Group #: 5

Akanksha Jagdale

Surmayi Shree

Aditi Deshpande

**SQL-Mongo Project – Seattle Airbnb Open Dataset**

BUAN 6320

|  |  |  |  |
| --- | --- | --- | --- |
| **Activity** | **Akanksha Jagdale** | **Surmayi Shree** | **Aditi Deshpande** |
| Prepared Data Model and Created Physical DB | x |  |  |
| Loaded Data into Database |  | x |  |
| Wrote SQL Queries | x | x | x |
| Prepared Mongo Database |  |  | x |
| Loaded data into Mongo DB |  | x |  |
| Wrote Mongo Queries | x | x | x |
| Prepared Report |  |  | x |
| Reviewed Report | x | x |  |

Contents

Relational Data Model 4

Assumptions/Notes About Data Entities and Relationships 4

Entity-Relationship Diagram 5

Physical MySQL Database 6

Assumptions/Notes About Data Set 6

Screen shot of Physical Database objects 6

Data in the Database 11

SQL Queries 16

SQL Query 1 16

Question 16

Notes/Comments About SQL Query and Results (Include # of Rows in Result) 16

Translation/Cleanup 16

Screen Shot of SQL Query and Results 16

SQL Query 2 17

Question 17

Notes/Comments About SQL Query and Results (Include # of Rows in Result) 17

Translation/Cleanup 17

Screen Shot of SQL Query and Results 17

SQL Query 3 18

Question 18

Notes/Comments About SQL Query and Results (Include # of Rows in Result) 18

Translation/Cleanup 18

Screen Shot of SQL Query and Results 18

SQL Query 4 19

Question 19

Notes/Comments About SQL Query and Results (Include # of Rows in Result) 19

Translation/Cleanup 19

Screen Shot of SQL Query and Results 19

SQL Query 5 20

Question 20

Notes/Comments About SQL Query and Results (Include # of Rows in Result) 20

Translation/Cleanup 20

Screen Shot of SQL Query and Results 20

Data Review for MongoDB 21

Assumptions/Notes About Data Collections, Attributes and Relationships between Collections 21

Physical Mongo Database 22

Assumptions/Notes About Data Set 22

Screen shot of Physical Database objects (Database, Collections and Attributes) 22

Data in the Database 22

MongoDB Queries/Code 23

Mongo Query 1 23

Question 23

Notes/Comments About MongoDB Query/Code and Results (Include # of Documents in Result) 23

Translation 23

Screen Shot of MongoDB Query/Code and Results 23

Mongo Query 2 25

Question 25

Notes/Comments About MongoDB Query/Code and Results (Include # of Documents in Result) 25

Translation 25

Screen Shot of MongoDB Query/Code and Results 25

Mongo Query 3 27

Question 27

Notes/Comments About MongoDB Query/Code and Results (Include # of Documents in Result) 27

Translation 27

Screen Shot of MongoDB Query/Code and Results 27

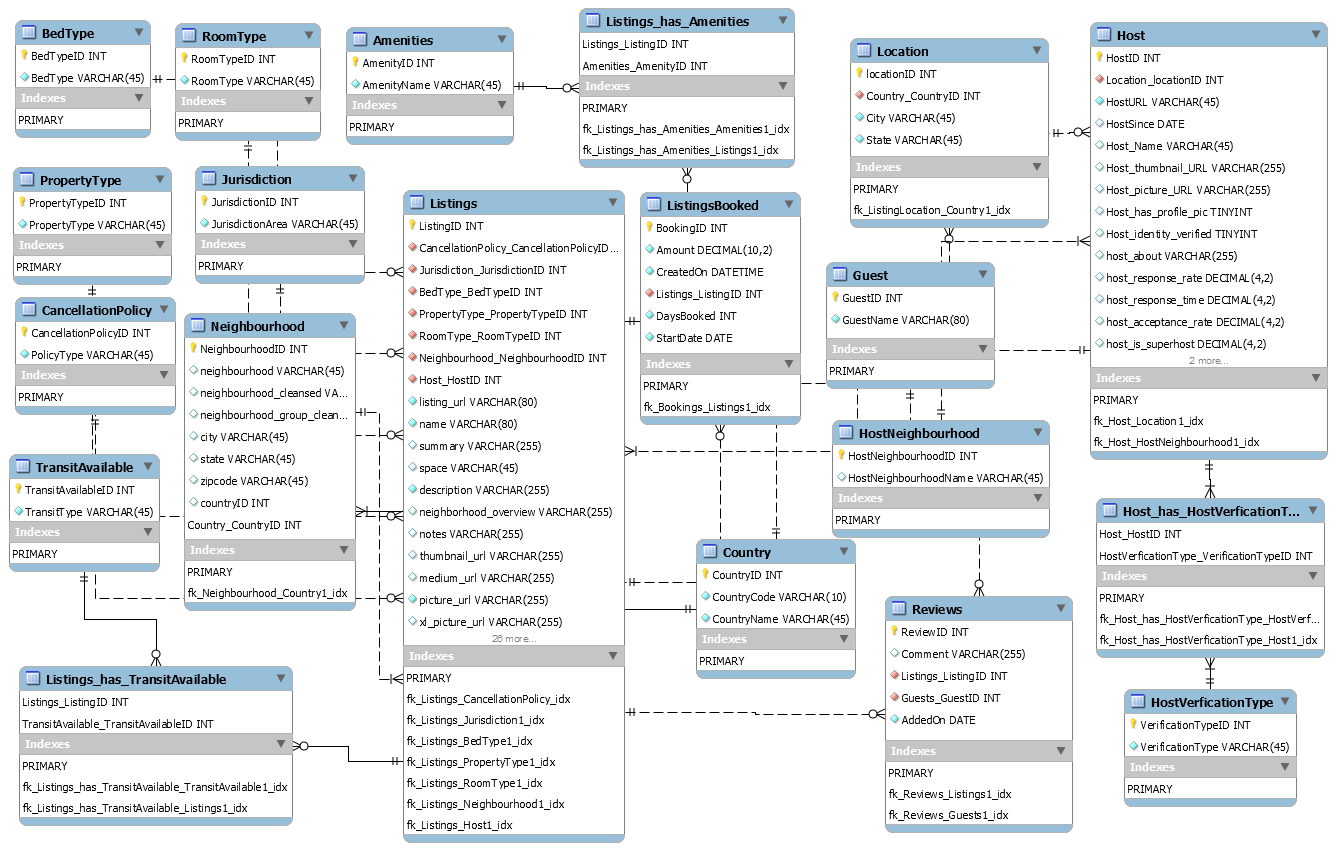
# **Relational Data Model**

## Assumptions/Notes about Data Entities and Relationships

|  |  |  |  |
| --- | --- | --- | --- |
| **Sr. NO** | **Entity** | **Description** | **Assumption** |
| 1 | Listings | Contains all the information about the listings which are available and the attributes related to it |  |
| 2 | Amenities | Contains the amenity name and the amenity ID which corresponds to each listing | One amenity can have many listings and one listing can have many amenities |
| 3 | Listings\_has\_amenities | Contains information about the listing ID and amenity ID in a many to many relationship |  |
| 4 | Transit Available | Contains the details of the transit type with respect to listings | One listing can have many transits and one transit can have many listings |
| 5 | Listings\_has\_transit\_available | Contains information about the listing ID and transit available ID, in a many to many relationship |  |
| 6 | Listings\_booked | Contains all the information about the bookings which are booked by guests in Airbnb | One listing can have many listings\_booked and one listings\_booked can have one listing |
| 7 | Country | Contains information about the country where the listings are located | One Country can have many Locations but one location can be a part of only one country. |
| 8 | Host | Details about the Hosts working in various locations. | One listing can have one host and one host can have many listings |
| 10 | Host\_has\_HostVerificationType | With respect to many to many relationship between Host and HostVerificationType |  |
| 11 | Locations | Details about the location of the host. | One Host can have one location and one location can have many Hosts |
| 12 | Guest | Details of all the guests. | One guest can give multiple reviews but one review will be associated to only One Guest. |
| 13 | Reviews | Details about the reviews given by each guest for a particular listing | One listing can have many reviews and one review can have one listing |
| 14 | BedType | Details about types of beds | One listing can have one bed type and one bed type can have many listings |
| 15 | PropertyType | Details about the different types of properties | One listing can have one property type and one property type can have many listings |
| 16 | RoomType | Details of different types of rooms available in the apartments | One listing can have one room type and one room type can have many listings |
| 17 | CancellationPolicy | Details of cancellation policy | One listing can have one cancellation policy and one cancellation policy can have many listings |
| 18 | Jurisdiction | Details about the jurisdiction of the state | One listing can have one jurisdiction and one jurisdiction can have many listings |
| 19 | Neighbourhood | Contains details about the neighbourhood of the listings | One listing can have one neighbourhood and one neighbourhoods can have many listings |
| 20 | HostNeighbourhood | Contains details about the neighborhood of the host | One Host can have only one neighbourhood but one neighbourhood can have many Hosts |

## 

## Entity-Relationship Diagram:



# Physical MySQL Database

## Assumptions/Notes about Data Set

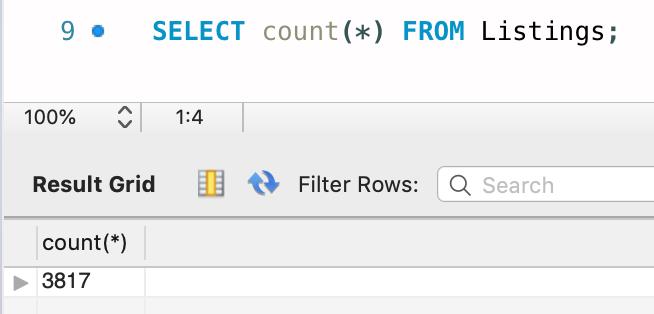
Include any assumptions made about data such as empty fields, sparse data, bad data, etc.

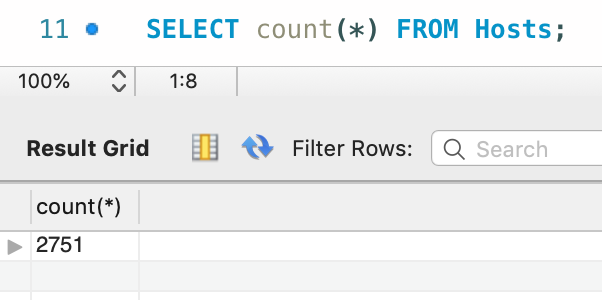
* 1. Assumption: We have included only the booked listings. To search for any available date for a listing we can join Listings and Listings\_booked for the given range and exclude date which are in listings\_booked.

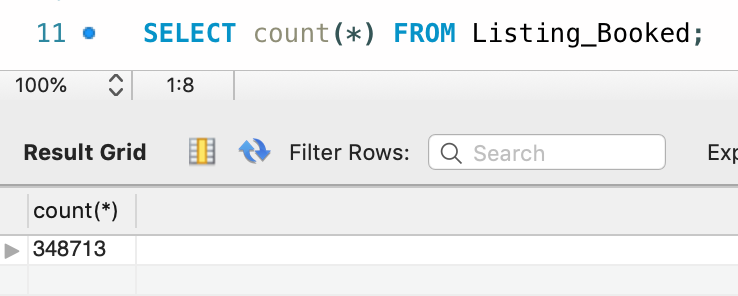
Eg: To check for availability\_30 for a particular listing, we can select 30 minus count of rows in listings\_booked for next 30 days. Similarly, we can do this for availability\_60, availability\_90, availability\_365.

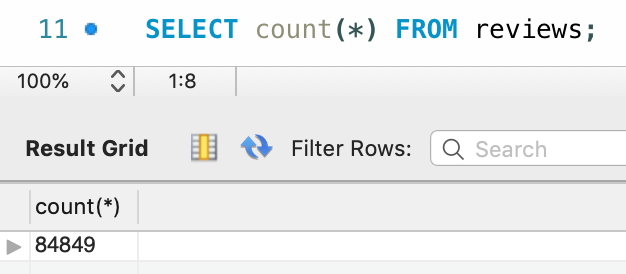
* 1. Review Count can be found by using Group By ListingID from Reviews Table. Similarly, we can find out for First\_review, Last\_review for a listing.
  2. Redundant columns – Market, smart location, last\_scraped, scrape ID, host\_listings\_count, host\_total\_listings\_count, first\_review, last\_review, reviews\_per\_month, calculated\_host\_listings\_count, availability\_30, availability\_60, availability\_90, availability\_365.
  3. Sparse\_data – GuestID not available for Listings\_Booked.
  4. Unknown/Empty: lisence, experiences\_offered

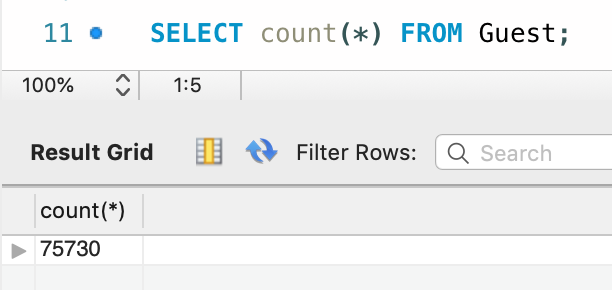
## Screen shot of Physical Database objects

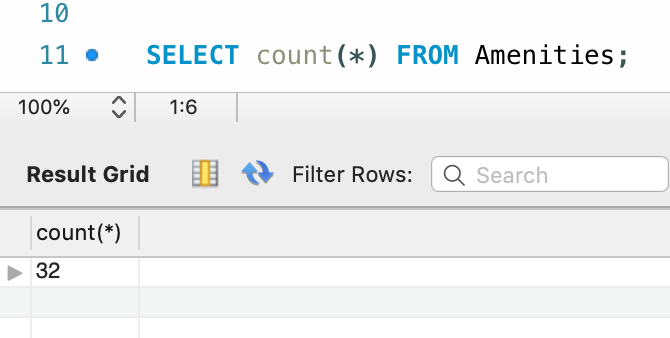


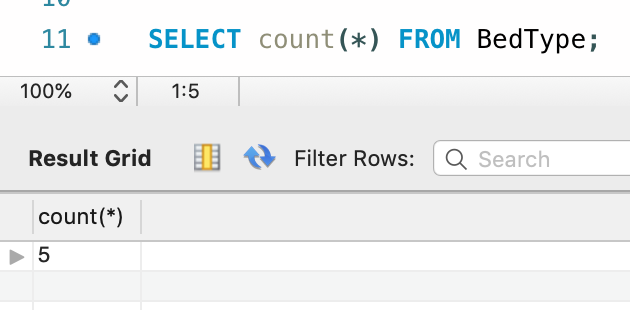


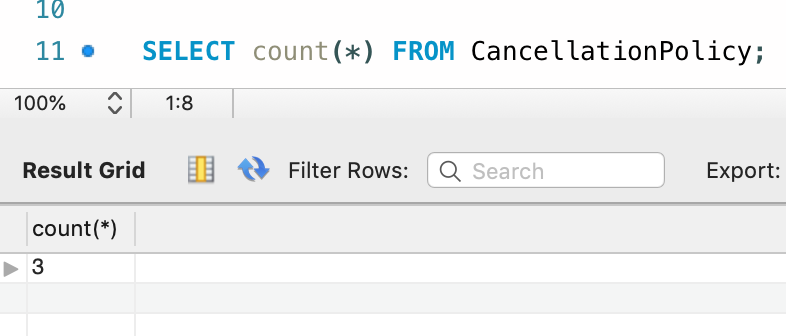


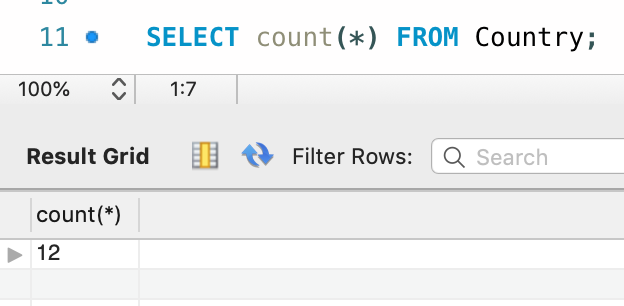


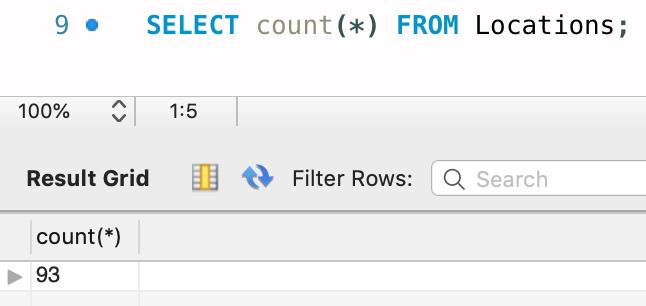


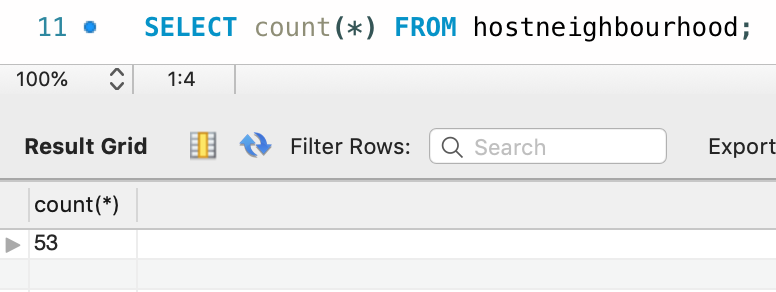


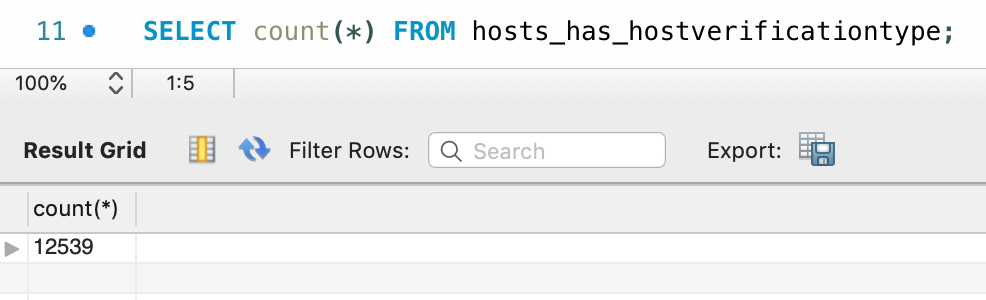


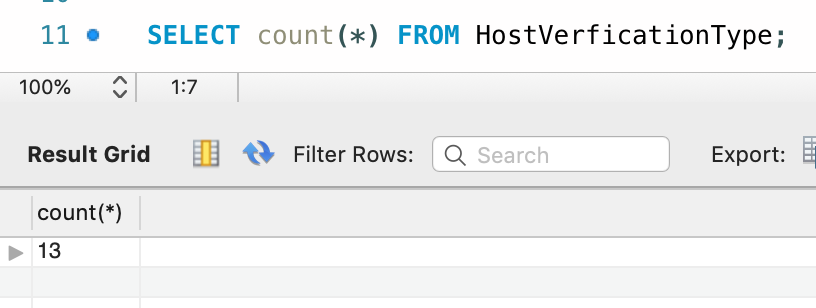


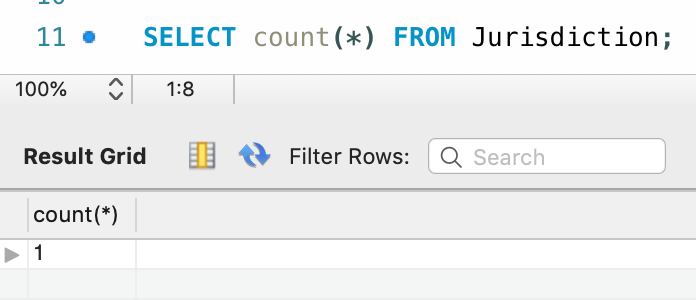


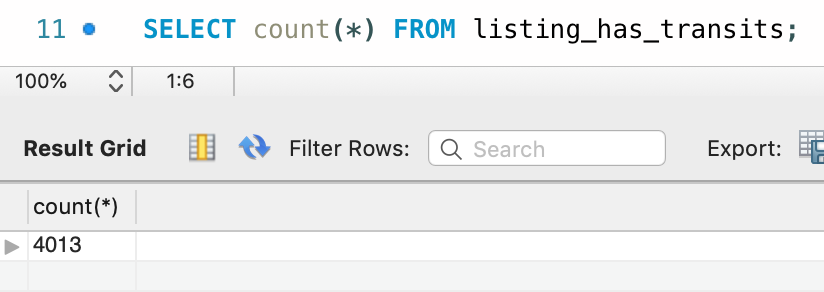


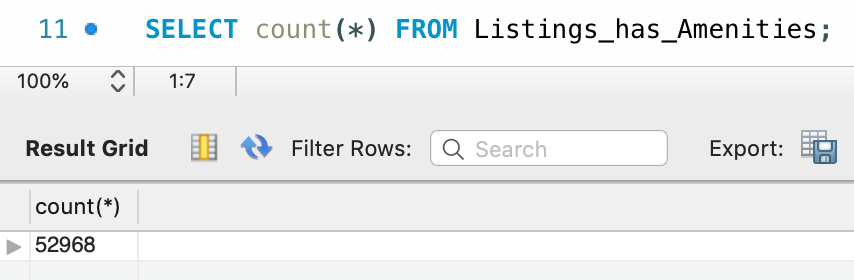


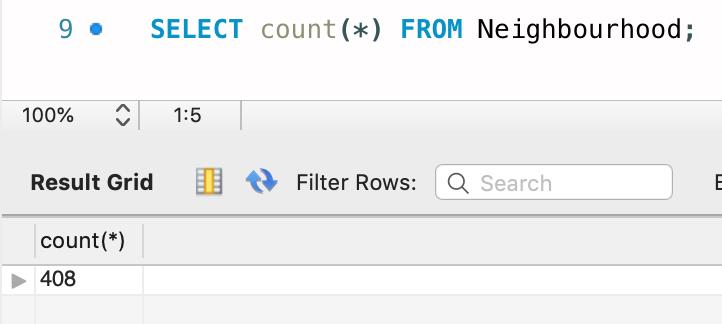


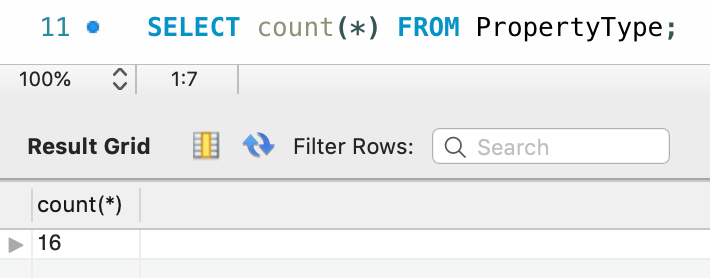


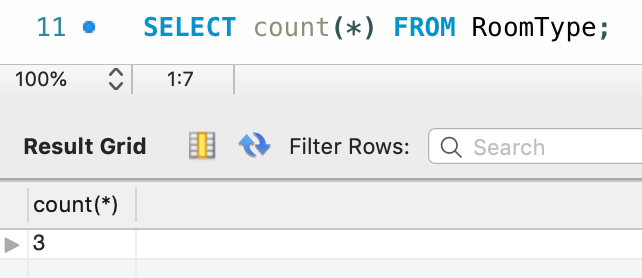


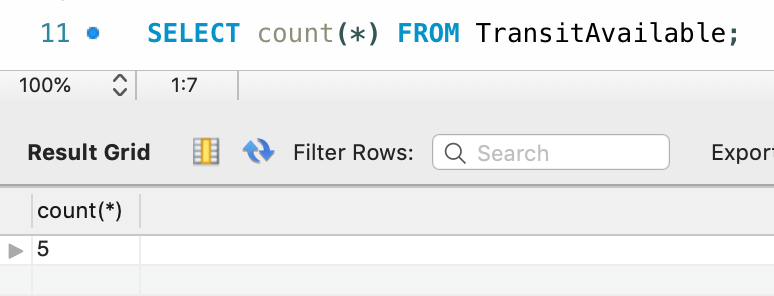












## **Data in the Database:**

|  |  |  |  |
| --- | --- | --- | --- |
| **Table Name** | **Primary Key** | **Foreign Key** | **# of Rows in Table** |
| Amenities | AmenityID |  | 32 |
| Bed Type | BedTypeID |  | 5 |
| Cancellation Policy | CancellationPolicyID |  | 3 |
| Country | CountryID |  | 12 |
| Guest | GuestID |  | 75730 |
| Location | LocationID | Country\_CountryID | 93 |
| Host\_Neighborhood | Host\_NeighborhoodID |  | 53 |
| Host | HostID | Location\_locationID  HostNeighborhood\_HostNeighborhoodID | 2751 |
| Hosts\_Has\_HostVerificationType | Hosts\_Has\_HostVerificationTypeID |  | 12539 |
| HostVerificationType | HostVerificationTypeID |  | 13 |
| Jurisdiction | JurisdictionID |  | 1 |
| Listings | ListingsID | CancellationPolicy\_CancellationPolicyID  Jurisdiction\_JurisdictionID  BedType\_BedTypeID  PropertyType\_PropertyTypeID  RoomType\_RoomTypeID  Neighbourhood\_NeighbourhoodID  Host\_HostID | 3817 |
| ListingsBooked | BookingID | Listings\_ListingID | 348713 |
| Listing\_has\_TransitAvailable | Listings\_ListingID  TransitAvailable\_TransitAvailableID |  | 4013 |
| Listings\_has\_Amenities | Listings\_ListingID  Amenities\_AmenityID |  | 52968 |
| Neighbourhood | NeighbourhoodID | Country\_CountryID | 408 |
| PropertyType | PropertyTypeID |  | 16 |
| Reviews | ReviewsID | Listings\_ListingID  Guests\_GuestID | 84849 |
| RoomType | RoomTypeID |  | 3 |
| TransitAvailable | TransitAvailableID |  | 5 |

# **SQL Queries**

## SQL Query 1

### Question-1 : Daily prices can be higher for properties with more bathrooms

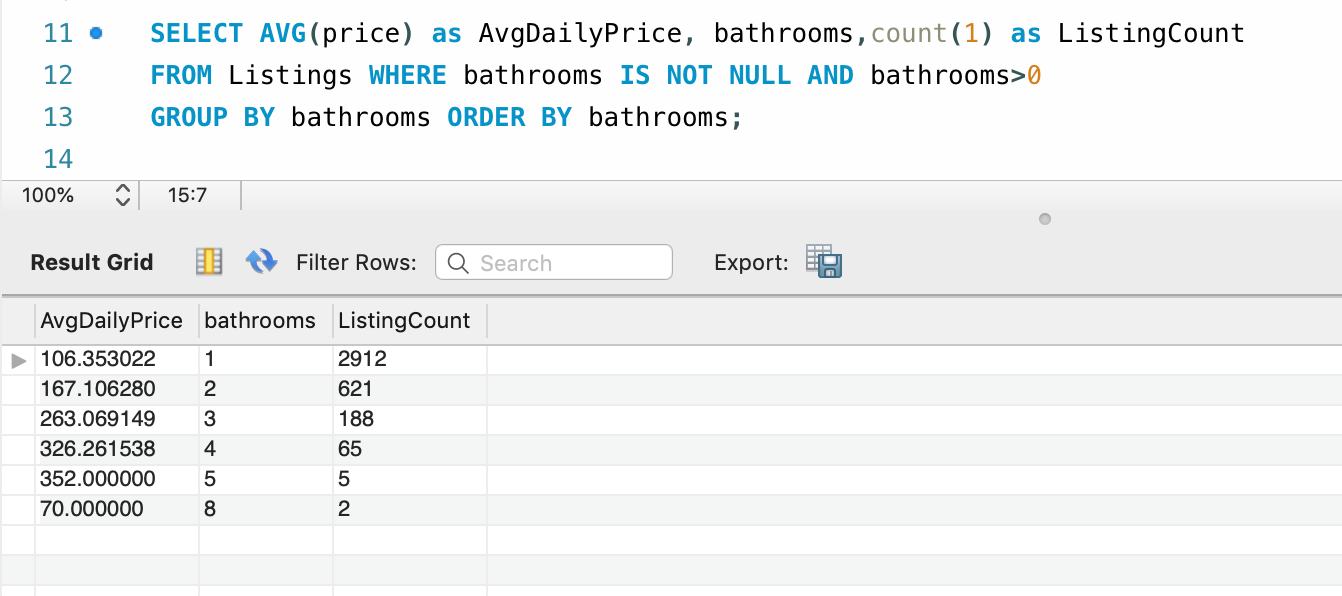
### Notes/Comments about SQL Query: As seen in the results, we prove the statement. Daily prices are higher for properties with more bathrooms with an exception of listings with 8 bathrooms.

### Results (# of Rows in Result): **6**

Translation: Select the average price and bathrooms from the Listings table where the bathrooms is not null and further group by bathrooms and order in ascending order by bathrooms. Then select the count of rows from the Listings table where the number of bathrooms is 8.

Clean-up: Select avg (price), bathrooms from Listings where bathrooms is not null group by bathrooms order by bathrooms;

### Screen Shot of SQL Query and Results:



## SQL Query 2

### Question-2: Weekly prices can be lower for properties with lesser bedrooms

### Notes/Comments about SQL Query: As seen in the results, we prove the statement.

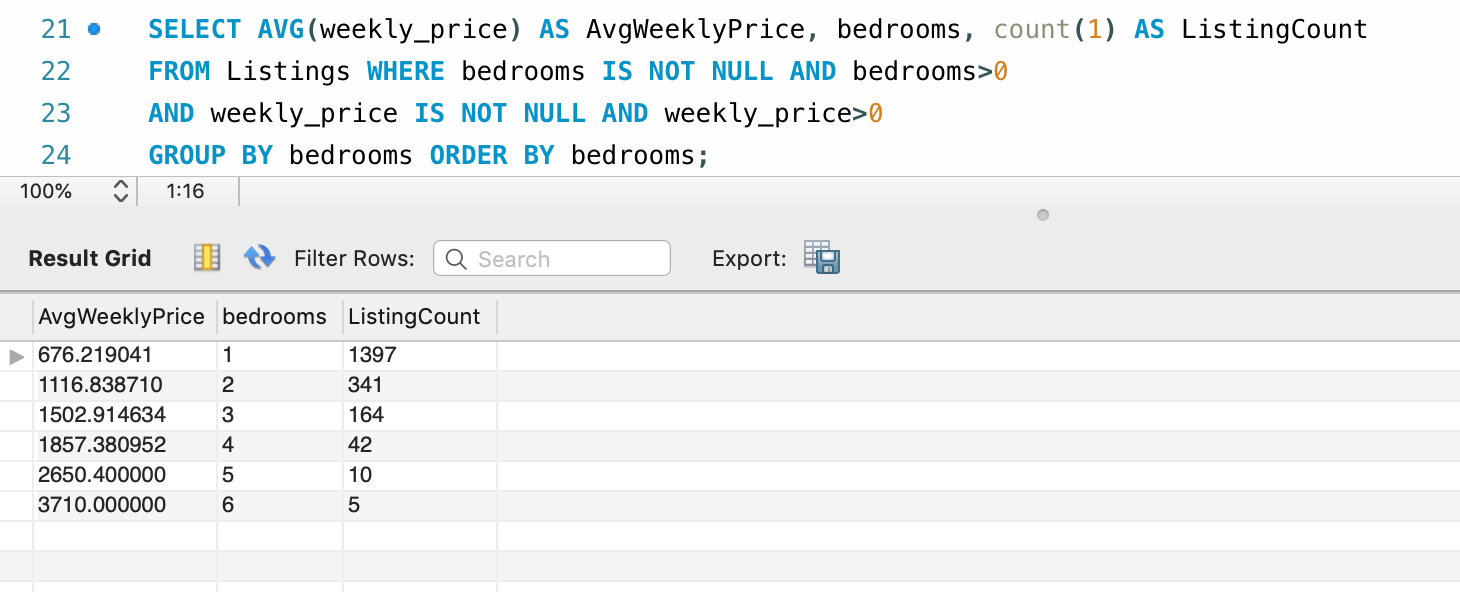
### Weekly prices are lower for properties with lesser number of bedrooms

### Results (# of Rows in Result): **6**

Translation: Select average weekly price and bedrooms from the Listings table where bedrooms and weekly price is greater than 0 and further group by bedrooms and order by bedrooms in ascending order.

Clean-up: select avg (weekly\_price), bedrooms from Listings where bedrooms>0 and weekly\_price>0 group by bedrooms order by bedrooms;

Screen Shot of SQL Query and Results:



## SQL Query 3

### Question-5: Properties with the higher review scores for location had higher occupancy rates

### Notes/Comments about SQL Query: As seen in the results, we disprove the statement. Properties with the higher review scores for location donot have higher occupancy rates.

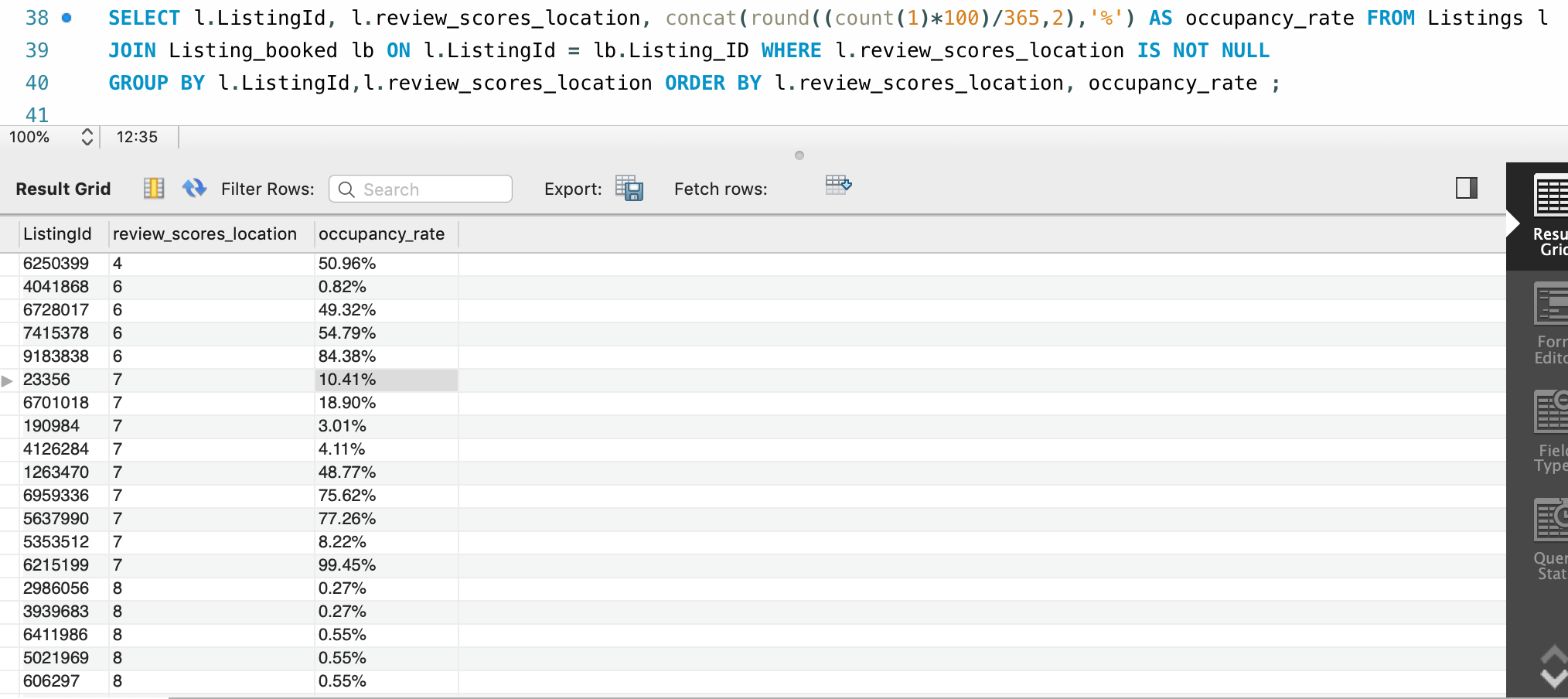
[Airbnb occupancy rate represents the percentage of days your Airbnb rental property is “occupied” by guests in a year]

### Results (# of Rows in Result): **2033**

### Translation: Select ListingId, review\_scores\_location,occupancy rate from Listings table inner joined to Listing\_booked table matching on ListingId of listings table and Listing\_ID from listings\_booked table and where review\_scores\_location is not null also group by ListingId, review\_scores\_location and order by review\_scores\_location

Clean-up :Select l.ListingId, l.review\_scores\_location, concat(round((count(1)\*100)/365,2),'%') as occupancy\_rate from Listings l join Listing\_booked lb on l.ListingId = lb.Listing\_ID where l.review\_scores\_location is not null group by l.ListingId,l.review\_scores\_location order by l.review\_scores\_location ;

### Screen Shot of SQL Query and Results :



## SQL Query 4

### Question-7: Properties with higher daily prices also charge higher security deposits and cleaning fees

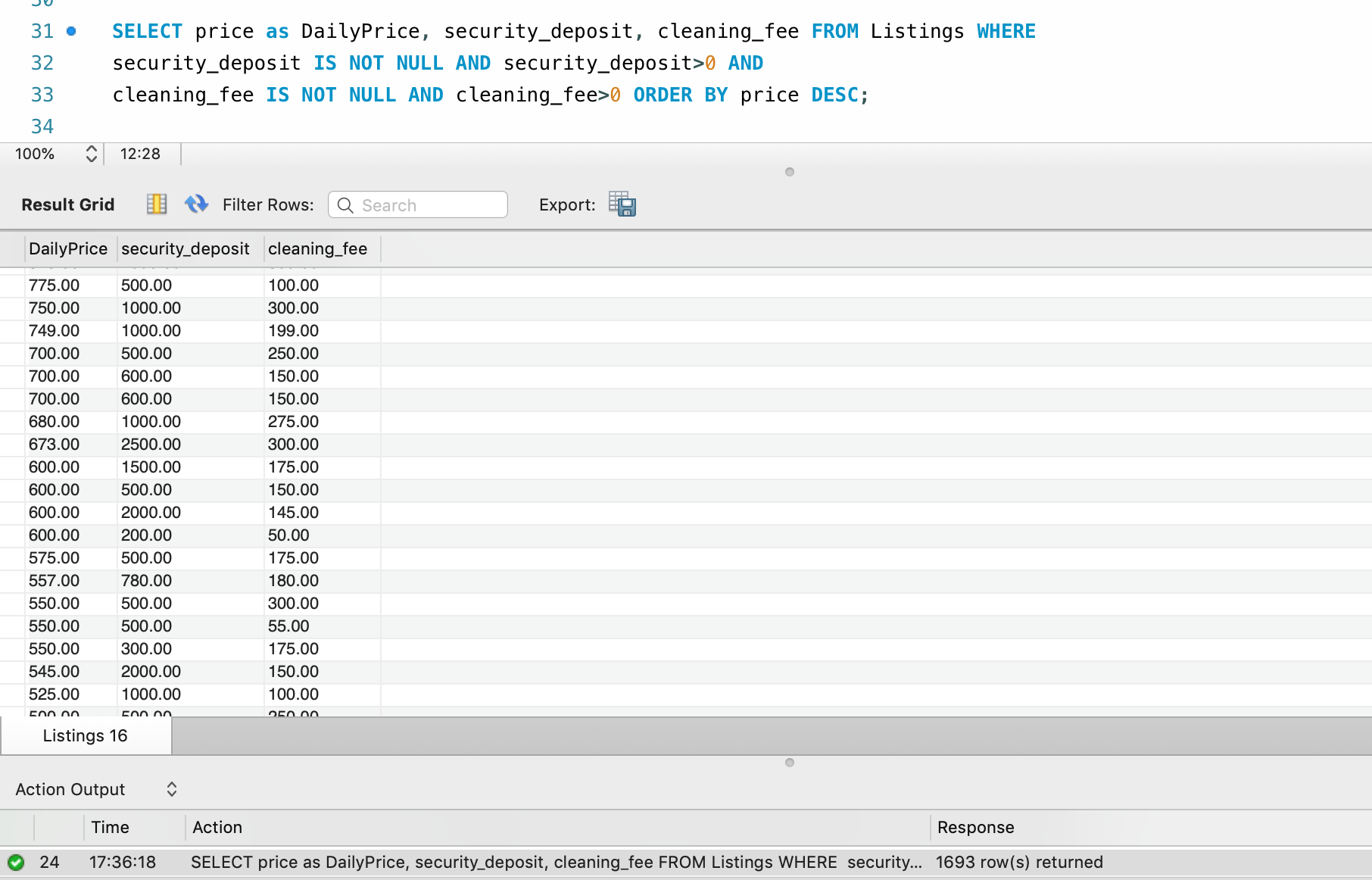
### Notes/Comments about SQL Query: As seen in the results, we disprove the statement. Properties with higher daily prices donot charge higher security deposits and cleaning fees

### Results (# of Rows in Result): **1693**

Translation: Select price, security deposit, and cleaning fee from Listings table where the  
security deposit and cleaning fee is not null and order by descending price

Clean-up: Select price, security\_deposit, cleaning\_fee from Listings where  
security\_deposit is not null and cleaning\_fee is not null order by price desc;

### Screen Shot of SQL Query and Results :



## SQL Query 5

### Question-8: Properties allowing higher number of extra people have higher occupancy rates

### Notes/Comments about SQL Query: As seen in the results, we disprove the statement. Properties allowing higher number of extra people donot have higher occupancy rates.

### Results (# of Rows in Result): **1884**

### Translation : select ListingId, count of extrapeople, occupancy\_rate from Listings table inner

### joinned Listing\_booked table matching on ListingId from listings table and Listing\_ID from

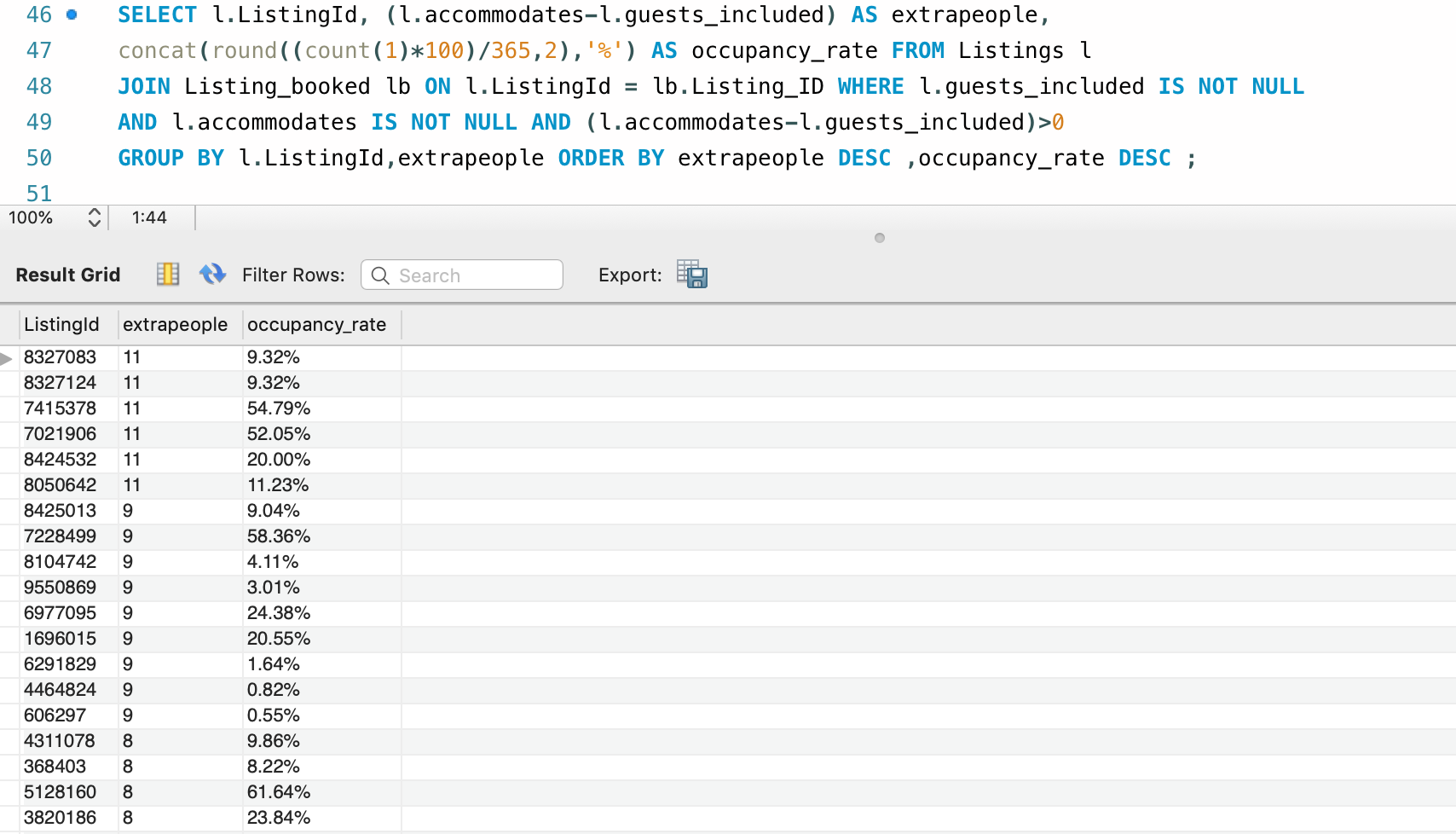
### listings\_booked table where l.guests\_included is not null and accommodates is not null

### and (l.accommodates-l.guests\_included)>-1 also group by ListingId,extrapeople and

### order by extrapeople in desc order

Clean-up : select l.ListingId, (l.accommodates-l.guests\_included) as extrapeople, concat(round((count(1)\*100)/365,2),'%') as occupancy\_rate from Listings l join Listing\_booked lb on l.ListingId = lb.Listing\_ID where l.guests\_included is not null and l.accommodates is not null and (l.accommodates-l.guests\_included)>-1 group by l.ListingId,extrapeople order by extrapeople desc;

### Screen Shot of SQL Query and Results:



# Data Review for Mongo DB

## Assumptions/Notes about Data Collections, Attributes and Relationships between Collections

Listings: This collection contains the details of listings and respective hosts.

Reviews: This collection contains the details of reviews for the listings and respective reviewers.

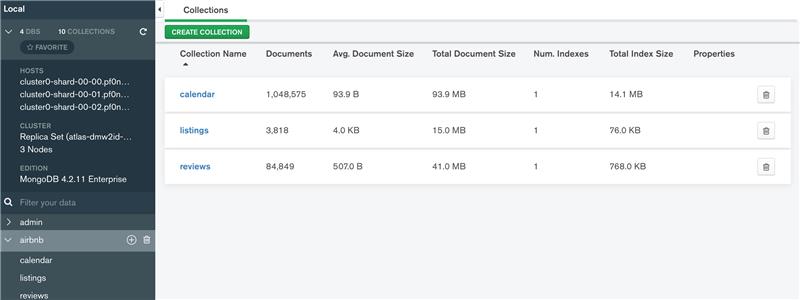
Calendar: This collection contains occupancy details for each listing for 1 year.

# Physical Mongo Database

## Assumptions/Notes about Data Set:

Raw data has been imported. So the collections are in 1NF.

## Screen shot of Physical Database objects (Database, Collections and Attributes):



## 

## Data in the Database:

|  |  |  |
| --- | --- | --- |
| **Collection Name** | **Relationships With Other Collections (if any)** | **# of Documents in Collection** |
| Listings | Review(ListingsId) | 3818 |
| Reviews | Listings(ListingsId) | 84849 |
| Calendar | Listings(ListingsId) | 1048575 |

# 

# **MongoDB Queries**

## Mongo Query 1

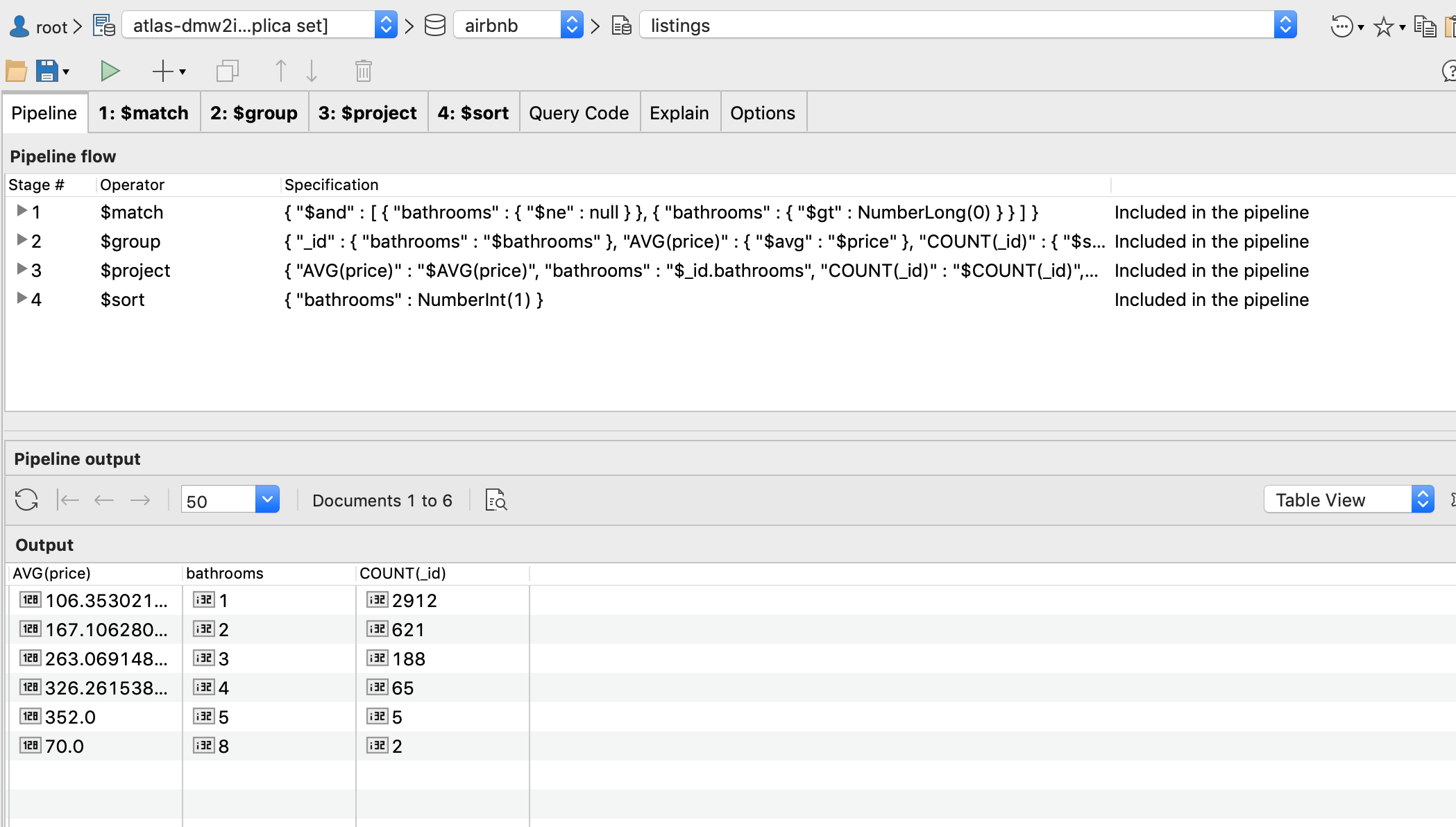
### Question-1: Daily prices can be higher for properties with more bathrooms

### Notes/Comments About MongoDB Query: As seen in the results, we prove the statement. Daily prices are higher for properties with more bathrooms with an exception of listings with 8 bathrooms.

### Results (Include # of Documents in Result): **6**

Translation: From the listings collection, get a list of average prices for each listings and the number of bathrooms.

### Screen Shot of MongoDB Query/Code and Results:



use airbnb**;**

db.getCollection**(**"listings"**)**.aggregate**(**

**[**

**{**

"$match" **:** **{**

"$and" **:** **[**

**{**

"bathrooms" **:** **{**

"$ne" **:** null

**}**

**},**

**{**

"bathrooms" **:** **{**

"$gt" **:** NumberLong**(**0**)**

**}**

**}**

**]**

**}**

**},**

**{**

"$group" **:** **{**

"\_id" **:** **{**

"bathrooms" **:** "$bathrooms"

**},**

"AVG(price)" **:** **{**

"$avg" **:** "$price"

**},**

"COUNT(\_id)" **:** **{**

"$sum" **:** NumberInt**(**1**)**

**}**

**}**

**},**

**{**

"$project" **:** **{**

"AVG(price)" **:** "$AVG(price)"**,**

"bathrooms" **:** "$\_id.bathrooms"**,**

"COUNT(\_id)" **:** "$COUNT(\_id)"**,**

"\_id" **:** NumberInt**(**0**)**

**}**

**},**

**{**

"$sort" **:** **{**

"bathrooms" **:** NumberInt**(**1**)**

**}**

**}**

**],**

**{**

"allowDiskUse" **:** true

**}**

**);**

## 

## Mongo Query 2

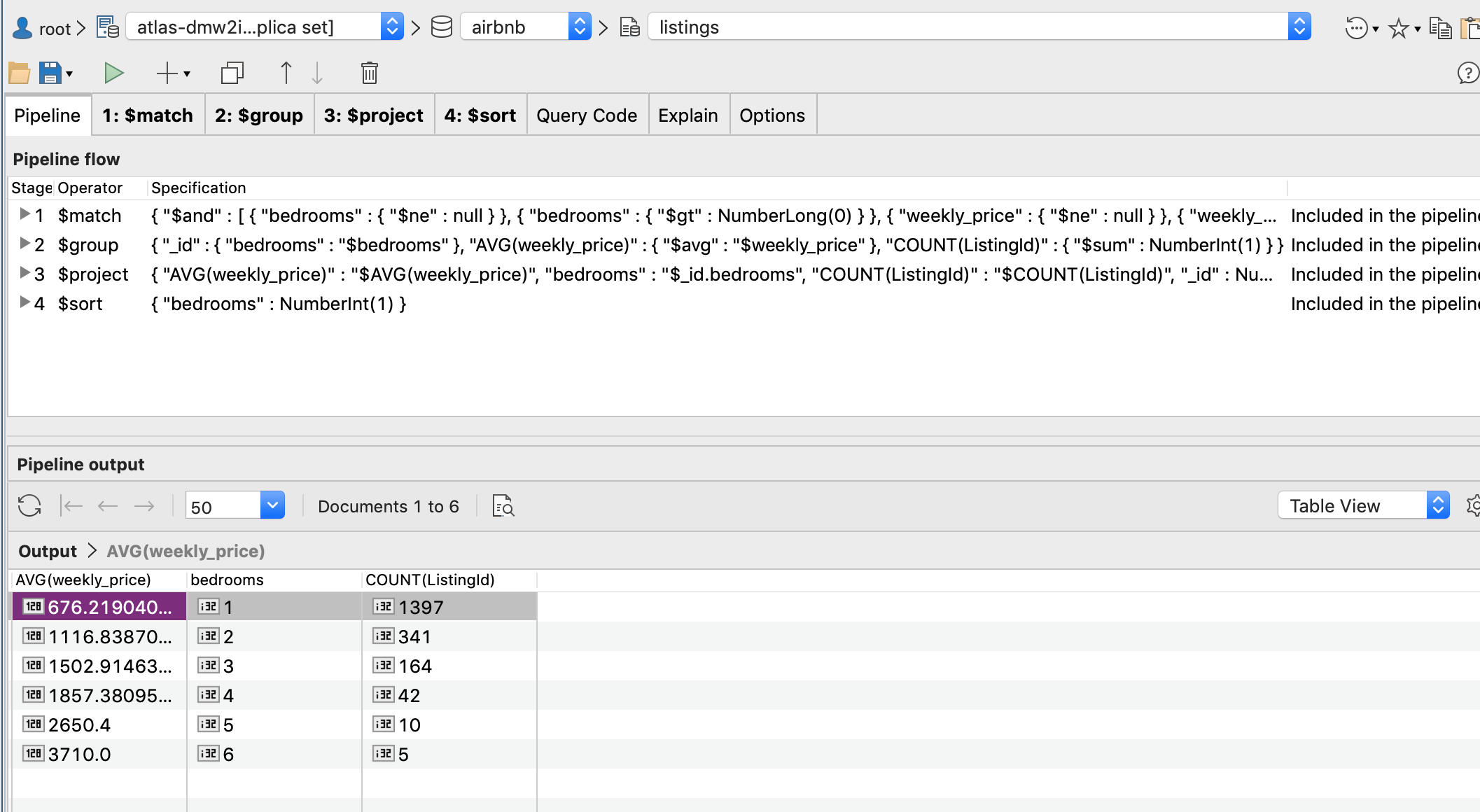
### Question-2: Weekly prices can be lower for properties with lesser bedrooms.

### Notes/Comments About MongoDB Query/Code: As seen in the results, we prove the statement. Weekly prices are lower for properties with lesser number of bedrooms

### Results (Include # of Documents in Result): 6

Translation: From the listings collection, get a list of average prices for each listings and the number of bedrooms.

### Screen Shot of MongoDB Query/Code and Results:



use airbnb**;**

db.getCollection**(**"listings"**)**.aggregate**(**

**[**

**{**

"$match" **:** **{**

"$and" **:** **[**

**{**

"bedrooms" **:** **{**

"$ne" **:** null

**}**

**},**

**{**

"bedrooms" **:** **{**

"$gt" **:** NumberLong**(**0**)**

**}**

**},**

**{**

"weekly\_price" **:** **{**

"$ne" **:** null

**}**

**},**

**{**

"weekly\_price" **:** **{**

"$gt" **:** NumberLong**(**0**)**

**}**

**}**

**]**

**}**

**},**

**{**

"$group" **:** **{**

"\_id" **:** **{**

"bedrooms" **:** "$bedrooms"

**},**

"AVG(weekly\_price)" **:** **{**

"$avg" **:** "$weekly\_price"

**},**

"COUNT(ListingId)" **:** **{**

"$sum" **:** NumberInt**(**1**)**

**}**

**}**

**},**

**{**

"$project" **:** **{**

"AVG(weekly\_price)" **:** "$AVG(weekly\_price)"**,**

"bedrooms" **:** "$\_id.bedrooms"**,**

"COUNT(ListingId)" **:** "$COUNT(ListingId)"**,**

"\_id" **:** NumberInt**(**0**)**

**}**

**},**

**{**

"$sort" **:** **{**

"bedrooms" **:** NumberInt**(**1**)**

**}**

**}**

**],**

**{**

"allowDiskUse" **:** true

**}**

**);**

## Mongo Query 3

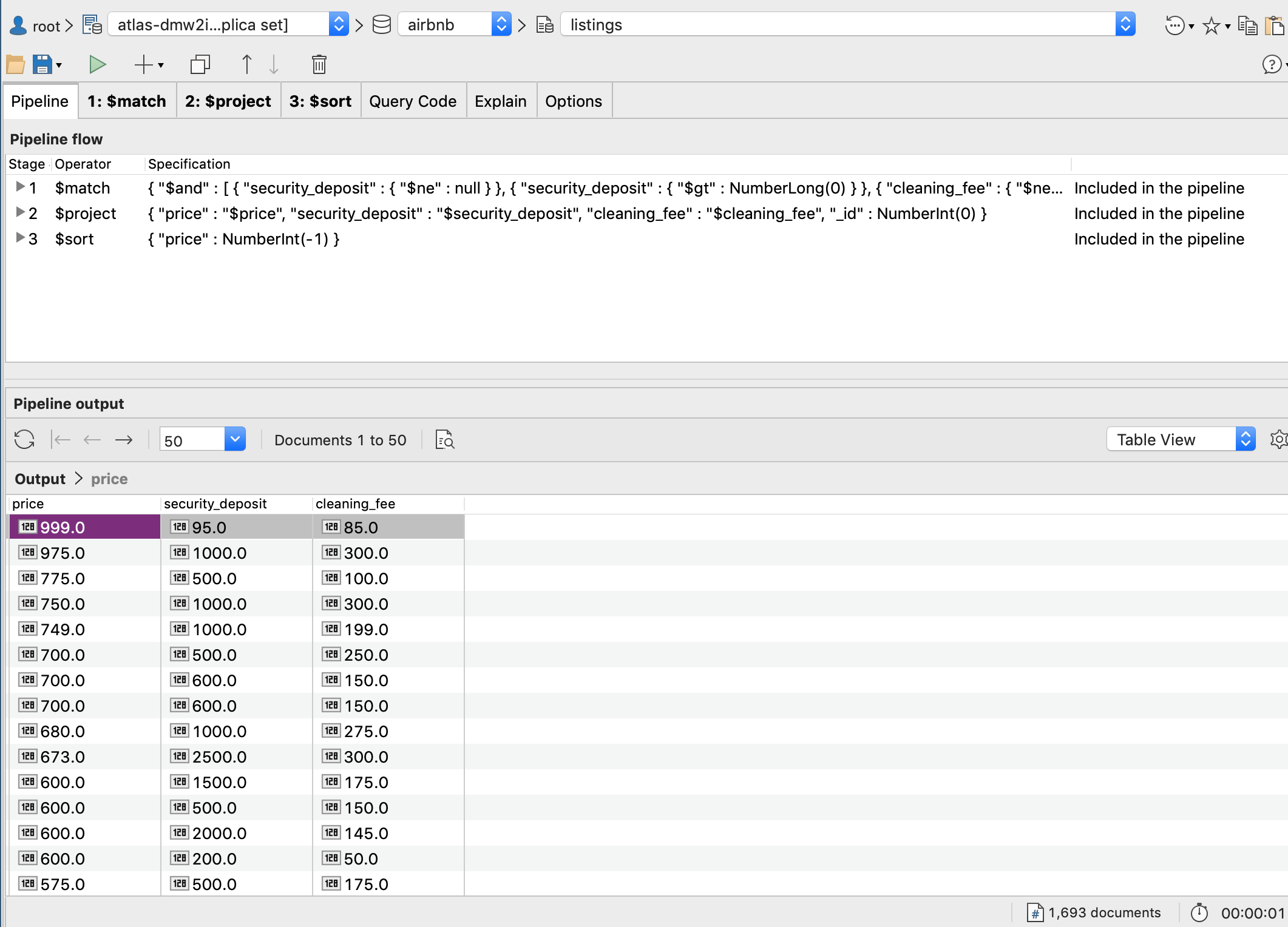
### Question-7: Properties with higher daily prices also charge higher security deposits and cleaning fees.

### Notes/Comments About MongoDB Query/Code: As seen from the results, we disprove the results. Properties with higher daily prices donot charge higher security deposits and cleaning fees.

### Results (Include # of Documents in Result): **1693**

Translation: From the listings collection, get a list of prices for each listings and the security fee and cleaning fee for the listings.

### Screen Shot of MongoDB Query/Code and Results:



use airbnb**;**

db.getCollection**(**"listings"**)**.find**(**

**{**

"$and" **:** **[**

**{**

"security\_deposit" **:** **{**

"$ne" **:** null

**}**

**},**

**{**

"security\_deposit" **:** **{**

"$gt" **:** NumberLong**(**0**)**

**}**

**},**

**{**

"cleaning\_fee" **:** **{**

"$ne" **:** null

**}**

**},**

**{**

"cleaning\_fee" **:** **{**

"$gt" **:** NumberLong**(**0**)**

**}**

**}**

**]**

**},**

**{**

"price" **:** "$price"**,**

"security\_deposit" **:** "$security\_deposit"**,**

"cleaning\_fee" **:** "$cleaning\_fee"**,**

"\_id" **:** NumberInt**(**0**)**

**}**

**)**.sort**(**

**{**

"price" **:** NumberInt**(**-1**)**

**}**

**);**