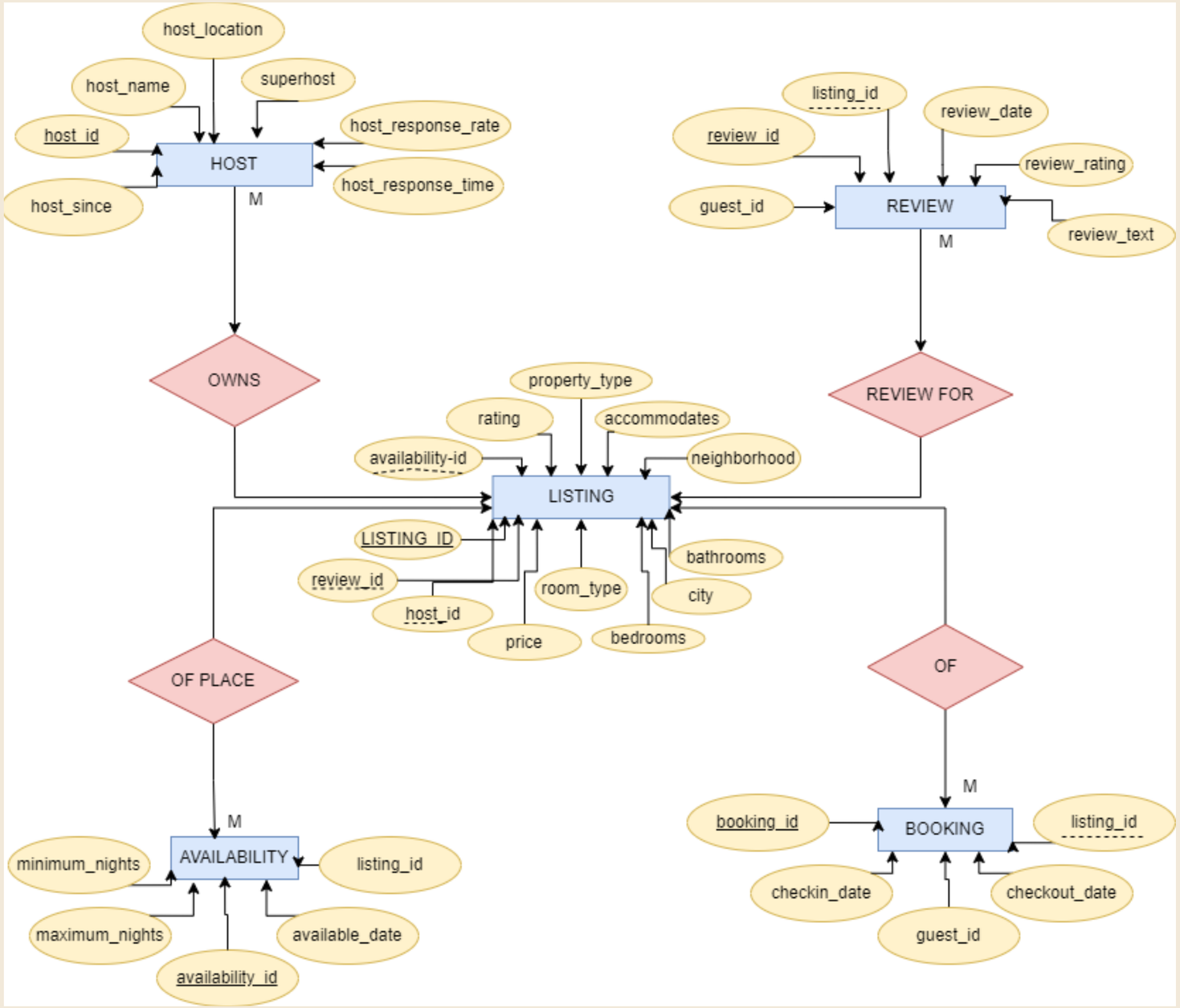
Review:

- Many-to-One relationship with **Listing**: Many reviews are associated with one listing.
- Many-to-One relationship with **Host**: Many reviews are written for one host.

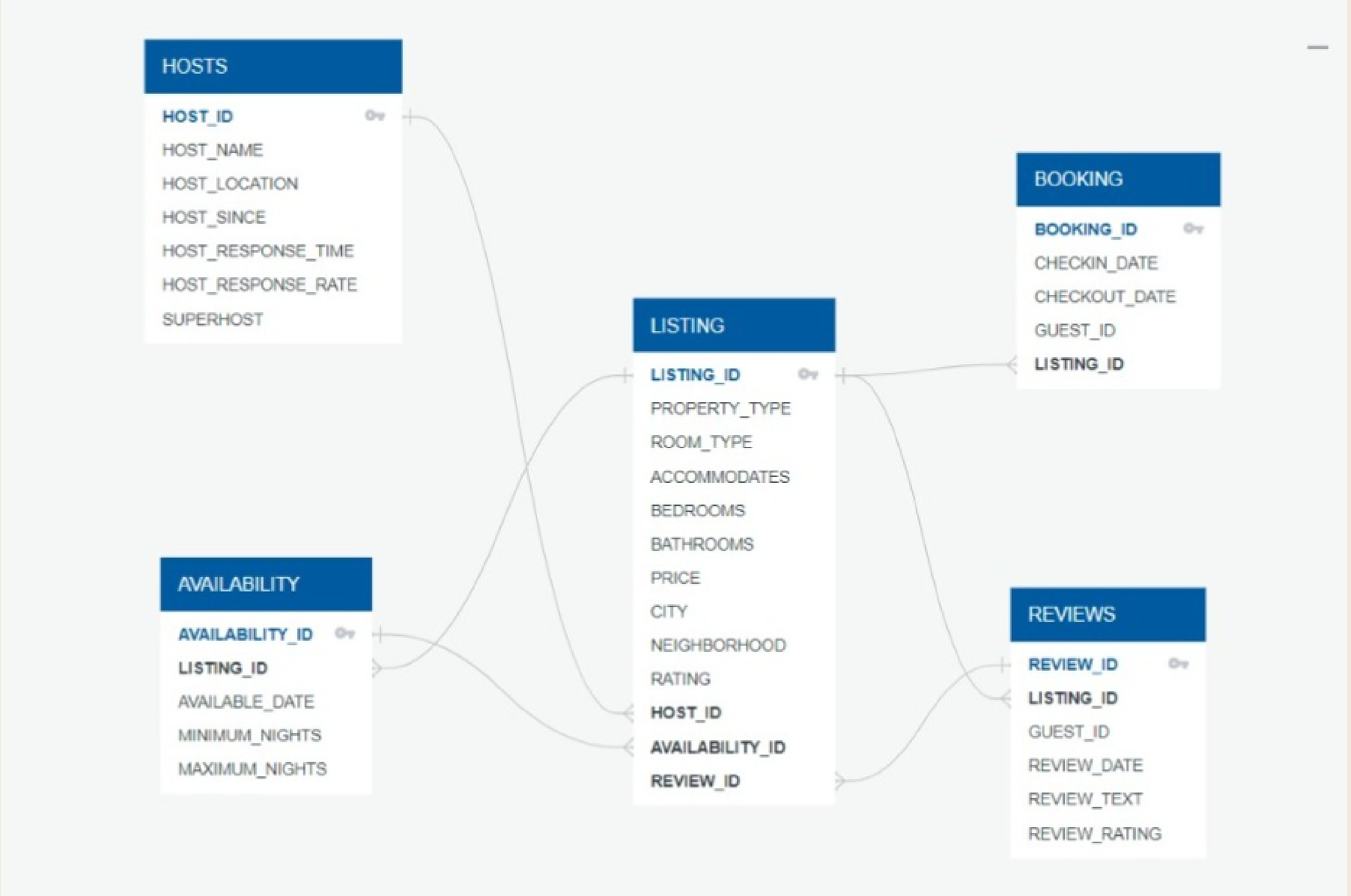
Booking:

- Many-to-One relationship with **Listing**: Many bookings are associated with one listing.

ER DIAGRAM


host_location
host_name
superhost
host_id
host_response_rate
HOST
host_response_time
host_since
M
listing_id
review_date
review_id
review_rating
guest_id
REVIEW
review_text
M
OWNS
REVIEW FOR
property_type
rating
accommodates
availability-id
neighborhood
LISTING
LISTING_ID
bathrooms
review_id
room_type
city
host_id
price
bedrooms
OF PLACE
OF
M
M
minimum_nights
AVAILABILITY
listing_id
booking_id
BOOKING
listing_id
maximum_nights
available_date
checkin_date
checkout_date
availability_id
guest_id

RELATIONAL SCHEMA



DDL STATEMENTS:-

```
CREATE DATABASE Airbnb;
USE Airbnb;
CREATE TABLE Listings (
   listing_id INT PRIMARY KEY,
   property_type VARCHAR(255),
   room_type VARCHAR(255),
   accommodates INT,
   bedrooms INT,
   bathrooms INT,
   price DECIMAL(10, 2),
   city VARCHAR(255),
   neighborhood VARCHAR(255),
   rating DECIMAL(3, 2),
   host_id INT,
   availability_id INT,
   review_id INT,
   FOREIGN KEY (host_id) REFERENCES
Hosts(host_id)
);
```

```
CREATE TABLE Hosts (
   host_id INT PRIMARY KEY,
   host_name VARCHAR(255),
   host_location VARCHAR(255),
   host_since DATE,
   host_response_time VARCHAR(255),
   host_response_rate DECIMAL(5, 2),
   superhost BOOLEAN
);
```

```
CREATE TABLE Availability (
 availability_id INT PRIMARY KEY,
 listing_id INT,
 available_date DATE,
 minimum_nights INT,
 maximum_nights INT,
 FOREIGN KEY (listing_id) REFERENCES
Listings(listing_id)
);
```

```
CREATE TABLE Reviews (
 review_id INT PRIMARY KEY,
 listing_id INT,
 reviewer_id INT,
 review_date DATE,
 review_text TEXT,
 review_rating DECIMAL(3, 2)
);
```

```
CREATE TABLE Booking (
 booking_id INT PRIMARY KEY,
 listing_id INT,
 Reviewer_id INT,
 checkin_date DATE,
 checkout_date DATE
);
```

CREATE VIEW

```
CREATE VIEW Listing_host AS
SELECT
 Listings.listing_id,
 Listings.property_type,
 Listings.room_type,
 Listings.accommodates,
 Listings.bedrooms,
 Listings.bathrooms,
Listings.price,
 Listings.city,
 Listings.neighborhood,
 Listings.rating,
 Hosts.host_name,
 Hosts.host_location,
 Hosts.host_since,
 Hosts.host_response_time,
 Hosts.host_response_rate,
 Hosts.superhost
FROM
 Listings
INNER JOIN
 Hosts ON Listings.host_id = Hosts.host_id;
```

```
CREATE VIEW LISTING_REVIEWS AS
SELECT
 Listings.listing_id,
 Listings.property_type,
 Reviews.review_id,
 Reviews.reviewer_id,
 Reviews.review_date,
 Reviews.review_text,
 Reviews.review_rating
FROM
 Listings
INNER JOIN Reviews ON
Listings.listing_id = Reviews.listing_id;
```

```
CREATE VIEW Avg_rating AS
SELECT
 H.host_id,
 H.host_name,
 AVG(R.review_rating) AS average_rating
FROM
 Hosts AS H
INNER JOIN Listings AS L USING(host_id)
LEFT JOIN Reviews AS R USING(listing_id)
GROUP BY
 H.host_id;

CREATE VIEW NO_OF_LISTINGS AS
 SELECT
 H.host_id,
 H.host_name,
 L.listing_id,
 L.property_type
FROM
 Hosts AS H
LEFT JOIN Listings AS L ON H.host_id = L.host_id
ORDER BY
 H.host_id;
```

```
CREATE VIEW Response_time AS
SELECT
 H.host_id,
 H.host_name,
 L.listing_id,
 L.property_type,
 L.city,
 H.host_response_time,
 R.review_rating,
 R.review_text
FROM
 Hosts AS H
INNER JOIN
 Listings AS L ON H.host_id = L.host_id
LEFT JOIN
 Reviews AS R ON L.listing_id = R.listing_id
ORDER BY
 H.host_id, L.listing_id;
```

```
CREATE VIEW list_of_properties AS
SELECT
 L.listing_id,
 L.property_type,
 L.room_type,
 L.city,
 H.host_id,
 H.host_name
FROM
 Listings AS L
INNER JOIN Hosts AS H ON L.host_id = H.host_id
ORDER BY
 L.listing_id;
```

```
ALTER TABLE Reviews
MODIFY reviewer_id INT NOT NULL;

SELECT * FROM Reviews;
ALTER TABLE Reviews
ADD COLUMN booking_id INT;

ALTER TABLE Reviews
ADD CONSTRAINT FK_Reviews_Booking
FOREIGN KEY (booking_id)
REFERENCES Booking(booking_id);

ALTER TABLE reviews
ADD UNIQUE(reviewer_id);
```

```
ALTER TABLE hosts
MODIFY host_name VARCHAR(255) NOT
NULL;

ALTER TABLE hosts
MODIFY host_since DATE NOT NULL;
```

```
ALTER TABLE listings
add unique(listing_id);
```

DML STATEMENTS:

```
INSERT INTO Listings (listing_id, property_type, room_type, accommodates, bedrooms, bathrooms, price, city, neighborhood, rating, host_id, availability_id, review_id)
VALUES
(1, 'Apartment', 'Entire home/apt', 4, 2, 1, 100.00, 'New York', 'Manhattan', 4.8, 101, 201, 301),
(2, 'House', 'Private room', 2, 1, 1, 60.00, 'Los Angeles', 'Venice Beach', 4.5, 102, 202, 302),
(3, 'Studio', 'Entire home/apt', 2, 1, 1, 80.00, 'San Francisco', 'Downtown', 4.7, 103, 203, 303),
(4, 'Villa', 'Entire home/apt', 6, 3, 2, 250.00, 'Miami', 'Coconut Grove', 4.9, 104, 204, 304),
(5, 'Cabin', 'Entire home/apt', 4, 2, 1, 120.00, 'Asheville', 'Blue Ridge', 4.8, 105, 205, 305),
(6, 'Apartment', 'Private room', 2, 1, 1, 50.00, 'Chicago', 'Wicker Park', 4.6, 106, 206, 306),
(7, 'House', 'Entire home/apt', 6, 3, 2, 180.00, 'Austin', 'South Congress', 4.7, 107, 207, 307),
(8, 'Condo', 'Entire home/apt', 2, 1, 1, 90.00, 'Seattle', 'Capitol Hill', 4.8, 108, 208, 308),
(9, 'Apartment', 'Private room', 2, 1, 1, 60.00, 'Paris', 'Le Marais', 4.5, 109, 209, 309),
(10, 'House', 'Entire home/apt', 4, 2, 2, 200.00, 'London', 'Notting Hill', 4.7, 110, 210, 310),
(11, 'Cottage', 'Entire home/apt', 2, 1, 1, 70.00, 'Sydney', 'Bondi Beach', 4.6, 111, 211, 311),
(12, 'Apartment', 'Entire home/apt', 4, 2, 1, 110.00, 'Barcelona', 'Gothic Quarter', 4.8, 112, 212, 312),
(13, 'House', 'Entire home/apt', 6, 3, 2, 220.00, 'Tokyo', 'Shinjuku', 4.9, 113, 213, 313),
(14, 'Villa', 'Entire home/apt', 8, 4, 3, 350.00, 'Bali', 'Ubud', 4.9, 114, 214, 314),
(15, 'Apartment', 'Private room', 2, 1, 1, 55.00, 'Rome', 'Trastevere', 4.5, 115, 215, 315),
(16, 'Cabin', 'Entire home/apt', 4, 2, 1, 125.00, 'Denver', 'Rocky Mountains', 4.8, 116, 216, 316),
(17, 'Studio', 'Entire home/apt', 2, 1, 1, 75.00, 'San Diego', 'Pacific Beach', 4.7, 117, 217, 317),
(18, 'Apartment', 'Private room', 2, 1, 1, 48.00, 'Berlin', 'Mitte', 4.6, 118, 218, 318),
(19, 'House', 'Entire home/apt', 5, 3, 2, 190.00, 'Toronto', 'Downtown', 4.7, 119, 219, 319),
(20, 'Condo', 'Entire home/apt', 2, 1, 1, 95.00, 'Dubai', 'Marina', 4.8, 120, 220, 320);
```

```
INSERT INTO Hosts (host_id, host_name, host_location, host_since, host_response_time, host_response_rate, superhost)
VALUES
(101, 'John Smith', 'New York, NY', '2020-01-15', 'Within an hour', 95.0, 1),
(102, 'Emily Johnson', 'Los Angeles, CA', '2018-03-10', 'Within a few hours', 90.5, 0),
(103, 'Michael Brown', 'Chicago, IL', '2019-05-20', 'Within a day', 88.0, 0),
(104, 'Sarah Davis', 'Miami, FL', '2021-09-10', 'Within a few hours', 92.5, 1),
(105, 'David Wilson', 'San Francisco, CA', '2017-12-05', 'Within an hour', 96.0, 1),
(106, 'Lisa Jones', 'Los Angeles, CA', '2020-08-15', 'Within an hour', 94.5, 1),
(107, 'Robert Miller', 'Seattle, WA', '2019-04-22', 'Within a day', 87.5, 0),
(108, 'Mary Taylor', 'Austin, TX', '2020-03-01', 'Within a few hours', 91.0, 1),
(109, 'James Johnson', 'New York, NY', '2016-11-28', 'Within an hour', 98.0, 1),
(110, 'Linda Harris', 'San Diego, CA', '2018-06-17', 'Within a day', 86.5, 0),
(111, 'Christopher Anderson', 'London, UK', '2019-02-14', 'Within a day', 89.5, 0),
(112, 'Karen White', 'Paris, France', '2021-03-12', 'Within a few hours', 93.5, 1),
(113, 'William Clark', 'Sydney, Australia', '2017-07-19', 'Within a few hours', 92.0, 0),
(114, 'Jennifer Lee', 'Barcelona, Spain', '2022-04-30', 'Within an hour', 97.5, 1),
(115, 'Richard Harris', 'Tokyo, Japan', '2020-02-03', 'Within a day', 90.0, 0),
(116, 'Donna Martin', 'Bali, Indonesia', '2019-08-25', 'Within a few hours', 91.5, 1),
(117, 'George Wilson', 'Rome, Italy', '2018-04-07', 'Within an hour', 97.0, 1),
(118, 'Nancy Adams', 'Denver, CO', '2020-10-12', 'Within an hour', 93.0, 1),
(119, 'Thomas Young', 'Berlin, Germany', '2017-01-04', 'Within a few hours', 89.0, 0),
(120, 'Patricia Hall', 'Toronto, Canada', '2021-06-29', 'Within a day', 87.0, 0);
```

```
INSERT INTO Availability (availability_id, listing_id, available_date, minimum_nights, maximum_nights)
VALUES
(201, 1, '2023-10-25', 3, 30),
(202, 2, '2023-11-01', 2, 15),
(203, 3, '2023-10-27', 2, 20),
(204, 4, '2023-10-30', 3, 25),
(205, 5, '2023-11-02', 1, 10),
(206, 6, '2023-11-05', 2, 30),
(207, 7, '2023-10-29', 2, 15),
(208, 8, '2023-10-26', 3, 20),
(209, 9, '2023-11-03', 2, 25),
(210, 10, '2023-10-28', 1, 10),
(211, 11, '2023-11-04', 3, 30),
(212, 12, '2023-11-06', 2, 15),
(213, 13, '2023-10-31', 2, 20),
(214, 14, '2023-11-07', 3, 25),
(215, 15, '2023-10-26', 1, 10),
(216, 16, '2023-11-01', 3, 30),
(217, 17, '2023-11-08', 2, 15),
(218, 18, '2023-10-30', 2, 20);

INSERT INTO Reviews (review_id, listing_id, reviewer_id, review_date, review_text, review_rating)
VALUES
(301, 1, 1001, '2023-10-30' ,'Great place to stay!', 4.9),
(302, 2, 1002, '2023-10-22','Nice host and comfortable room.', 4.5),
(303, 3, 1003, '2023-10-15','Enjoyed my stay here. Clean and cozy.', 4.8),
(304, 4, 1004, '2023-09-30','Great villa, perfect for a family.', 4.9),
(305, 5, 1005, '2023-10-25','Beautiful cabin in a stunning location.', 4.7),
```

```
(306, 6, 1006, '2023-11-06','The host was very accommodating.', 4.5),
(307, 7, 1007, '2023-11-05','Lovely house with a great view.', 4.8),
(308, 8, 1008, '2023-11-02','Wonderful condo in a convenient location.', 4.6),
(309, 9, 1009, '2023-11-10','Charming apartment in the heart of Paris.', 4.5),
(310, 10, 1010, '2023-09-26','Loved the house in Notting Hill.', 4.7),
(311, 11, 1011, '2023-11-08','Cozy cottage by the beach.', 4.6),
(312, 12, 1012, '2023-10-21','Clean apartment in a historic neighborhood.', 4.8),
(313, 13, 1013, '2023-11-16','Great villa in the heart of Tokyo.', 4.9),
(314, 14, 1014, '2023-11-14','The view from this villa is breathtaking.', 4.9),
(315, 15, 1015, '2023-11-11','Comfortable apartment in a great location.', 4.7),
(316, 16, 1016, '2023-10-12','Perfect cabin for a mountain getaway.', 4.8),
(317, 17, 1017, '2023-11-13','Enjoyed the studio by the beach.', 4.7),
(318, 18, 1018, '2023-11-28','Great apartment in the heart of Berlin.', 4.6),
(319, 19, 1019, '2023-11-19','Nice house in the heart of Toronto.', 4.7),
(320, 20, 1020, '2023-10-05','Great condo with a beautiful view in Dubai.', 4.8),
(321, 21, 1021, '2023-10-10','Enjoyed my stay in Hong Kong.', 4.6),
(322, 22, 1022, '2023-10-09','Cozy cabin in the mountains.', 4.7),
(323, 23, 1023, '2023-10-04','Beautiful villa in Bali.', 4.9),
(324, 24, 1024, '2023-11-30','Clean apartment in the heart of Rome.', 4.6);

INSERT INTO Booking (booking_id, listing_id, guest_id, checkin_date, checkout_date)
VALUES
-- Booking 1
(1, 1, 1001, '2023-11-01', '2023-11-07'),
-- Booking 2
(2, 2, 1002, '2023-11-02', '2023-11-10'),
-- Booking 3
(3, 3, 1003, '2023-11-03', '2023-11-05'),
-- Booking 4 (Repeating guest)
(4, 4, 1001, '2023-11-06', '2023-11-12'),
-- Booking 5 (Repeating guest)
(5, 5, 1002, '2023-11-08', '2023-11-15'),
-- Booking 6
(6, 6, 1004, '2023-11-04', '2023-11-09'),
-- Booking 7
(7, 7, 1005, '2023-11-07', '2023-11-10'),
-- Booking 8 (Repeating guest)
(8, 8, 1001, '2023-11-10', '2023-11-17'),
-- Booking 9
(9, 9, 1006, '2023-11-05', '2023-11-12'),
-- Booking 10
(10, 10, 1007, '2023-11-08', '2023-11-14'),
-- Booking 11
(11, 11, 1008, '2023-11-11', '2023-11-15'),
-- Booking 12 (Repeating guest)
(12, 12, 1002, '2023-11-13', '2023-11-20'),
-- Booking 13
(13, 13, 1009, '2023-11-14', '2023-11-16'),
-- Booking 14
(14, 14, 1010, '2023-11-17', '2023-11-20'),
-- Booking 15
(15, 15, 1011, '2023-11-18', '2023-11-22'),
-- Booking 16 (Repeating guest)
(16, 16, 1004, '2023-11-20', '2023-11-27'),
-- Booking 17 (Repeating guest)
(17, 17, 1006, '2023-11-23', '2023-11-30'),
-- Booking 18
(18, 18, 1012, '2023-11-25', '2023-11-30'),
-- Booking 19
(19, 19, 1013, '2023-11-27', '2023-12-02'),
-- Booking 20
(20, 20, 1014, '2023-11-30', '2023-12-05');
```

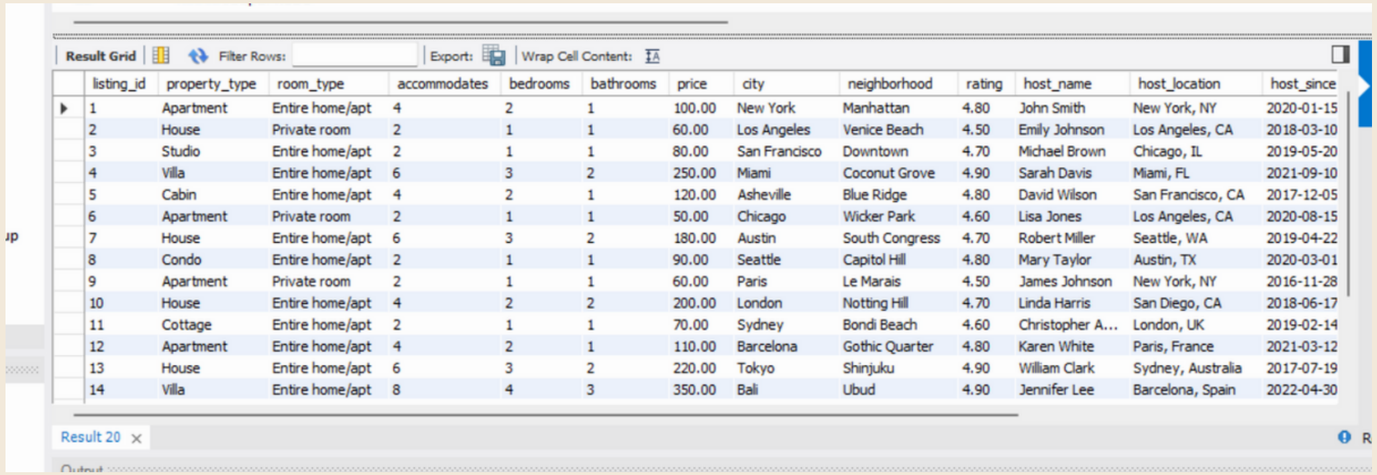
DQL STATEMENTS:

```
select * from listings;
select * from availability;
select * from reviews;
select * from hosts;
```

- **Detailed information about each listing, along with the details of the corresponding host.**

```
SELECT
  Listings.listing_id,
  Listings.property_type,
  Listings.room_type,
  Listings.accommodates,
  Listings.bedrooms,
  Listings.bathrooms,
  Listings.price,
  Listings.city,
  Listings.neighborhood,
  Listings.rating,
  Hosts.host_name,
  Hosts.host_location,
  Hosts.host_since,
  Hosts.host_response_time,
  Hosts.host_response_rate,
  Hosts.superhost
FROM
  Listings
INNER JOIN
  Hosts ON Listings.host_id =
Hosts.host_id;
```

OUTPUT:



Result Grid | Filter Rows: | Export: | Wrap Cell Content:

listing_id	property_type	room_type	accommodates	bedrooms	bathrooms	price	city	neighborhood	rating	host_name	host_location	host_since
1	Apartment	Entire home/apt	4	2	1	100.00	New York	Manhattan	4.80	John Smith	New York, NY	2020-01-15
2	House	Private room	2	1	1	60.00	Los Angeles	Venice Beach	4.50	Emily Johnson	Los Angeles, CA	2018-03-10
3	Studio	Entire home/apt	2	1	1	80.00	San Francisco	Downtown	4.70	Michael Brown	Chicago, IL	2019-05-20
4	Villa	Entire home/apt	6	3	2	250.00	Miami	Coconut Grove	4.90	Sarah Davis	Miami, FL	2021-09-10
5	Cabin	Entire home/apt	4	2	1	120.00	Asheville	Blue Ridge	4.80	David Wilson	San Francisco, CA	2017-12-05
6	Apartment	Private room	2	1	1	50.00	Chicago	Wicker Park	4.60	Lisa Jones	Los Angeles, CA	2020-08-15
7	House	Entire home/apt	6	3	2	180.00	Austin	South Congress	4.70	Robert Miller	Seattle, WA	2019-04-22
8	Condo	Entire home/apt	2	1	1	90.00	Seattle	Capitol Hill	4.80	Mary Taylor	Austin, TX	2020-03-01
9	Apartment	Private room	2	1	1	60.00	Paris	Le Marais	4.50	James Johnson	New York, NY	2016-11-28
10	House	Entire home/apt	4	2	2	200.00	London	Notting Hill	4.70	Linda Harris	San Diego, CA	2018-06-17
11	Cottage	Entire home/apt	2	1	1	70.00	Sydney	Bondi Beach	4.60	Christopher A...	London, UK	2019-02-14
12	Apartment	Entire home/apt	4	2	1	110.00	Barcelona	Gothic Quarter	4.80	Karen White	Paris, France	2021-03-12
13	House	Entire home/apt	6	3	2	220.00	Tokyo	Shinjuku	4.90	William Clark	Sydney, Australia	2017-07-19
14	Villa	Entire home/apt	8	4	3	350.00	Bali	Ubud	4.90	Jennifer Lee	Barcelona, Spain	2022-04-30

Result 20

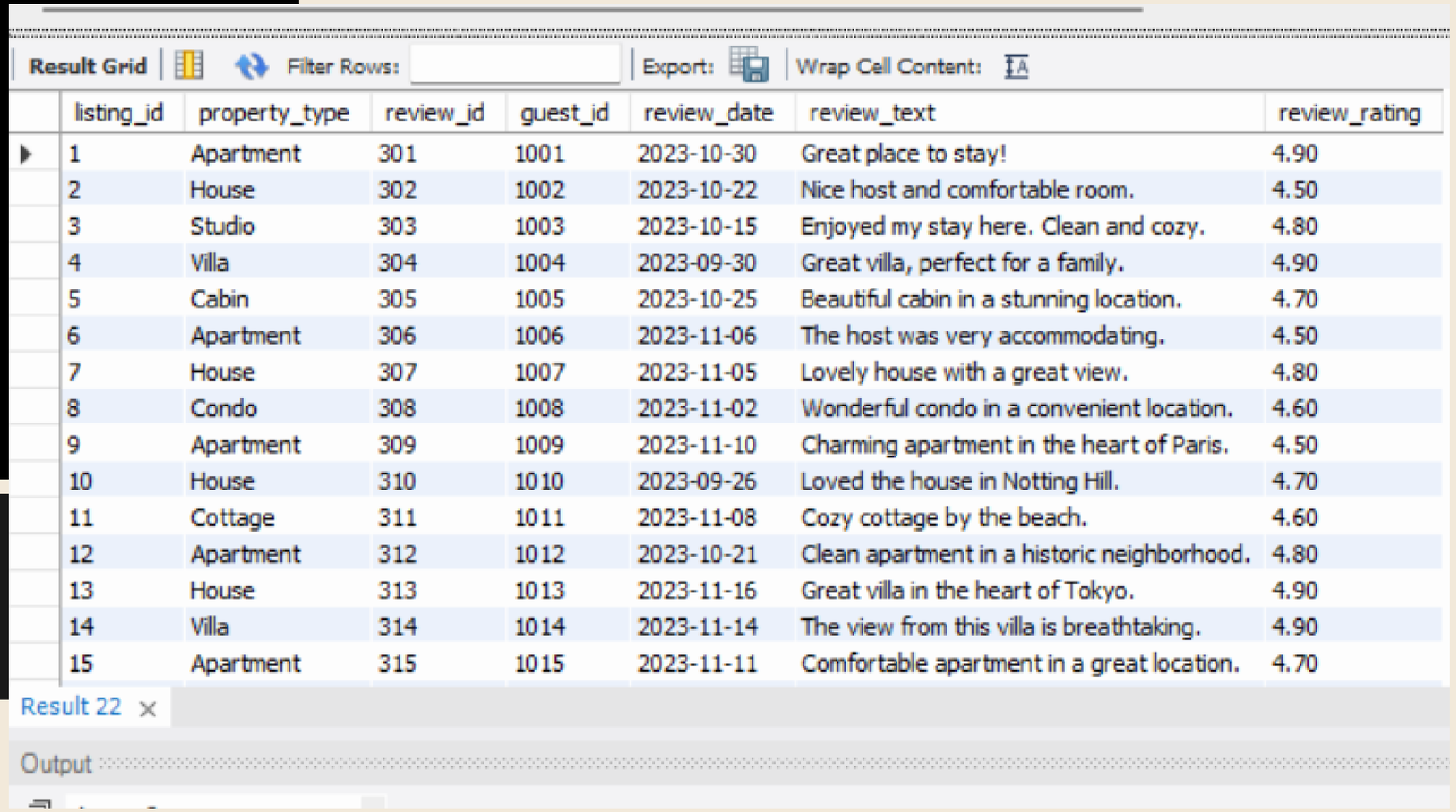
We specify an INNER JOIN between the Listings and Reviews tables using the listing_id column as the joining condition.

```
SELECT
 Listings.listing_id,
 Listings.property_type,
 Reviews.review_id,
 Reviews.guest_id,
 Reviews.review_date,
 Reviews.review_text,
 Reviews.review_rating
FROM
 Listings
INNER JOIN Reviews ON Listings.listing_id =
Reviews.listing_id;
DESC REVIEWS;
```

Output of view listing_reviews

```
SELECT * FROM Listing_reviews;
```

OUTPUT:



Result Grid | Filter Rows: | Export: | Wrap Cell Content:

listing_id	property_type	review_id	guest_id	review_date	review_text	review_rating
1	Apartment	301	1001	2023-10-30	Great place to stay!	4.90
2	House	302	1002	2023-10-22	Nice host and comfortable room.	4.50
3	Studio	303	1003	2023-10-15	Enjoyed my stay here. Clean and cozy.	4.80
4	Villa	304	1004	2023-09-30	Great villa, perfect for a family.	4.90
5	Cabin	305	1005	2023-10-25	Beautiful cabin in a stunning location.	4.70
6	Apartment	306	1006	2023-11-06	The host was very accommodating.	4.50
7	House	307	1007	2023-11-05	Lovely house with a great view.	4.80
8	Condo	308	1008	2023-11-02	Wonderful condo in a convenient location.	4.60
9	Apartment	309	1009	2023-11-10	Charming apartment in the heart of Paris.	4.50
10	House	310	1010	2023-09-26	Loved the house in Notting Hill.	4.70
11	Cottage	311	1011	2023-11-08	Cozy cottage by the beach.	4.60
12	Apartment	312	1012	2023-10-21	Clean apartment in a historic neighborhood.	4.80
13	House	313	1013	2023-11-16	Great villa in the heart of Tokyo.	4.90
14	Villa	314	1014	2023-11-14	The view from this villa is breathtaking.	4.90
15	Apartment	315	1015	2023-11-11	Comfortable apartment in a great location.	4.70

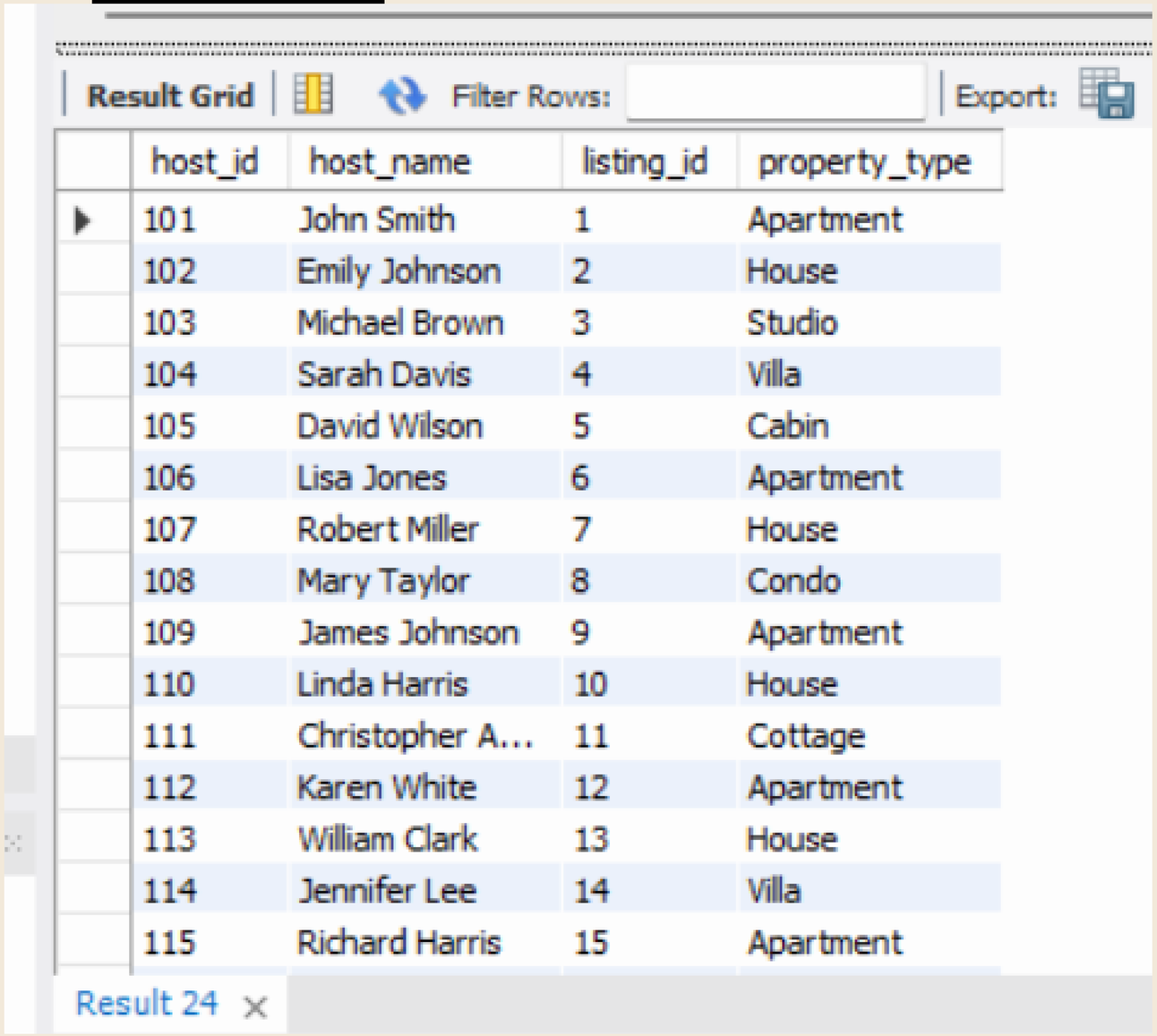
Result 22

Output

Hosts and Their Properties: a LEFT JOIN to combine the Hosts and Listings tables, showing a list of hosts along with the properties they are hosting.

```
SELECT
   H.host_id,
   H.host_name,
   L.listing_id,
   L.property_type
FROM
   Hosts AS H
LEFT JOIN Listings AS L ON H.host_id = L.host_id
ORDER BY
   H.host_id;
```

OUTPUT:



Result Grid | Filter Rows: | Export:

host_id	host_name	listing_id	property_type
101	John Smith	1	Apartment
102	Emily Johnson	2	House
103	Michael Brown	3	Studio
104	Sarah Davis	4	Villa
105	David Wilson	5	Cabin
106	Lisa Jones	6	Apartment
107	Robert Miller	7	House
108	Mary Taylor	8	Condo
109	James Johnson	9	Apartment
110	Linda Harris	10	House
111	Christopher A...	11	Cottage
112	Karen White	12	Apartment
113	William Clark	13	House
114	Jennifer Lee	14	Villa
115	Richard Harris	15	Apartment

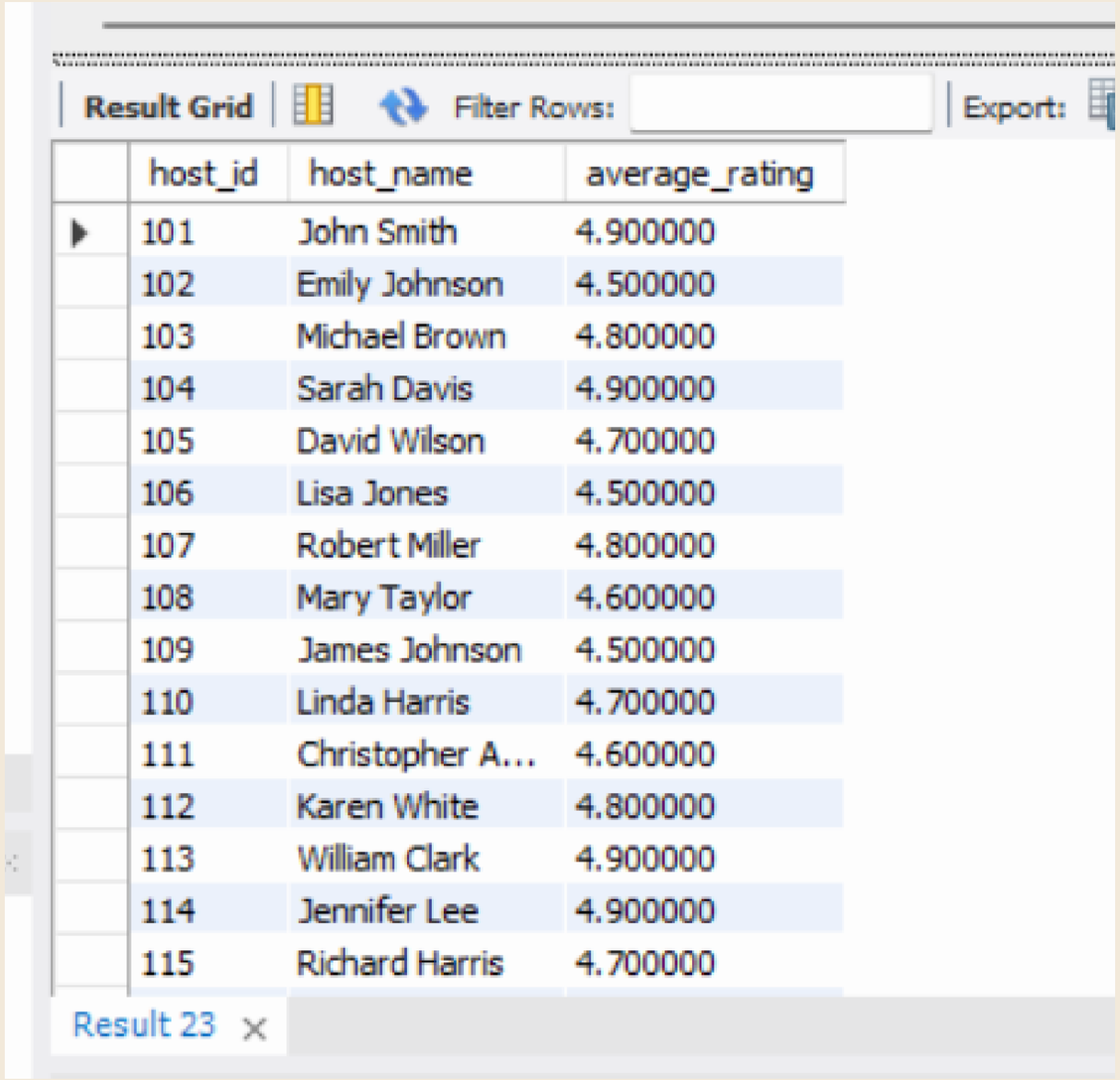
Result 24

The result will provide you with each host's ID, name, and their average rating based on the reviews they have received.

```
SELECT
 H.host_id,
 H.host_name,
 AVG(R.review_rating) AS average_rating
FROM  Hosts AS H
INNER JOIN Listings AS L USING(host_id)
LEFT JOIN Reviews AS R USING(listing_id)
GROUP BY
 H.host_id;
```

- SELECT * FROM AVG_RATING;

OUTPUT:



Result Grid | Filter Rows: | Export:

host_id	host_name	average_rating
101	John Smith	4.900000
102	Emily Johnson	4.500000
103	Michael Brown	4.800000
104	Sarah Davis	4.900000
105	David Wilson	4.700000
106	Lisa Jones	4.500000
107	Robert Miller	4.800000
108	Mary Taylor	4.600000
109	James Johnson	4.500000
110	Linda Harris	4.700000
111	Christopher A...	4.600000
112	Karen White	4.800000
113	William Clark	4.900000
114	Jennifer Lee	4.900000
115	Richard Harris	4.700000

Result 23

Host Response Time and Ratings: By joining the Hosts, Listings, and Reviews tables, we can analyze how host response time relates to guest ratings and reviews.

- **SELECT**

```
  H.host_id,
  H.host_name,
  L.listing_id,
  L.property_type,
  L.city,
  H.host_response_time,
  R.review_rating,
  R.review_text
FROM
  Hosts AS H
INNER JOIN
  Listings AS L ON H.host_id = L.host_id
LEFT JOIN
  Reviews AS R ON L.listing_id = R.listing_id
ORDER BY
 H.host_id, L.listing_id;
 H.host_id;
```

- **SELECT * FROM Response_time;**

OUTPUT:

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

host_id	host_name	listing_id	property_type	city	host_response_time	review_rating	review_text
101	John Smith	1	Apartment	New York	Within an hour	4.90	Great place to stay!
102	Emily Johnson	2	House	Los Angeles	Within a few hours	4.50	Nice host and comfortable room.
103	Michael Brown	3	Studio	San Francisco	Within a day	4.80	Enjoyed my stay here. Clean and cozy.
104	Sarah Davis	4	Villa	Miami	Within a few hours	4.90	Great villa, perfect for a family.
105	David Wilson	5	Cabin	Asheville	Within an hour	4.70	Beautiful cabin in a stunning location.
106	Lisa Jones	6	Apartment	Chicago	Within an hour	4.50	The host was very accommodating.
107	Robert Miller	7	House	Austin	Within a day	4.80	Lovely house with a great view.
108	Mary Taylor	8	Condo	Seattle	Within a few hours	4.60	Wonderful condo in a convenient location.
109	James Johnson	9	Apartment	Paris	Within an hour	4.50	Charming apartment in the heart of Paris.
110	Linda Harris	10	House	London	Within a day	4.70	Loved the house in Notting Hill.
111	Christopher A...	11	Cottage	Sydney	Within a day	4.60	Cozy cottage by the beach.
112	Karen White	12	Apartment	Barcelona	Within a few hours	4.80	Clean apartment in a historic neighborhood.
113	William Clark	13	House	Tokyo	Within a few hours	4.90	Great villa in the heart of Tokyo.
114	Jennifer Lee	14	Villa	Bali	Within an hour	4.90	The view from this villa is breathtaking.
115	Richard Harris	15	Apartment	Rome	Within a day	4.70	Comfortable apartment in a great location.

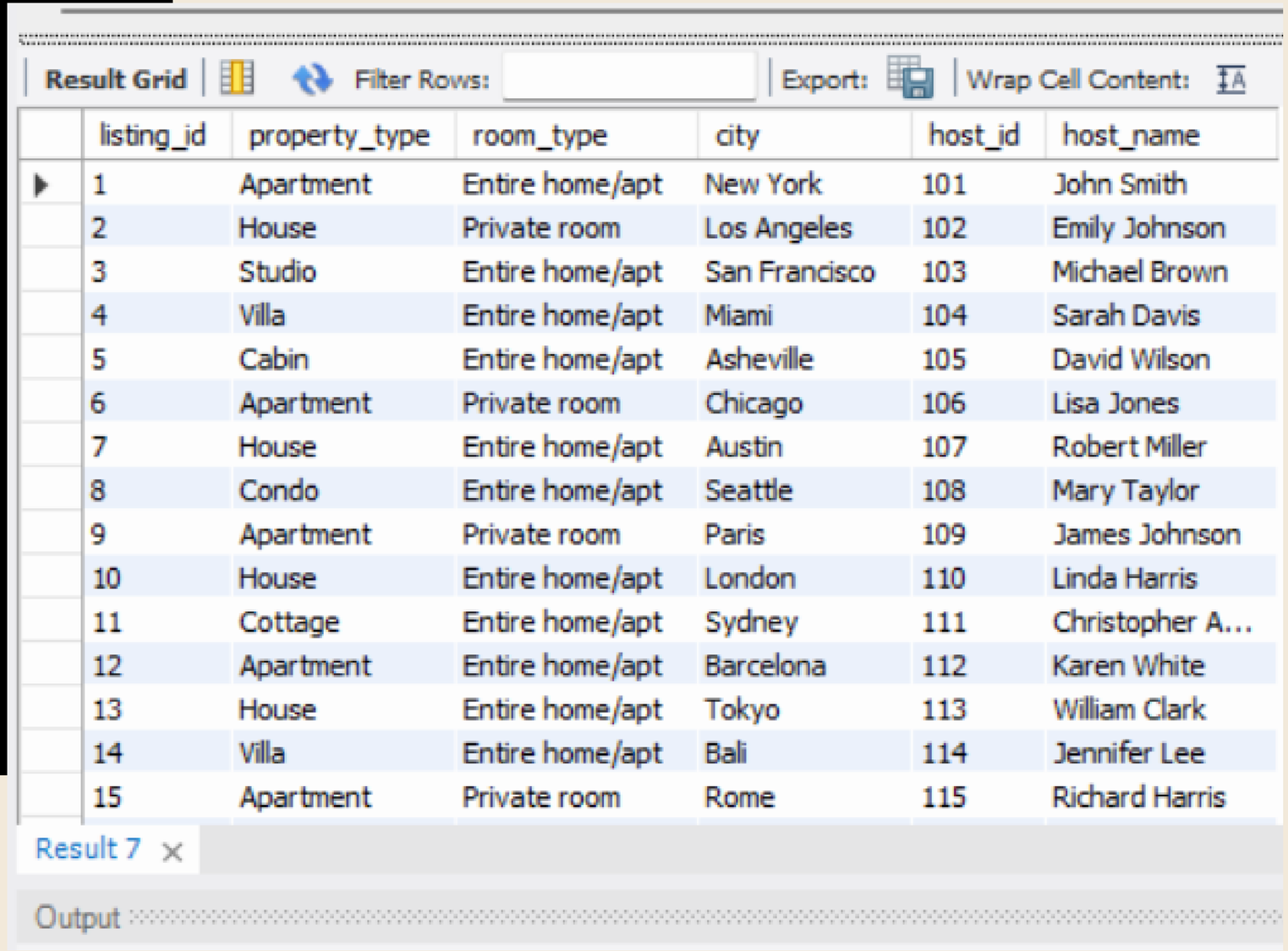
Result 25

Output

Listings by City: a JOIN between the Listings and Hosts tables using the city column to see a list of properties in each city along with the respective hosts.

```
SELECT
 L.listing_id,
 L.property_type,
 L.room_type,
 L.city,
 H.host_id,
 H.host_name
FROM
 Listings AS L
INNER JOIN Hosts AS H ON L.host_id = H.host_id
ORDER BY
 L.listing_id;
SELECT * FROM list_of_properties;
```

OUTPUT:



Result Grid | Filter Rows: | Export: | Wrap Cell Content:

listing_id	property_type	room_type	city	host_id	host_name
1	Apartment	Entire home/apt	New York	101	John Smith
2	House	Private room	Los Angeles	102	Emily Johnson
3	Studio	Entire home/apt	San Francisco	103	Michael Brown
4	Villa	Entire home/apt	Miami	104	Sarah Davis
5	Cabin	Entire home/apt	Asheville	105	David Wilson
6	Apartment	Private room	Chicago	106	Lisa Jones
7	House	Entire home/apt	Austin	107	Robert Miller
8	Condo	Entire home/apt	Seattle	108	Mary Taylor
9	Apartment	Private room	Paris	109	James Johnson
10	House	Entire home/apt	London	110	Linda Harris
11	Cottage	Entire home/apt	Sydney	111	Christopher A...
12	Apartment	Entire home/apt	Barcelona	112	Karen White
13	House	Entire home/apt	Tokyo	113	William Clark
14	Villa	Entire home/apt	Bali	114	Jennifer Lee
15	Apartment	Private room	Rome	115	Richard Harris

Result 7

Output

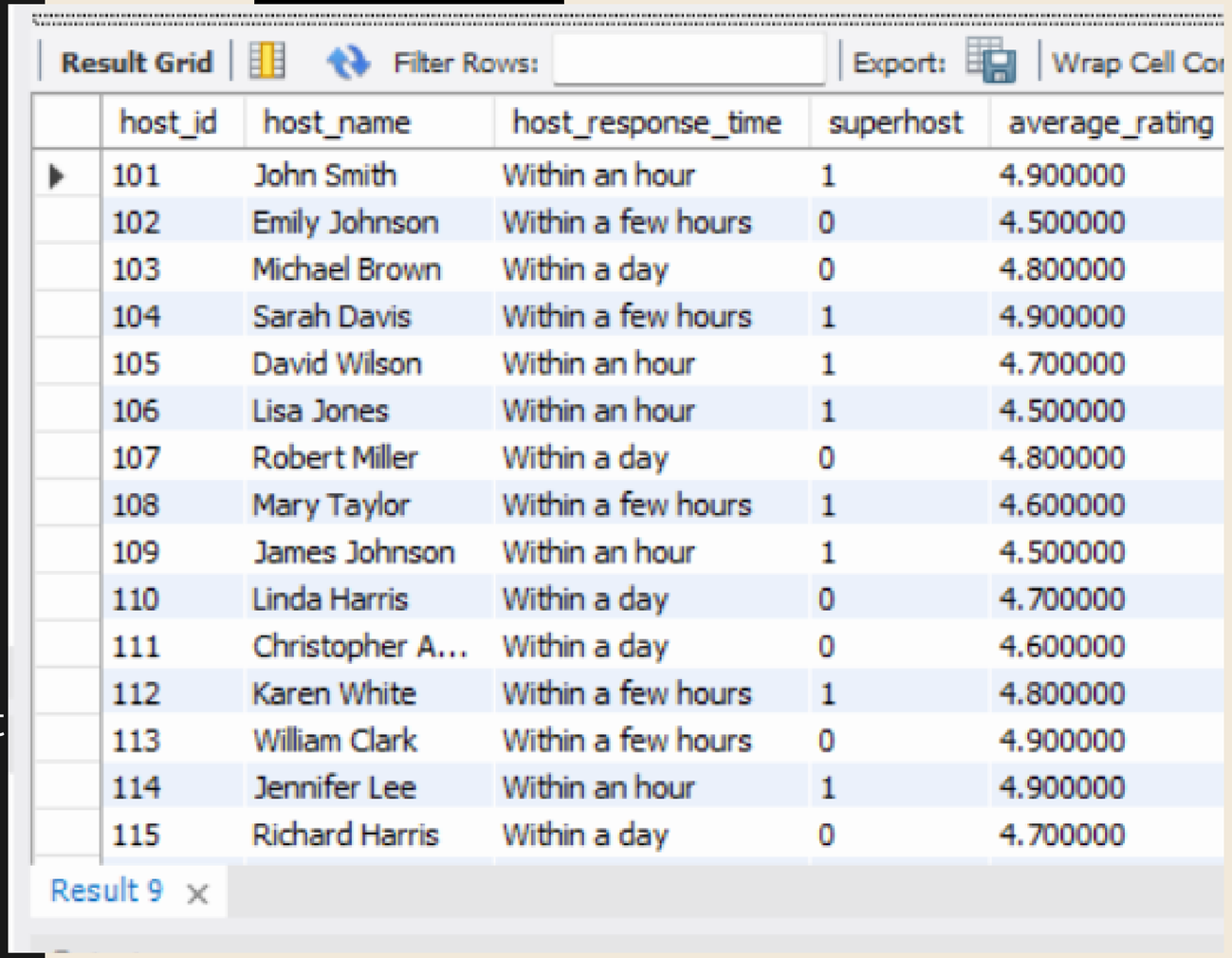
What is the average rating for each host, and how does it correlate with their response time and superhost status.

- SELECT

```
H.host_id,
H.host_name,
H.host_response_time,
H.superhost,
AVG(R.review_rating) AS average_rating
FROM
Hosts AS H
LEFT JOIN Listings AS L USING(Host_id)
LEFT JOIN Reviews AS R USING(Listing_id)
GROUP BY
H.host_id, H.host_name, H.host_response_time, H.superhost
ORDER BY
H.host_id;
```

- SELECT * FROM hosts;

OUTPUT:



Result Grid | Filter Rows: | Export: | Wrap Cell Co

host_id	host_name	host_response_time	superhost	average_rating
101	John Smith	Within an hour	1	4.900000
102	Emily Johnson	Within a few hours	0	4.500000
103	Michael Brown	Within a day	0	4.800000
104	Sarah Davis	Within a few hours	1	4.900000
105	David Wilson	Within an hour	1	4.700000
106	Lisa Jones	Within an hour	1	4.500000
107	Robert Miller	Within a day	0	4.800000
108	Mary Taylor	Within a few hours	1	4.600000
109	James Johnson	Within an hour	1	4.500000
110	Linda Harris	Within a day	0	4.700000
111	Christopher A...	Within a day	0	4.600000
112	Karen White	Within a few hours	1	4.800000
113	William Clark	Within a few hours	0	4.900000
114	Jennifer Lee	Within an hour	1	4.900000
115	Richard Harris	Within a day	0	4.700000

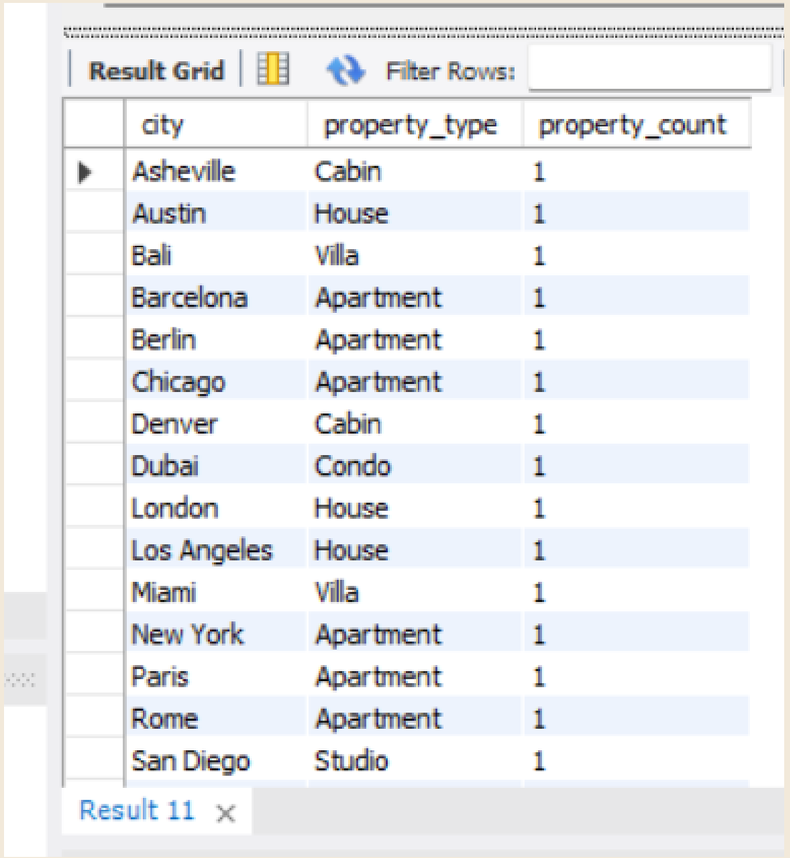
Result 9

In which cities are Airbnb properties most prevalent, and what types of properties are popular in each location

```
SELECT
 L.city,
 L.property_type,
 COUNT(*) AS property_count
FROM
 Listings AS L
GROUP BY
 L.city, L.property_type
ORDER BY
 L.city, property_count DESC;
```

These queries analyze seasonal trends in property availability and pricing, allowing us to identify peak booking periods, high-demand seasons, and pricing variations throughout the year.

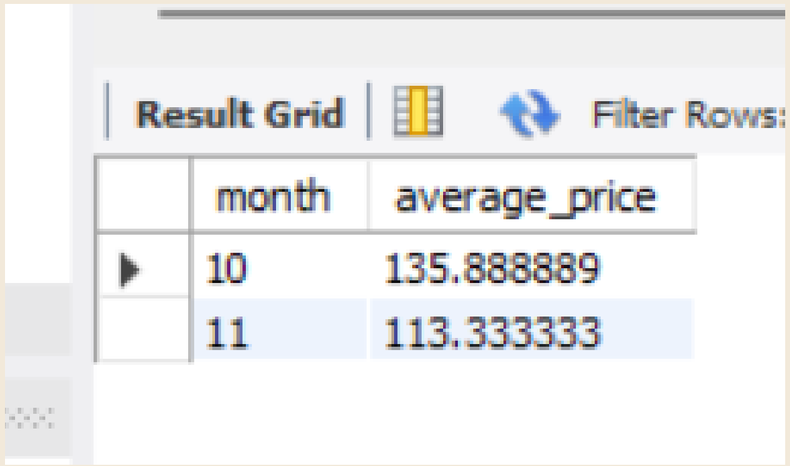
```
SELECT
 EXTRACT(MONTH FROM available_date) AS month,
 AVG(price) AS average_price
FROM
 Listings AS L
JOIN
 Availability AS A ON L.listing_id = A.listing_id
GROUP BY
 EXTRACT(MONTH FROM available_date)
ORDER BY
 month;
```



Result Grid | Filter Rows:

city	property_type	property_count
Asheville	Cabin	1
Austin	House	1
Bali	Villa	1
Barcelona	Apartment	1
Berlin	Apartment	1
Chicago	Apartment	1
Denver	Cabin	1
Dubai	Condo	1
London	House	1
Los Angeles	House	1
Miami	Villa	1
New York	Apartment	1
Paris	Apartment	1
Rome	Apartment	1
San Diego	Studio	1

Result 11



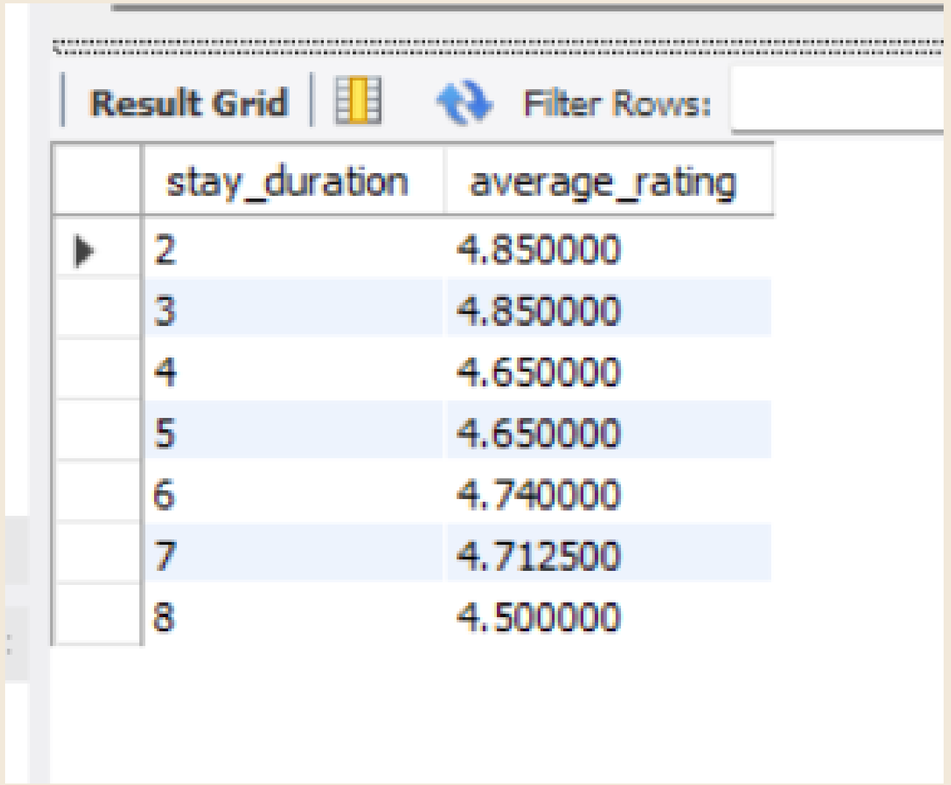
Result Grid | Filter Rows:

month	average_price
10	135.888889
11	113.333333

This query analyze whether guests tend to leave more positive reviews
when they stay for longer periods. We can assess whether there is a correlation
between the length of the stay and the average review rating, which can provide
insights into guest satisfaction.

```
SELECT
 DATEDIFF(checkout_date, checkin_date) AS stay_duration,
 AVG(review_rating) AS average_rating
FROM
 Booking AS B
JOIN
 Reviews AS R ON B.booking_id = R.booking_id
GROUP BY
 stay_duration
ORDER BY
 stay_duration;
```

OUTPUT:



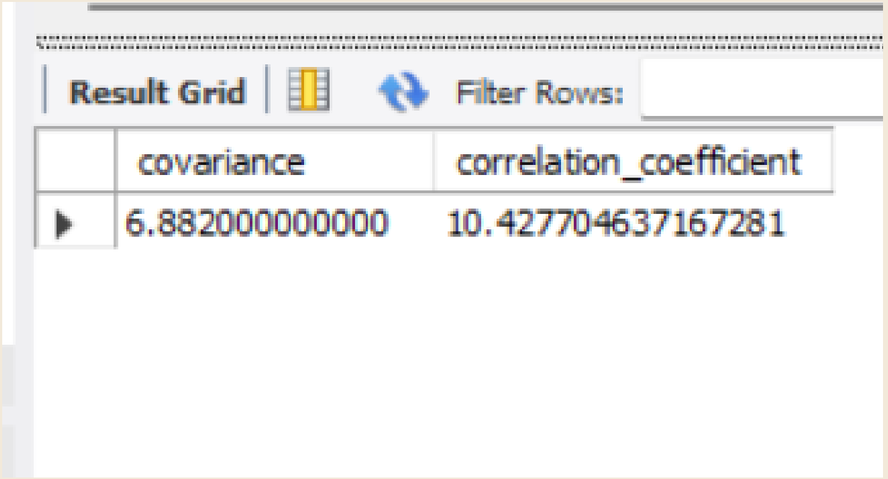
stay_duration	average_rating
2	4.850000
3	4.850000
4	4.650000
5	4.650000
6	4.740000
7	4.712500
8	4.500000

This query determine whether there is a correlation between
pricing and guest ratings, allowing us to assess whether guests are more likely to leave
positive reviews for higher-priced listings or vice versa.

```
SELECT
 AVG(price * review_rating) - AVG(price) *
AVG(review_rating) AS covariance,
 SQRT((AVG(price * price) - AVG(price) * AVG(price)) *
(AVG(review_rating *
 review_rating) - AVG(review_rating) * AVG(review_rating)))
AS correlation_coefficient

FROM
   Listings AS L
JOIN
   Reviews AS R ON L.listing_id = R.listing_id;
```

OUTPUT:


Result Grid | Filter Rows:

	covariance	correlation_coefficient
▶	6.882000000000	10.427704637167281

SUMMARY OF THE PROJECT:

Steps Taken:

- **Requirement Analysis:** The project began with a thorough analysis of framework, and what questions we want to solve and who are our stakeholders.

- **Database Design:** Based on the gathered requirements, an Entity-Relationship Diagram (ERD) was created to represent the relationships between different entities like booking, listing, hosts, review.

- **Schema Creation:** Using the ERD as a guide, the database schema was created. This involved defining tables, their attributes, relationships, and constraints.

- **Data Population:** Dummy data was generated and inserted into the database to simulate real-world scenarios. This helped in testing the system's functionality.

- **Query Development:**
  Created various Data Manipulation Language (DML) queries for data retrieval, insertion, updating, and deletion.
  Formulated Data Query Language (DQL) queries for retrieving specific information from the database.

Benefits of the System:

- Improved Efficiency: The system automated many manual processes, reducing paperwork and administrative overhead.

- Enhanced Data Accuracy: With structured data entry forms, the chances of errors and inconsistencies were significantly reduced

- Real-time Reporting: The system provided instant access to key performance indicators, enabling timely decision-making.