DBMI 2021 - Assignment 2 - Anastasios Moraitis - p2822124 / Ioannis Dekoulakos - p2822110

ReportBnb

Analyzing Investing Opportunities on Short Term Rental Properties

How to invest in real estate via Airbnb? Is it suitable for you? A detailed Market analysis and its trends.

Airbnb is an online marketplace that connects people who want to rent out their homes with people who are looking for accommodations in that locale. It currently covers more than 100,000 cities and 220 countries worldwide.

a. Business goals, description of the problem/domain

1. Application targeting

   Airbnb accommodations are spreading around the world because hosts often provide personalized services and strategic locations at lower prices than hotels. That makes Airbnb the ideal investing opportunity for maximizing their income.

   Our reporting targets to owners who are thinking of converting their properties and listing them for short term rental via Airbnb or buying a property in reporting areas for investments reasons (airbnb).

   Our analysis targets 3 of the most touristic areas in Italy(source):

   - Florence
   - Rome
   - Venice

   Airbnb is a new trend in terms of real estate investment. An ever larger number of property owners have to cope with an investing dilemma: ***short*** or ***long*** term rental.

2. Problem definition

   We decided to focus specifically on four questions:

   1. Can we compare the location price for a listing, and which are the most common room types?
   2. How the hosts and listings' score-performance described for each area?
   3. What is the average occupancy rate for the next 7 months for each neighborhood based on the bedrooms and room type?
   4. How does the price of an Airbnb differ throughout neighborhoods in the reporting cities, and what neighborhoods/ are the best value for money?

3. Our data sources

- Inside Airbnb
  - Listings details dataset(.csv) - Example
  - Calendar details per listing dataset(.csv) - Example
  - Neighborhoods GIS data(.geojson) - Example
- Transport data for each reporting city in Google Transit API's format. Data Source: Transit Wiki
- Points of interest from itouristmaps.com

b. Description of data sources

1. Sources of data

1. Listings details dataset This dataset contains all the details for each unique listing. Includes information of the host the listing itself(location,)

   Listings Data dictionary:

Field	Type	Calculated	Description
id	integer		Airbnb's unique identifier for the listing
listing_url	text	y	
scrape_id	bigint	y	Inside Airbnb "Scrape" this was part of
last_scraped	datetime	y	UTC. The date and time this listing was "scraped".
name	text		Name of the listing
description	text		Detailed description of the listing
neighborhood_overview	text		Host's description of the neighbourhood
picture_url	text		URL to the Airbnb hosted regular sized image for the listing
host_id	integer		Airbnb's unique identifier for the host/user
host_url	text	y	The Airbnb page for the host
host_name	text		Name of the host. Usually just the first name(s).
host_since	date		The date the host/user was created. For hosts that are Airbnb guests this could be the date they registered as a guest.
host_location	text		The host's self reported location
host_about	text		Description about the host
host_response_time			
host_response_rate			
host_acceptance_rate			That rate at which a host accepts booking requests.
host_is_superhost	boolean [t=true; f=false]		
host_thumbnail_url	text		
host_picture_url	text		
host_neighbourhood	text		
host_listings_count	text		The number of listings the host has (per Airbnb calculations)
host_total_listings_count	text		The number of listings the host has (per Airbnb calculations)
host_verifications			
host_has_profile_pic	boolean [t=true; f=false]		
host_identity_verified	boolean [t=true; f=false]		
neighbourhood	text		
neighbourhood_cleansed	text	y	The neighbourhood as geocoded using the latitude and longitude against neighborhoods as defined by open or public digital shapefiles.
neighbourhood_group_cleanse	text	y	The neighbourhood group as geocoded using the latitude and longitude against neighborhoods as defined by open or public digital shapefiles.
latitude	numeric		Uses the World Geodetic System (WGS84) projection for latitude and longitude.
longitude	numeric		Uses the World Geodetic System (WGS84) projection for latitude and longitude.
property_type	text		Self selected property type. Hotels and Bed and Breakfasts are described as such by their hosts in this field
room_type	text		[Entire home/apt|Private room|Shared room|Hotel] All homes are grouped into the following three room types: Entire place Private room

			Shared room Entire place Entire places are best if you're seeking a home away from home. With an entire place, you'll have the whole space to yourself. This usually includes a bedroom, a bathroom, a kitchen, and a separate, dedicated entrance. Hosts should note in the description if they'll be on the property or not (ex: "Host occupies first floor of the home"), and provide further details on the listing. Private rooms Private rooms are great for when you prefer a little privacy, and still value a local connection. When you book a private room, you'll have your own private room for sleeping and may share some spaces with others. You might need to walk through indoor spaces that another host or guest may occupy to get to your room. Shared rooms Shared rooms are for when you don't mind sharing a space with others. When you book a shared room, you'll be sleeping in a space that is shared with others and share the entire space with other people.
accommodates	integer		The maximum capacity of the listing
bathrooms	numeric		The number of bathrooms in the listing
bathrooms_text	string		The number of bathrooms in the listing. On the Airbnb web-site, the bathrooms field has evolved from a number to a textual description. For older scrapes, bathrooms is used.
bedrooms	integer		The number of bedrooms
beds	integer		The number of bed(s)
amenities	json		
price	currency		daily price in local currency
minimum_nights	integer		minimum number of night stay for the listing (calendar rules may be different)
maximum_nights	integer		maximum number of night stay for the listing (calendar rules may be different)
minimum_minimum_nights	integer	y	the smallest minimum_night value from the calender (looking 365 nights in the future)
maximum_minimum_nights	integer	y	the largest minimum_night value from the calender (looking 365 nights in the future)
minimum_maximum_nights	integer	y	the smallest maximum_night value from the calender (looking 365 nights in the future)
maximum_maximum_nights	integer	y	the largest maximum_night value from the calender (looking 365 nights in the future)
minimum_nights_avg_ntm	numeric	y	the average minimum_night value from the calender (looking 365 nights in the future)
maximum_nights_avg_ntm	numeric	y	the average maximum_night value from the calender (looking 365 nights in the future)
calendar_updated	date		
has_availability	boolean		[t=true; f=false]
availability_30	integer	y	avaliability_x. The availability of the listing x days in the future as determined by the calendar. Note a listing may not be available because it has been booked by a guest or blocked by the host.
availability_60	integer	y	avaliability_x. The availability of the listing x days in the future as determined by the calendar. Note a listing may not be available because it has been booked by a guest or blocked by the host.
availability_90	integer	y	avaliability_x. The availability of the listing x days in the future as determined by the calendar. Note a listing may not be available because it has been booked by a guest or blocked by the host.
availability_365	integer	y	avaliability_x. The availability of the listing x days in the future as determined by the calendar. Note a listing may not be available because it has been booked by a guest or blocked by the host.
calendar_last_scraped	date		
number_of_reviews	integer		The number of reviews the listing has
number_of_reviews_ltm	integer	y	The number of reviews the listing has (in the last 12 months)
number_of_reviews_l30d	integer	y	The number of reviews the listing has (in the last 30 days)
first_review	date	y	The date of the first/oldest review
last_review	date	y	The date of the last/newest review
review_scores_rating			
review_scores_accuracy			
review_scores_cleanliness			
review_scores_checkin			
review_scores_communication			
review_scores_location			
review_scores_value			
license	text		The licence/permit/registration number
instant_bookable	boolean		[t=true; f=false]. Whether the guest can automatically book the listing without the host requiring to accept their booking request. An indicator of a commercial listing

calculated_host_listings_count	integer	y	The number of listings the host has in the current scrape, in the city/region geography.
calculated_host_listings_count_	integer	y	The number of Entire home/apt listings the host has in the current scrape, in the city/region geography
calculated_host_listings_count_	integer	y	The number of Private room listings the host has in the current scrape, in the city/region geography
calculated_host_listings_count_	integer	y	The number of Shared room listings the host has in the current scrape, in the city/region geography
reviews_per_month	numeric	y	The number of reviews the listing has over the lifetime of the listing

2. Calendar details per listing dataset

   Calendar Data dictionary

Field	Type	Description
listing_id		
date	date	
available	text	
price	text	default price
adjusted_price	text	An adjustment is an amount of money a host owes as a result of a cancellation, reservation change
minimum_nights	number	
maximum_nights	number	

3. Neighborhoods GIS data

```
gep.head(3)
```

✓ 0.1s

	Unnamed: 0	neighbourhood	neighbourhood_group	geometry
0	0	Rifredi	NaN	MULTIPOLYGON (((11.15421 43.786381, 11.153517 ...
1	1	Gavinana Galluzzo	NaN	MULTIPOLYGON (((11.231188 43.759284, 11.23121 ...
2	2	Campo di Marte	NaN	MULTIPOLYGON (((11.270991 43.764896, 11.271159...

4. Transport data for each reporting city from transitwiki.

```
stops.head(5)
```

[34] ✓ 0.1s

	stop_id	stop_name	stop_lat	stop_lon	stop_code
0	FM0002_5	San Zanobi	43.779172	11.255805	FM0002
1	FM0003_5	Fusinato	43.797920	11.274311	FM0003
2	FM0004_5	Il Cionfo	43.796374	11.273260	FM0004
3	FM0005_5	San Marco Vecchio	43.793724	11.272051	FM0005
4	FM0006_5	Palancola	43.792008	11.270311	FM0006

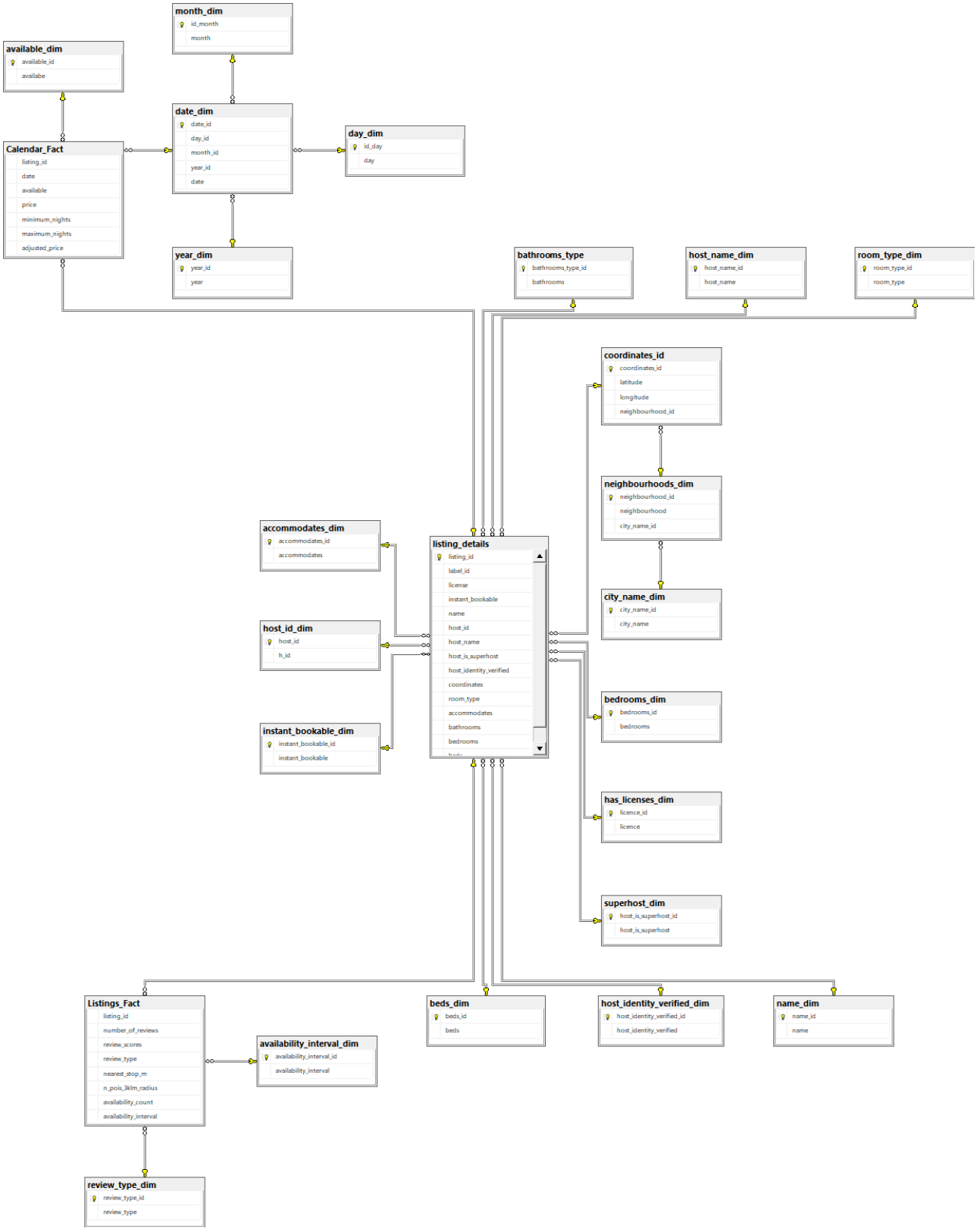
5. Points of interest from itouristmaps.com

```
geo.head(5)
```

9] ✓ 0.8s

	Unnamed: 0	name	cmt	desc	geometry
0	0	Archaeological Museum	National Archaeological Museum	National Archaeological Museum	POINT Z (11.262413 43.77616 0)
1	1	Basilica di San Lorenzo	Basilica di San Lorenzo	Basilica di San Lorenzo	POINT Z (11.2539 43.7749 0)
2	2	Basilica of Santa Croce	Basilica of Santa Croce	Basilica of Santa Croce	POINT Z (11.2626 43.7685 0)
3	3	Boboli Gardens	Boboli Gardens	Boboli Gardens	POINT Z (11.248198 43.76254 0)
4	4	Dante Alighieri Statue	Dante Alighieri Statue	Dante Alighieri Statue	POINT Z (11.262258 43.768829 0)

2. Description of the relational design of your fact and dimension tables



3. Import methods/Cleaning/Transformation

The steps we followed to import our data to the data warehouse were various, because of the multiple data sources and the data cleaning/transformation we wanted to apply to those.

At first we wanted to cleanup unwanted columns and replace null and unknown values at our convenience.

We created a script to:

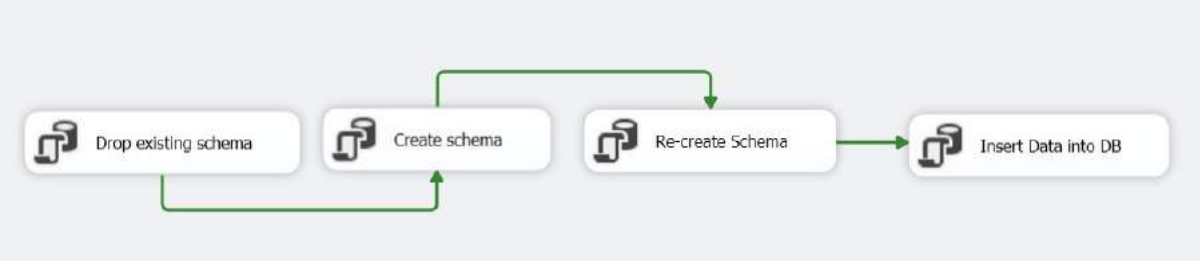
1. Scrape our data from Airbnb
    1. With the help of a scraping library we downloaded the files with HTTP and stored them locally.
    2. Extracted the zipped files into a folder.
    3. And imported the raw CSV data as data frames to perform calculations
2. Reset the state of our staging/application database
    1. Drop existing staging schema
    2. Recreate tables and views
3. Perform any necessary cleaning on the columns
    1. We used the SQL table columns and took the intersection of those with the existing ones.
    2. Replaced `na` with 0 on numeric columns that would be used as dimensions
    3. Replaced `na` with the static 'UNKNOWN` value on text columns and 'u' the classification columns that would be used as dimensions which otherwise have t(true) or f(false).
4. Replace and transform values not needed as is
    1. We replaced the values on columns that had meaningless for our analysis text.
        1. We reduced the `license` column, because we didn't need the actual text value representing the license id of the listing. We used `t`(true) where a value was present and we assigned `f`(values) to missing values.
        2. We fixed the `price` text column on the calendar data source and we made it a numeric value. Initial format $##.## => Resulting format ##. Decimal values were not important and in the majority of the rows(if not all) that decimal part was equal to 0.
5. Combine geo datasets to extract new metrics
    1. We combined the transit data by loading these data into our script and calculating for each listing the nearest transit stop from that.
    2. We, also, loaded the dataset with the Points of Interest for each city and we calculated how many of those were on a three kilometer range from the listing.(We also dropped rows were those were N/A)

    Performed graph search on geo data to find nearest distances **In both cases we made use of the Ball Tree algorithm for efficiency.** The input of the algorithm was the coordinates of the listings and the points of each secondary dataset respectively.
6. Import the raw data into the staging database(some of those in batches)
    1. We imported the listings dataframe into the db
    2. We imported the calendar csv as batches into the staging db
7. Transformation Once the data were in 2 SQL tables, we used 2 SQL Views to unpivot the data.
    1. Reviews unpivot: 7 unique types of reviews were present in the dataset and we inpivoted those columns to allow us to perform dynamic filtering on the scores, based o the review

type

2. Availability unpivot: we unpivoted the availability columns, having an interval in the set of {30, 60, 90, 365} and we kept that set as a separate reference column.

8. DW import We used the last view to import the data using an SSIS package.



9. Some importnant queries for data import

```
CREATE TABLE calendar(
listing_id INT not null,
date date NULL,
available nvarchar(1) NULL,
price int NULL,
adjusted_price int NULL,
minimum_nights int NULL,
maximum_nights int NULL,

FOREIGN KEY (listing_id)
    REFERENCES listings(id)
);
```

```
CREATE TABLE listings (
id int not null primary key,
name nvarchar(500) NULL,
host_id int NULL,
host_name nvarchar(100) NULL,
host_is_superhost nvarchar(1) NULL,
host_identity_verified nvarchar(1) NULL,
neighbourhood_cleansed nvarchar(80) NULL,
latitude Decimal(8,6) NULL,
longitude Decimal(9,6) NULL,
city_name nvarchar(20) NULL,
room_type nvarchar(50) NULL,
accommodates int NULL,
bathrooms_text nvarchar(100) NULL,
bedrooms int NULL,
beds int NULL,
number_of_reviews int NULL,
review_scores_rating decimal(3,2) NULL,
review_scores_accuracy decimal(3,2) NULL,
review_scores_cleanliness decimal(3,2) NULL,
review_scores_checkin decimal(3,2) NULL,
review_scores_communication decimal(3,2) NULL,
```

```
  review_scores_location decimal(3,2) NULL,
  review_scores_value decimal(3,2) NULL,
  license nvarchar(1) NULL,
  instant_bookable nvarchar(1) NULL,
  nearest_stop_m decimal(15,2) NULL,
  n_pois_3klm_radius int null,
  availability_30 int null,
  availability_60 int null,
  availability_90 int null,
  availability_365 int null);
```

```
CREATE VIEW listings_availability_unpivoted AS
SELECT [id]
    ,[name]
    ,[host_id]
    ,[host_name]
    ,[host_is_superhost]
    ,[host_identity_verified]
    ,[neighbourhood_cleansed]
    ,[latitude]
    ,[longitude]
    ,[city_name]
    ,[room_type]
    ,[accommodates]
    ,[bathrooms_text]
    ,[bedrooms]
    ,[beds]
    ,[number_of_reviews]
    ,review_scores
    ,review_type
    ,[license]
    ,[instant_bookable]
    ,[nearest_stop_m]
    ,[n_pois_3klm_radius],
    (CASE availability_interval
            WHEN 'availability_30' THEN '30'
            WHEN 'availability_60' THEN '60'
            WHEN 'availability_90' THEN '90'
            WHEN 'availability_365' THEN '365'
        END
    ) as availability_interval,
    NULLIF(availability_count, -1) AS availability_count
FROM (SELECT [id]
    ,[name]
    ,[host_id]
    ,[host_name]
    ,[host_is_superhost]
    ,[host_identity_verified]
    ,[neighbourhood_cleansed]
    ,[latitude]
    ,[longitude]
```

```
    ,[city_name]
    ,[room_type]
    ,[accommodates]
    ,[bathrooms_text]
    ,[bedrooms]
    ,[beds]
    ,[number_of_reviews]
    ,[review_scores]
    ,[review_type]
    ,[license]
    ,[instant_bookable]
    ,[nearest_stop_m]
    ,[n_pois_3klm_radius],
    coalesce(availability_30, -1) AS availability_30,
    coalesce(availability_60, -1) AS availability_60,
    coalesce(availability_90, -1) AS availability_90,
    coalesce(availability_365, -1) AS availability_365
    FROM [listings_review_scores_unpivoted]
    ) pvt
    UNPIVOT (
        availability_count FOR availability_interval IN (availability_30,
availability_60, availability_90,   availability_365)
    ) as unpivoted
        ```

          ```
          USE airbnb_dw;

INSERT INTO airbnb_dw.dbo.accommodates_dim SELECT DISTINCT accommodates from
airbnb.dbo.listings_availability_unpivoted
ORDER BY accommodates;
INSERT INTO airbnb_dw.dbo.bathrooms_type SELECT DISTINCT bathrooms_text from
airbnb.dbo.listings_availability_unpivoted;
INSERT INTO airbnb_dw.dbo.bedrooms_dim SELECT DISTINCT bedrooms from
airbnb.dbo.listings_availability_unpivoted
ORDER BY bedrooms;
INSERT INTO airbnb_dw.dbo.beds_dim SELECT DISTINCT beds  from
airbnb.dbo.listings_availability_unpivoted
ORDER BY beds;
INSERT INTO airbnb_dw.dbo.city_name_dim SELECT DISTINCT city_name  from
airbnb.dbo.listings_availability_unpivoted;
-- coordinates id
INSERT INTO airbnb_dw.dbo.neighbourhoods_dim
SELECT DISTINCT airbnb.dbo.listings_availability_unpivoted.neighbourhood_cleansed
as neighbourhood, ctdim.city_name_id as city_name_id from
airbnb.dbo.listings_availability_unpivoted
LEFT OUTER JOIN airbnb_dw.dbo.city_name_dim as ctdim ON (ctdim.city_name =
airbnb.dbo.listings_availability_unpivoted.city_name);

INSERT INTO coordinates_id
SELECT DISTINCT lst.latitude, lst.longitude, nb_dim.neighbourhood_id from
airbnb.dbo.listings_availability_unpivoted as lst
LEFT OUTER JOIN airbnb_dw.dbo.neighbourhoods_dim AS nb_dim ON
(nb_dim.neighbourhood = lst.neighbourhood_cleansed);
```

```
INSERT INTO airbnb_dw.dbo.has_licenses_dim SELECT DISTINCT license  from
airbnb.dbo.listings_availability_unpivoted;
INSERT INTO airbnb_dw.dbo.host_id_dim SELECT DISTINCT host_id as h_id  from
airbnb.dbo.listings_availability_unpivoted;
INSERT INTO airbnb_dw.dbo.name_dim SELECT DISTINCT name from
airbnb.dbo.listings_availability_unpivoted;

INSERT INTO airbnb_dw.dbo.host_identity_verified_dim SELECT DISTINCT
host_identity_verified  from airbnb.dbo.listings_availability_unpivoted;
INSERT INTO airbnb_dw.dbo.host_name_dim SELECT DISTINCT host_name from
airbnb.dbo.listings_availability_unpivoted;
INSERT INTO airbnb_dw.dbo.instant_bookable_dim SELECT DISTINCT instant_bookable
from airbnb.dbo.listings_availability_unpivoted;

INSERT INTO airbnb_dw.dbo.room_type_dim SELECT DISTINCT room_type  from
airbnb.dbo.listings_availability_unpivoted;
INSERT INTO airbnb_dw.dbo.superhost_dim SELECT DISTINCT [host_is_superhost]  from
airbnb.dbo.listings_availability_unpivoted;

INSERT INTO airbnb_dw.dbo.availability_interval_dim SELECT DISTINCT
availability_interval from airbnb.dbo.listings_availability_unpivoted
INSERT INTO airbnb_dw.dbo.review_type_dim SELECT DISTINCT review_type from
airbnb.dbo.listings_availability_unpivoted


INSERT INTO airbnb_dw.dbo.listing_details
(label_id, name, host_id, host_name, host_is_superhost, host_identity_verified,
coordinates, room_type, accommodates, bathrooms, bedrooms, beds, license,
instant_bookable)

SELECT lt.id as label_id, name_dim.name_id as name, host_id_dim.host_id as
host_id, host_name_dim.host_name_id as host_name,
superhost_dim.host_is_superhost_id as host_is_superhost,
host_identity_verified_dim.host_identity_verified_id as host_identity_verified,
coordinates_id.coordinates_id as coordinates, room_type_dim.room_type_id as
room_type, accommodates_dim.accommodates_id as accommodates,
bathrooms_type.bathrooms_type_id as bathrooms, bedrooms_dim.bedrooms_id as
bedrooms, beds_dim.beds_id as beds, has_licenses_dim.licence_id as license,
instant_bookable_dim.instant_bookable_id as instant_bookable
FROM (SELECT DISTINCT id from airbnb.dbo.listings)AS lt
INNER JOIN airbnb.dbo.listings as l on (lt.id =l.id)
INNER  JOIN accommodates_dim ON (accommodates_dim.accommodates = l.accommodates)
INNER  JOIN bathrooms_type ON (bathrooms_type.bathrooms = l.bathrooms_text)
INNER  JOIN bedrooms_dim ON (bedrooms_dim.bedrooms = l.bedrooms)
INNER  JOIN beds_dim ON (beds_dim.beds = l.beds)
INNER  JOIN city_name_dim ON (city_name_dim.city_name = l.city_name)
INNER  JOIN neighbourhoods_dim ON (neighbourhoods_dim.neighbourhood =
l.neighbourhood_cleansed)
INNER  JOIN name_dim ON (name_dim.name = l.name)
INNER  JOIN coordinates_id ON (coordinates_id.latitude = l.latitude and
coordinates_id.longitude = l.longitude)
INNER  JOIN has_licenses_dim ON (has_licenses_dim.licence = l.license)
INNER  JOIN host_id_dim ON (host_id_dim.h_id = l.host_id)
```

```
INNER  JOIN host_identity_verified_dim ON
(host_identity_verified_dim.host_identity_verified = l.host_identity_verified)
INNER  JOIN host_name_dim ON (host_name_dim.host_name = l.host_name)
INNER  JOIN instant_bookable_dim ON (instant_bookable_dim.instant_bookable =
l.instant_bookable)
INNER  JOIN room_type_dim ON (room_type_dim.room_type = l.room_type)
INNER  JOIN superhost_dim ON (superhost_dim.host_is_superhost =
l.host_is_superhost);

  INSERT INTO airbnb_dw.dbo.Listings_Fact
(listing_id, number_of_reviews, review_scores, review_type, availability_count,
availability_interval, nearest_stop_m, n_pois_3klm_radius)
SELECT
listing_details.listing_id as listing_id,number_of_reviews, review_scores,
review_type_dim.review_type_id as review_type, availability_count,
availability_interval_dim.availability_interval_id as
availability_interval,nearest_stop_m,n_pois_3klm_radius
FROM airbnb.dbo.listings_availability_unpivoted AS l
LEFT OUTER JOIN listing_details ON (listing_details.label_id = l.id)
INNER JOIN availability_interval_dim ON
(availability_interval_dim.availability_interval = l.availability_interval)
INNER JOIN review_type_dim ON (review_type_dim.review_type = l.review_type);

INSERT INTO available_dim SELECT DISTINCT available from airbnb.dbo.calendar;
INSERT INTO day_dim SELECT DISTINCT DAY(date) as day from airbnb.dbo.calendar
ORDER BY day;
INSERT INTO month_dim SELECT DISTINCT MONTH(date) as month from
airbnb.dbo.calendar
ORDER BY month;
INSERT INTO year_dim SELECT DISTINCT YEAR(date) as year from airbnb.dbo.calendar
ORDER BY year;
INSERT INTO date_dim
SELECT DISTINCT day_dim.id_day as day_id, month_dim.id_month as month_id,
year_dim.year_id as year_id, calendar.date as date from airbnb.dbo.calendar
LEFT OUTER JOIN day_dim ON (day_dim.day = DAY(calendar.date))
LEFT OUTER JOIN month_dim ON (month_dim.month = MONTH(calendar.date))
LEFT OUTER JOIN year_dim ON (year_dim.year = YEAR(calendar.date));

INSERT INTO Calendar_Fact(listing_id, available, date, price, adjusted_price,
minimum_nights, maximum_nights)
SELECT listing_details.listing_id as listing_id, available_dim.available_id as
available, date_dim.date_id as date, price, adjusted_price, minimum_nights,
maximum_nights
FROM airbnb.dbo.calendar as c
INNER JOIN listing_details ON (listing_details.label_id = c.listing_id)
INNER JOIN available_dim ON (airbnb_dw.dbo.available_dim.availabe = c.available)
INNER JOIN day_dim ON (day_dim.day = DAY(c.date))
INNER JOIN month_dim ON (month_dim.month = MONTH(c.date))
INNER JOIN year_dim ON (year_dim.year = YEAR(c.date))
INNER JOIN date_dim ON (date_dim.day_id = day_dim.id_day and date_dim.month_id =
month_dim.id_month and date_dim.year_id = year_dim.year_id)
```

1. Import/cleaning/transformation challenges and treatment Some of the challenges we had are presented below.

    1. Automatic reprocessing of the raw dataset to the staging database

        1. We used a script to do that job. Parts of it was the whole mining-cleaning-transformation of the main dataset.
        2. We also created an SSIS package to re-import the data into the Data Warehouse whenever the source were being updated.

    2. Finding the most performant algorithm to create the distance related metrics from the secondary dataset.

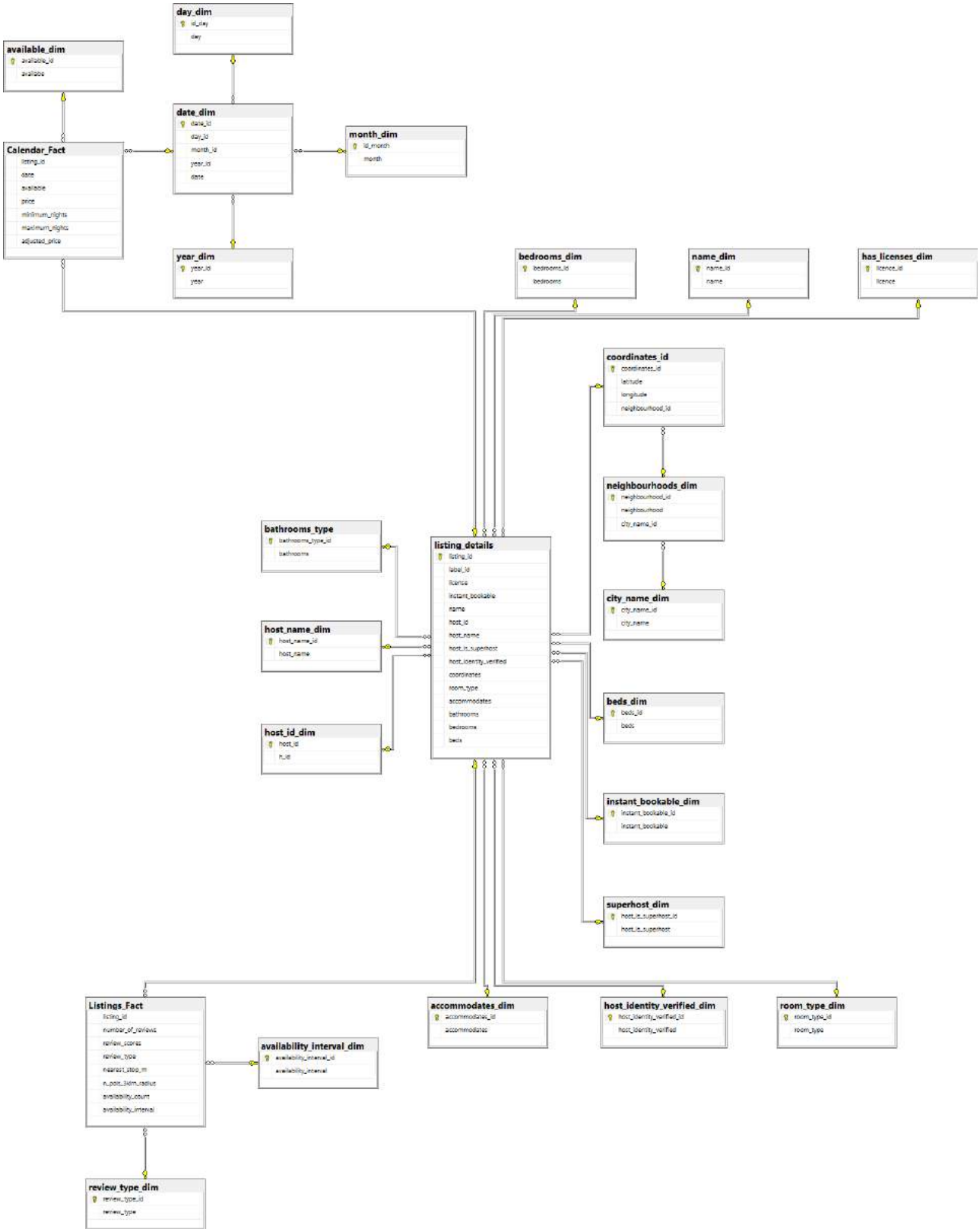
        We could use a brute-force approach because of the small dataset, approximately 45 thousands listings, or the K-dimensional tree algorithm. While researching how we could measure these distances, we found another more performant algorithm which was that of the Ball-Tree algorithm. We applied that to our datasets using the implementation of the `sklearn.neighbors` python library.

    3. Cleaning/Importing calendar data into staging server As one can imagine day-to-day data were stored in a flat file(one per city) named calendar.csv.

        When we tried to clean and transform that raw dataset, we found that this raised a Stack Overflow error. After our research we found that the library we used had support for reading a flat file into chunks. That helped and also reduced the time of processing, even in small flat files between the three. Resulting on a 20-25 minutes of processing for 16 million rows. We could also examine data in parallel but time didn't permit us to focus on that two.

c. Design of the data warehouse/cubes

1. Data warehouse



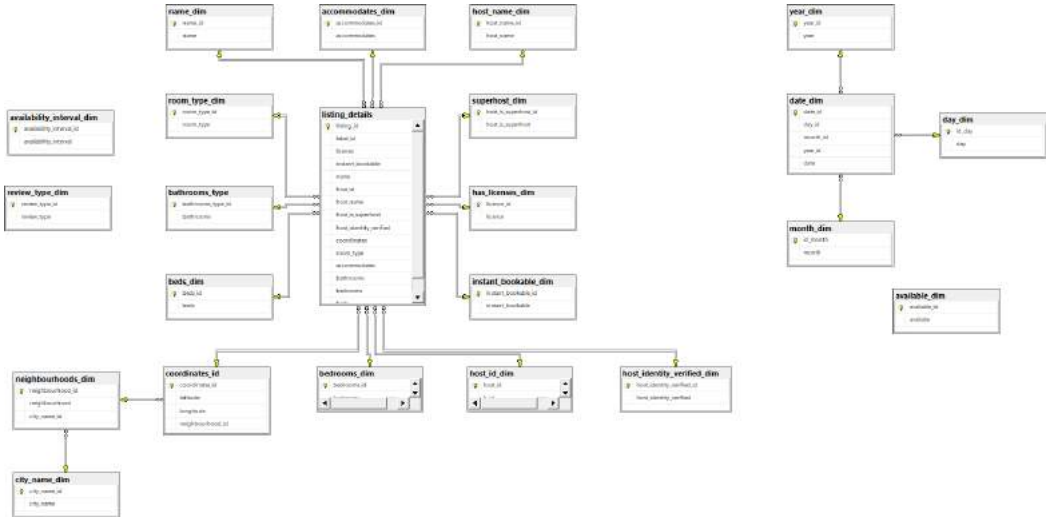
2. Cube on top of DW schema

    1. Dimensions Below we see the available dimensions created and connected with the Fact tables in our schema.

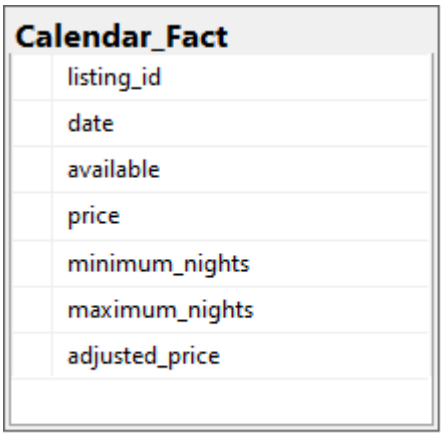
        - **Listing_Details**, availability_interval_dim and review_type_dim are dimensions connected with the Listings_Fact Fact table.

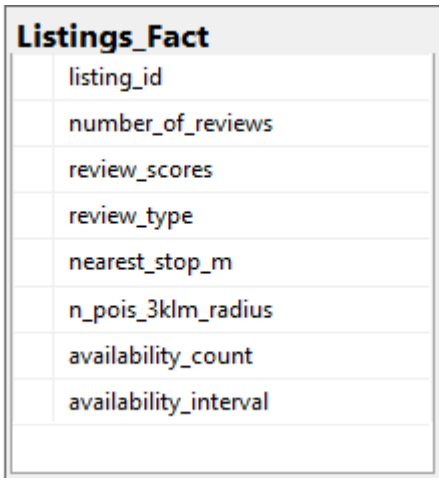
- Availability interval is the description column of the availability_count(see fact table) column created from the unpivoted listings csv
- Room type is the description column of the review_scores(see fact table) column created from the unpivoted listings csv)
- coordinates_id, neighbourhoods_dim and city_name_dim is a set of associated tables and are structured in a hierarchy.

- **Listings_Details**, date_dim and available_dim are all dimensions of the Calendar_Fact fact table.
    - date_dim, day_dim, month_dim, year_dim dimensions set is a set of associated tables and are structured in a hierarchy.



2. Measures





- Listings Fact table

- Calendar Fact table

3. Calculated measures

   Below we find the custom measures created to calculate reporting measures and their MDX code respectively:

   - [Average Occupancy - 90 days]

     Occupancy calculates as such: availability minus the selected availability interval - for our case 90(see availability_interval_dim).

     ```
     ABS([Measures].[Averge Availability] - 90)
     ```

   - [Count hosts]

     ```
     sum(existing [Listing Details].[h Id].MEMBERS,count(existing [Listing Details].[h Id].CurrentMember))-1
     ```

   - [Total Number of Cities]

     ```
     sum(existing [Listing Details].[City Name].MEMBERS,count(existing [Listing Details].[City Name].CurrentMember))-1
     ```

   - [Average Review Score - nonDistinct]

     ```
     [Measures].[Review Scores]/[Measures].[Listings Fact Count]
     ```

   - [ListingsFact_DistingCount]

     ```
     sum(existing [Listing Details].[Listing Id].MEMBERS,count(existing [Listing Details].[Listing Id].CurrentMember))-1
     ```

   - [Reservations per listing]

     Reservations are per listing, so we take the rounded value of the calendar fact count per the distinct count of listings contained. So, on the report we have to filter by the available_dim on false(false means that the for that day the listings is not available)

     ```
     Round([Measures].[Calendar Fact Count]/[Measures].[ListingsFact_DistingCount])
     ```

   - [Reservations per Year]

     On the assumption that a year has 365 days

     ```
     [Measures].[Calendar Fact Count]/365
     ```

   - [Averge Availability]

```
[Measures].[Availability Count]/[Measures].[Listings Fact Count]
```

- [Average Adgusted Price]

```
[Measures].[Adjusted Price]/[Measures].[Calendar Fact Count]
```

- [Average Price]

```
[Measures].[Price]/[Measures].[Calendar Fact Count]
```

- [Average Points of Interest]

```
[Measures].[n Pois 3klm Radius]/[Measures].[Listings Fact Count]
```

- [Average Bedrooms Distinct]

```
[Listing Details].[Bedrooms].[Bedrooms]/[Measures].
[ListingsFact_DistingCount]
```

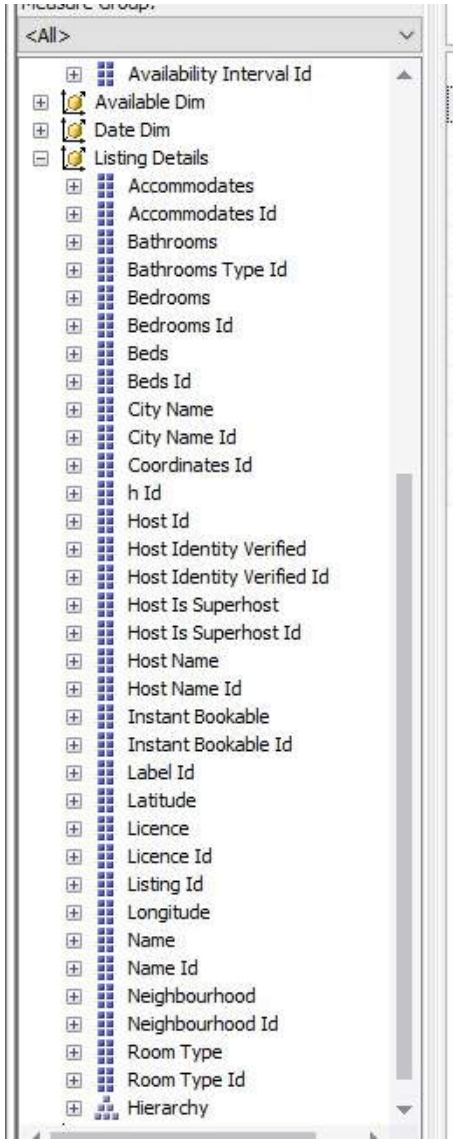
- [Count available]

```
sum(existing [Available Dim].[Availabe].MEMBERS,count(existing
[Available Dim].[Availabe].CurrentMember))-1
```

e. Examples

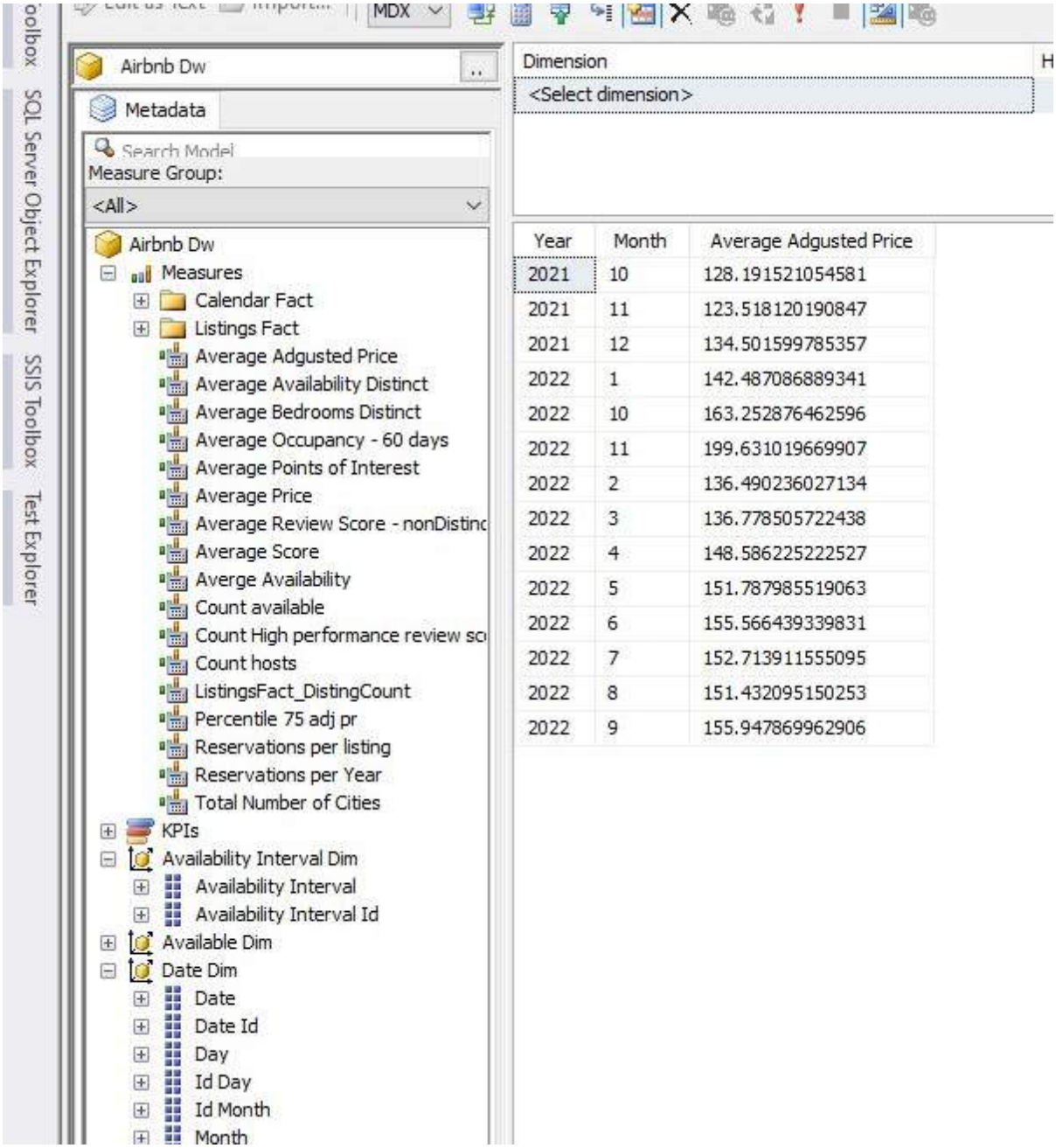
1. OLAP Reports

- Average availability in the next X days per city

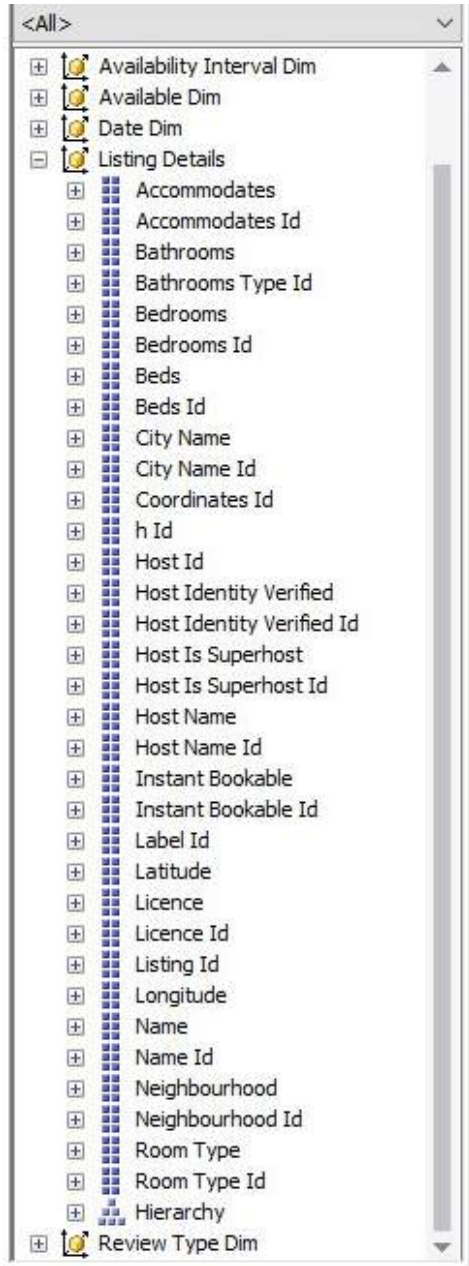


City Name	Availability Interval	Averge Availability
Florence	30	9.4069017823284
Florence	365	181.555365946151
Florence	60	23.8486916951081
Florence	90	39.3378839590444
Rome	30	9.37910471183648
Rome	365	208.178626073993
Rome	60	25.6948487582879
Rome	90	43.3321825101024
Venice	30	14.2248303934871
Venice	365	217.01289009498
Venice	60	30.8647218453189
Venice	90	50.8408412483039

- Average price drilled down by year and month

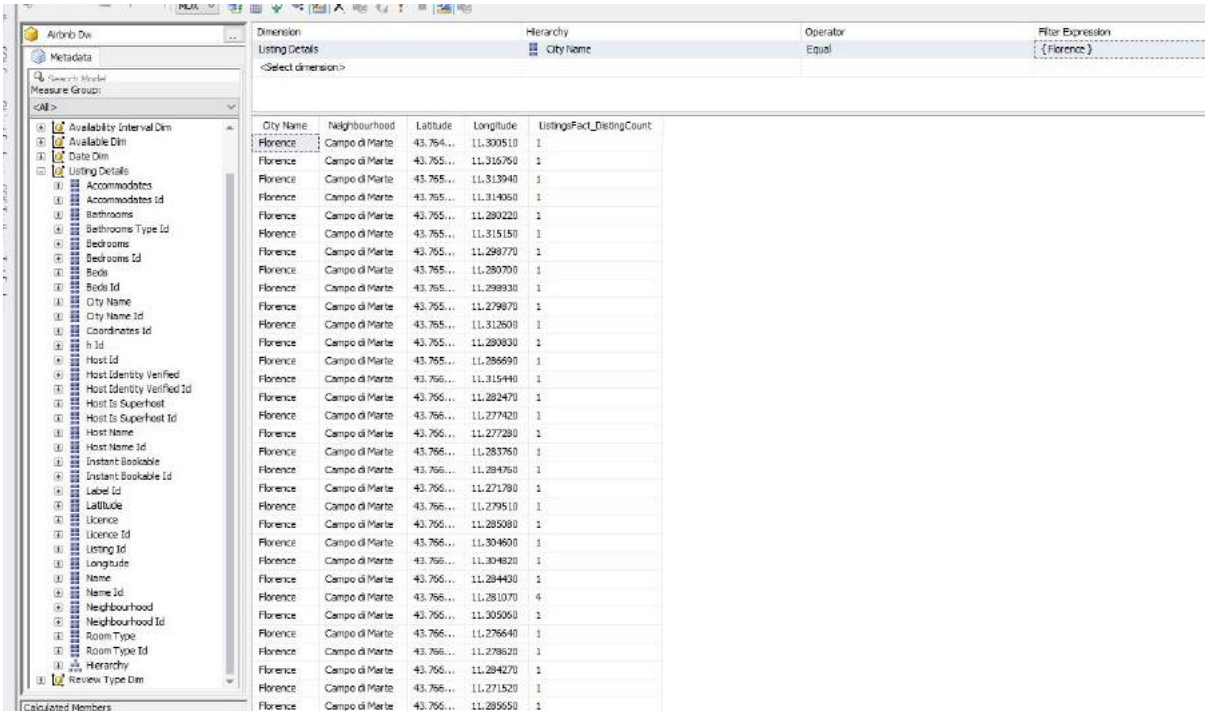


- Total Listings drilled down by city and neighbourhood



City Name	Neighbourhood	ListingsFact_DistingCount
Florence	Campo di Marte	1049
Florence	Centro Storico	7734
Florence	Gavinana Gall...	424
Florence	Isolotto Legnaia	481
Florence	Rifredi	860
Rome	I Centro Storico	13800
Rome	II Parioli/Nome...	1842
Rome	III Monte Sacro	392
Rome	IV Tiburtina	474
Rome	IX Eur	348
Rome	V Prenestino/...	985
Rome	VI Roma delle ...	218
Rome	VII San Giova...	1885
Rome	VIII Appia Antica	722
Rome	X Ostia/Acilia	655
Rome	XI Arvalia/Port...	452
Rome	XII Monte Verde	1220
Rome	XIII Aurelia	1520
Rome	XIV Monte Mario	518
Rome	XV Cassia/Fla...	458
Venice	Aeroporto	8
Venice	Alberoni	35
Venice	Altobello	31
Venice	Bissuola	46
Venice	Burano	20
Venice	Ca' Brentelle	2
Venice	Ca' Emiliani	8
Venice	Ca' Sabbioni	3
Venice	Campalto	17
Venice	Campalto Bag...	6
Venice	Campalto CEP	4
Venice	Campalto Gobbi	2
Venice	Cannaregio	1431
Venice	Carpenedo	78
Venice	Case Dosa	2
Venice	Castello	1409
Venice	Chirignago	30

- Location hierarchy

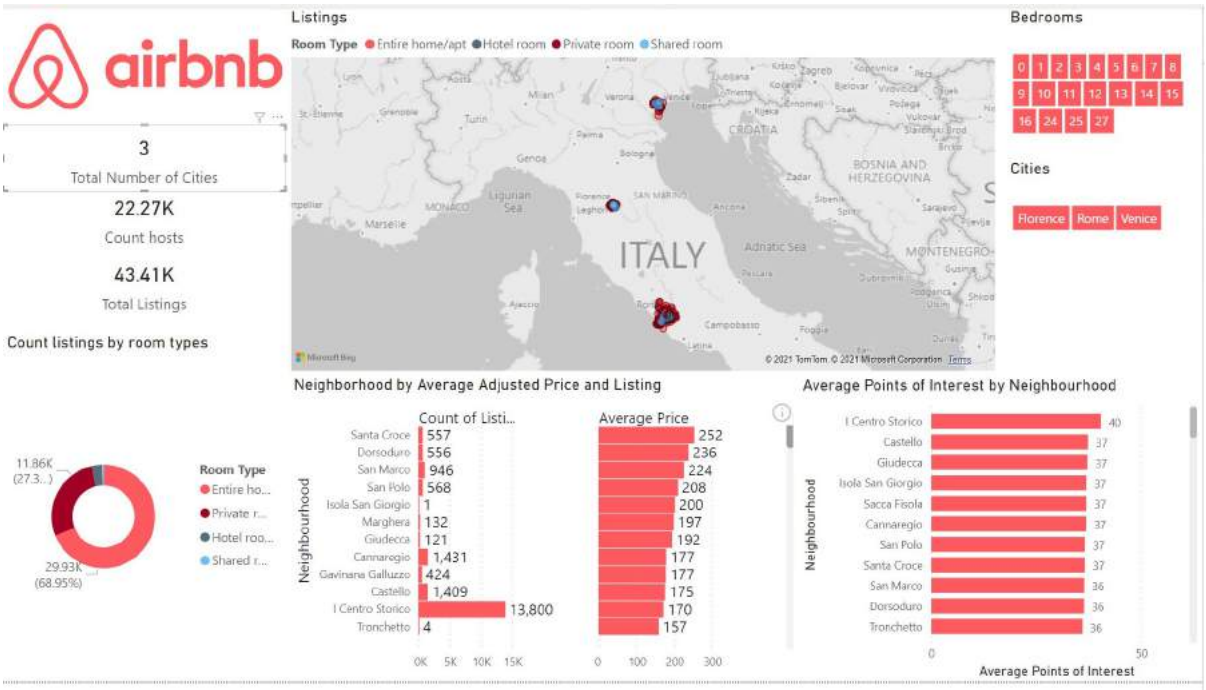


2. Visualization

For this task, we have used both native and third-party controls like Mapbox. Mapbox is used to draw a chloropath for the regions on the map.

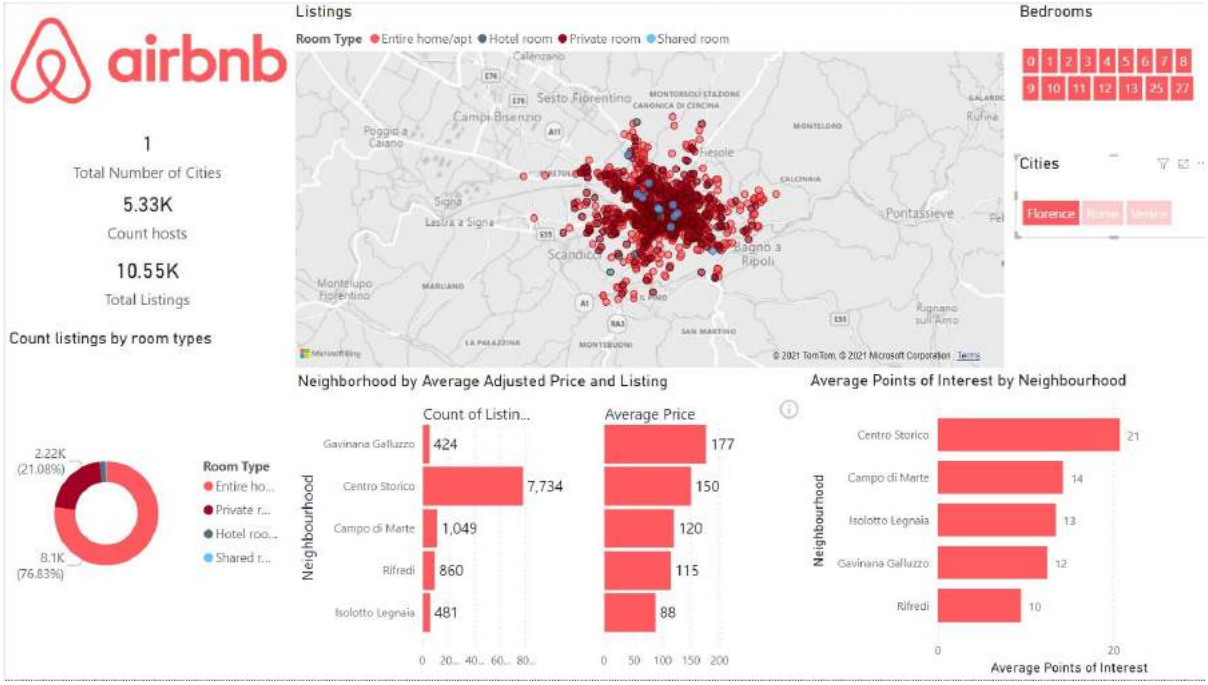
a. **Listings in Map**

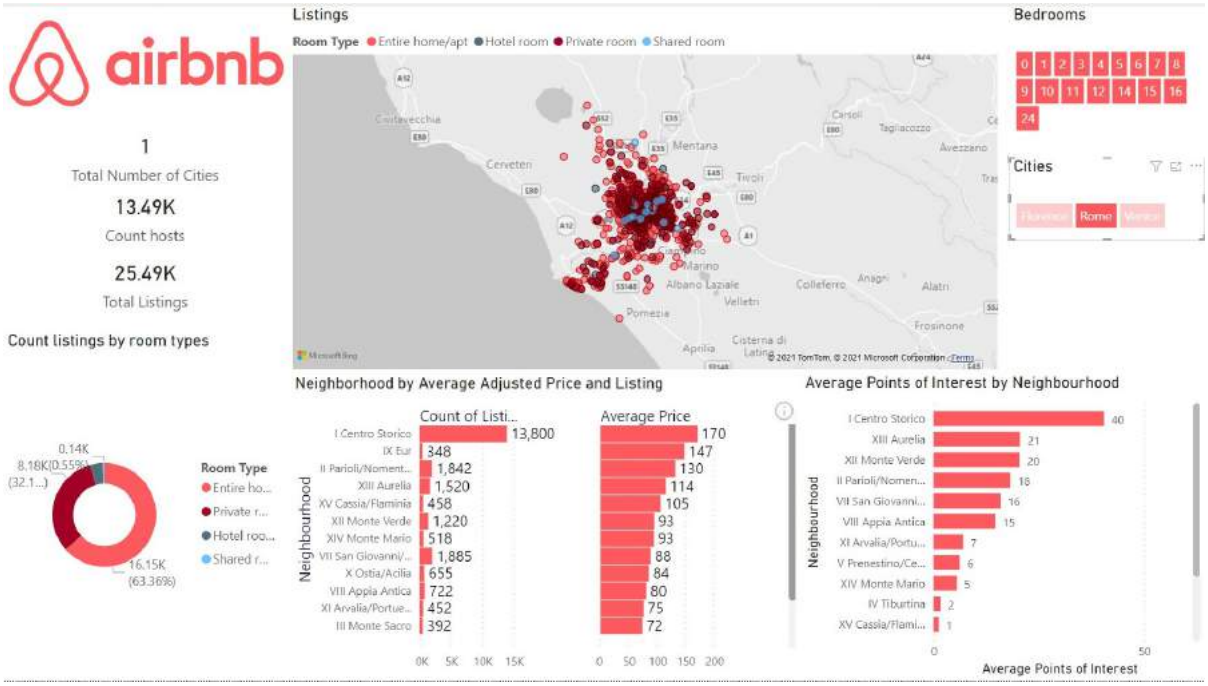
- **Total Listings Report**

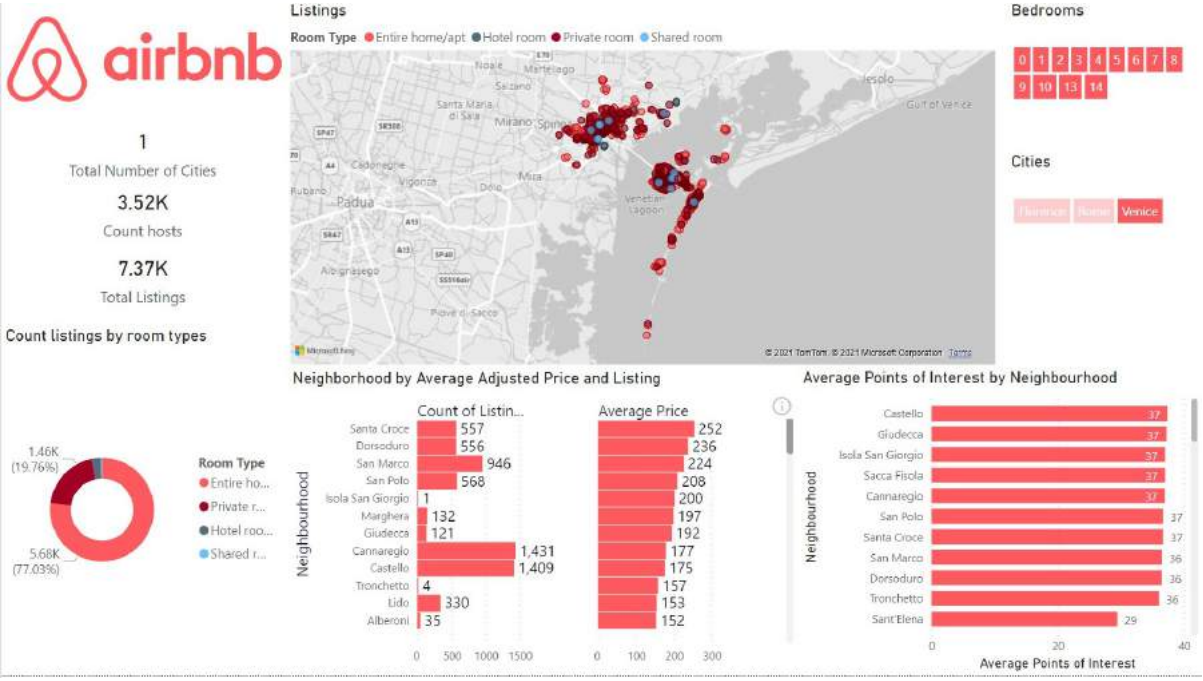


1. We analyze three cities with 43.41K listings and 22.27K hosts in total.
2. Observing the number of listings shared between four different room types we notice that Entire homes/houses have the biggest share with the Private room following.

- **Total Listing Report foreach city**


airbnb
1
Total Number of Cities
5.33K
Count hosts
10.55K
Total Listings
Count listings by room types
2.22K (21.08%)
8.1K (76.83%)
Room Type
Entire ho...
Private r...
Hotel roo...
Shared r...
Listings
Room Type Entire home/apt Hotel room Private room Shared room
Bedrooms
0 1 2 3 4 5 6 7 8 9 10 11 12 13 25 27
Cities
Florence Rome Venice
Neighborhood by Average Adjusted Price and Listing
Count of Listin...
Average Price
Gavinana Galluzzo 424 177
Centro Storico 7,734 150
Campo di Marte 1,049 120
Rifredi 860 115
Isolotto Legnaia 481 88
Neighbourhood
Average Points of Interest by Neighbourhood
Centro Storico 21
Campo di Marte 14
Isolotto Legnaia 13
Gavinana Galluzzo 12
Rifredi 10
Average Points of Interest

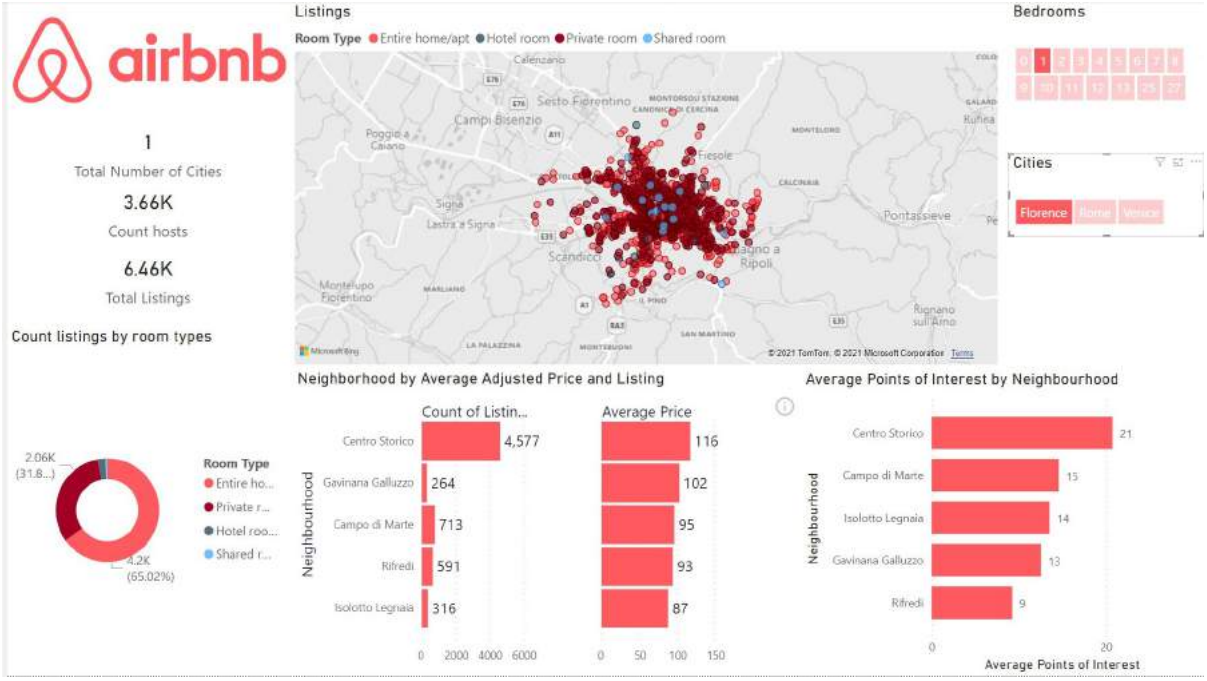

airbnb
1
Total Number of Cities
13.49K
Count hosts
25.49K
Total Listings
Count listings by room types
0.14K
8.18K (0.55%)
(32.1...)
16.15K (63.36%)
Room Type
Entire ho...
Private r...
Hotel roo...
Shared r...
Listings
Room Type Entire home/apt Hotel room Private room Shared room
Bedrooms
0 1 2 3 4 5 6 7 8 9 10 11 12 14 15 16 24
Cities
Florence Rome Venice
Neighborhood by Average Adjusted Price and Listing
Count of Listi...
Average Price
I Centro Storico 13,800 170
IX Eur 348 147
II Parioli/Noment... 1,842 130
XIII Aurelia 1,520 114
XV Cassia/Flaminia 458 105
XII Monte Verde 1,220 93
XIV Monte Mario 518 93
VII San Giovanni/... 1,885 88
X Ostia/Acilia 655 84
VIII Appia Antica 722 80
XI Arvalia/Portue... 452 75
III Monte Sacro 392 72
Neighbourhood
Average Points of Interest by Neighbourhood
I Centro Storico 40
XIII Aurelia 21
XII Monte Verde 20
II Parioli/Nomen... 18
VII San Giovanni... 16
VIII Appia Antica 15
XI Arvalia/Portu... 7
V Prenestino/Ce... 6
XIV Monte Mario 5
IV Tiburtina 2
XV Cassia/Flami... 1
Average Points of Interest

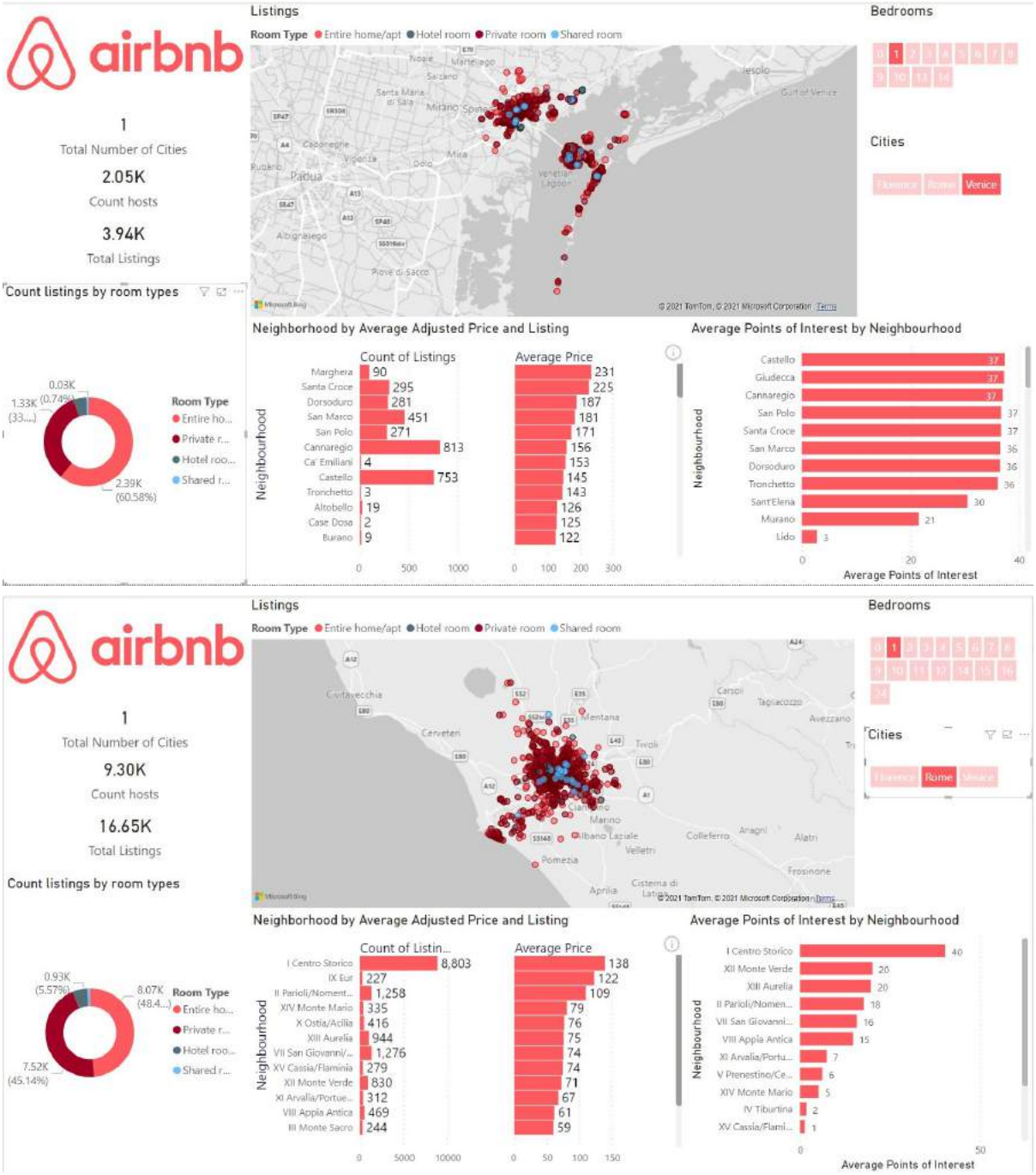


1. The maximum average price noticed in Venice
2. In all the 3 cities. we observe that the maximum share of Room Types is 'Entire Home/House' and the minimum is 'Shared Room'
3. As we can see,
    - In Rome it seems that the most listings are located in the Historical Centre with those having the maximum average price compared with other neighbourhoods,
    - In Florence it seems that the most listings are located in the Historical Centre with those having the maximum average price compared with other neighbourhoods being in 'Gavinana Galluzzo', and
    - On the contrary, for Venice most listings are located in Cannaregio and Castello with average price $176.
4. As we can see,
    - In Rome it seems that the most average Points of Interest are located in the Historical Centre,
    - In Florence it seems that the most average Points of Interest are located in the Historical Centre, and
    - On the contrary, for Venice most neighbourhoods are in the touristic part of the city and, as expected, the average Points of Interest is the maximum compared to the others.

And some filter combinations can be seen below.

- **Total 1-bedroom listings report foreach city**

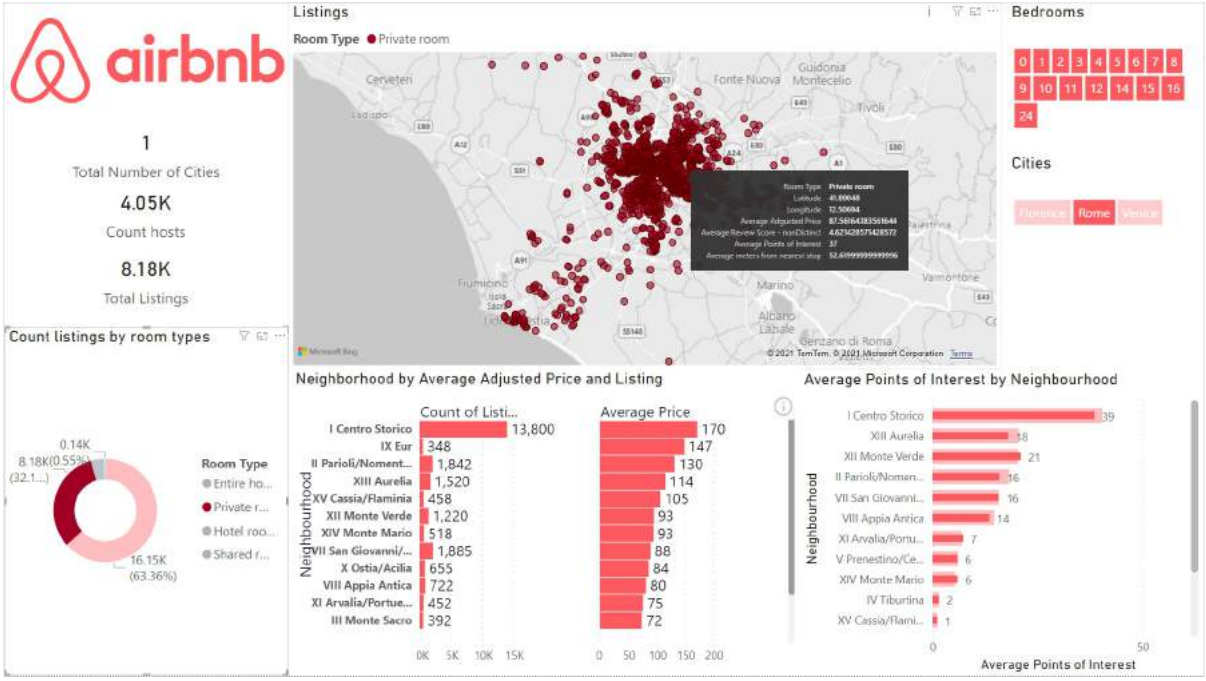




Drilling down by city and selecting Florence and filtering by listings with 1 bedroom:

1. There are approximately 3.65K, 2K and 9.3K hosts managing 1-bedroom listings in Florence, Venice and Rome respectively and the total listings with this setup are 6.46K, ~4K and 16.65K as well.
2. As we noticed in the previous report, we observe that the maximum average price for 1-bedroom listings is in Venice and the minimum is in Florence.
3. In Rome and Florence, it seems that the most 1-bedroom listings are located in the Historical Centre with those having the maximum average price compared with other neighbourhoods of each city. On the contrary, for Venice most 1-bedroom listings are located in Cannaregio with average price 156(6th smaller) and Marghera has the maximum average price in the city with only 90 listings.

- **Total private rooms report for Rome**

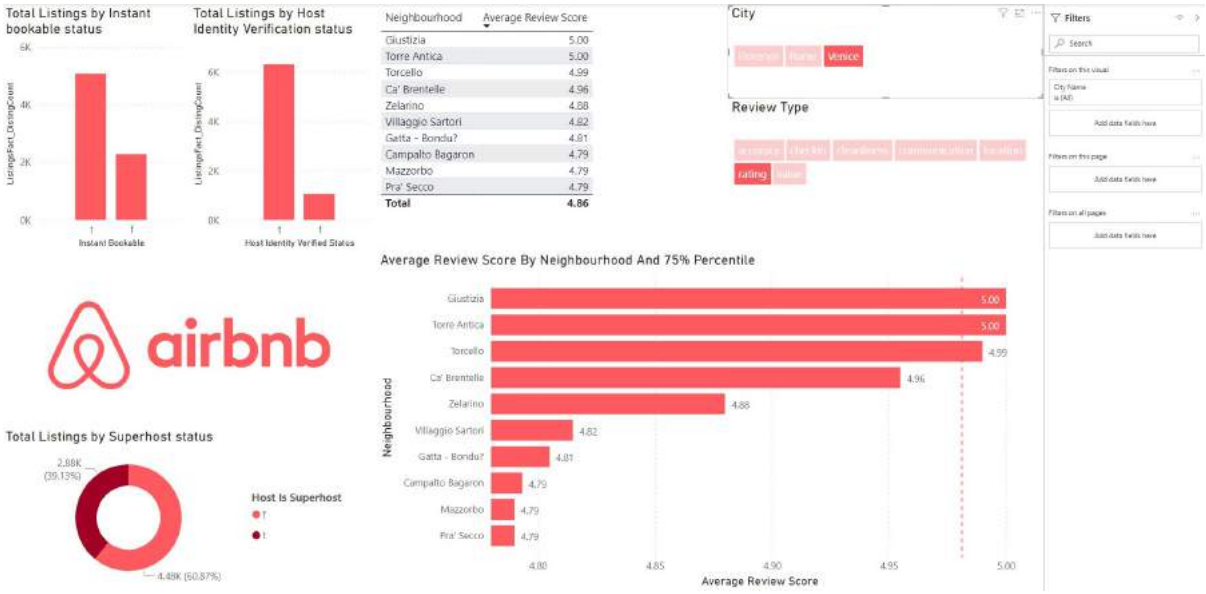


Drilling down by city and selecting Rome and filtering by private room:

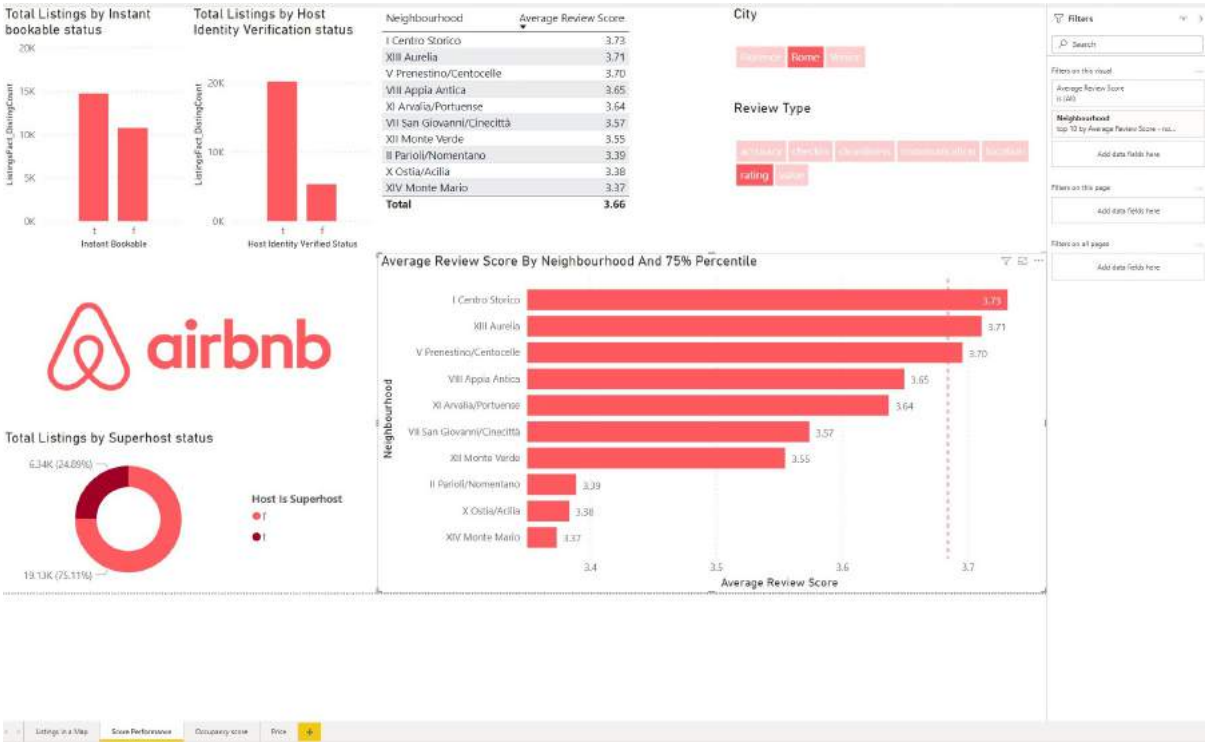
1. There are approximately 4K hosts managing private rooms in Rome and the total listings with this setup are 8.2K as well.
2. We observe that the most private rooms are part of the Historic Centre and the average price is $170.
3. If we select a specific listing on the map, we can see some extra details(This is applied at all filters):
    1. The average adjusted price,
    2. average review score,
    3. average points of interest, and
    4. average distance(in meters) from the nearest stop.

b. **Performance**

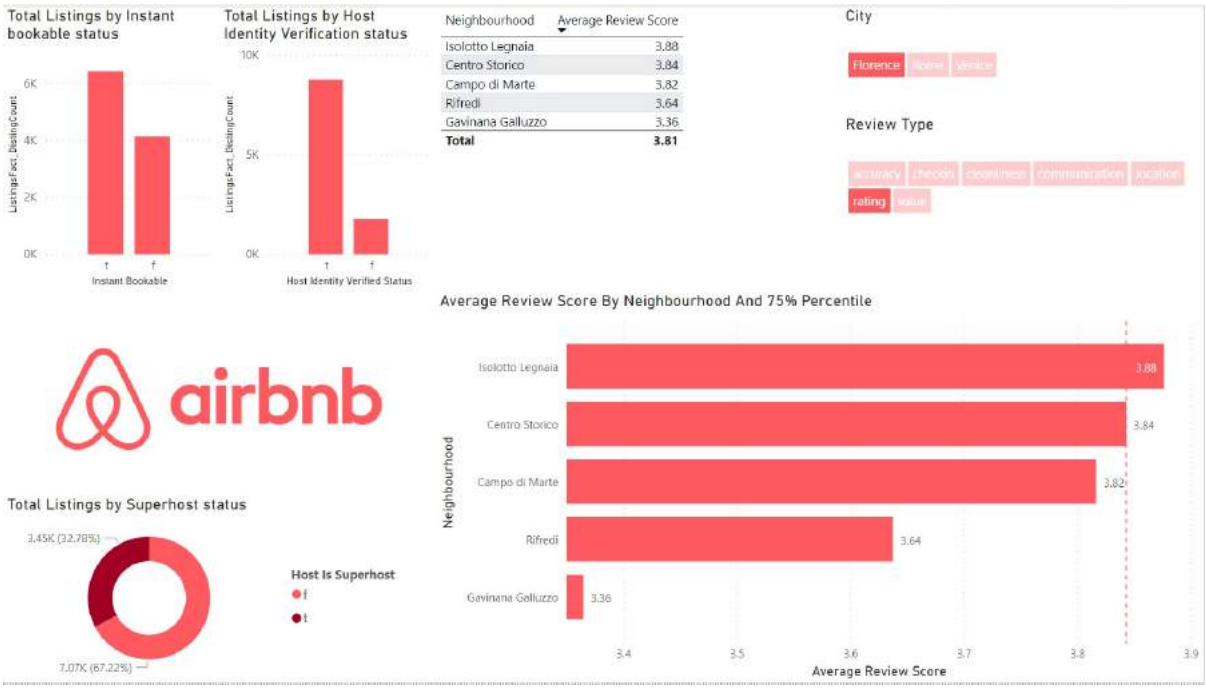
- Score on Rating foreach city



- The maximum rate score is observed for neighbourhoods: Giustizia and Torre Antica.



- The maximum rate score is observed for neighbourhoods: I Centro Storico and XIII Aurelia.

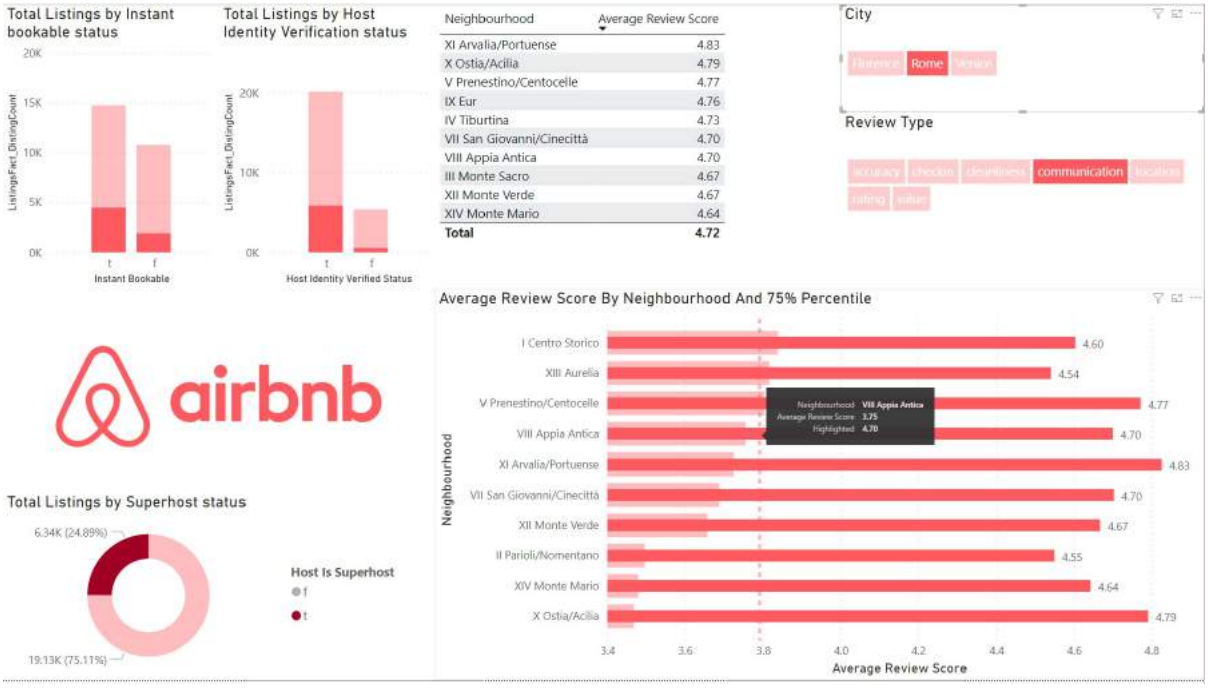


- The maximum rate score is observed for neighbourhoods: Isolotto Legnaia and Centro Storico.

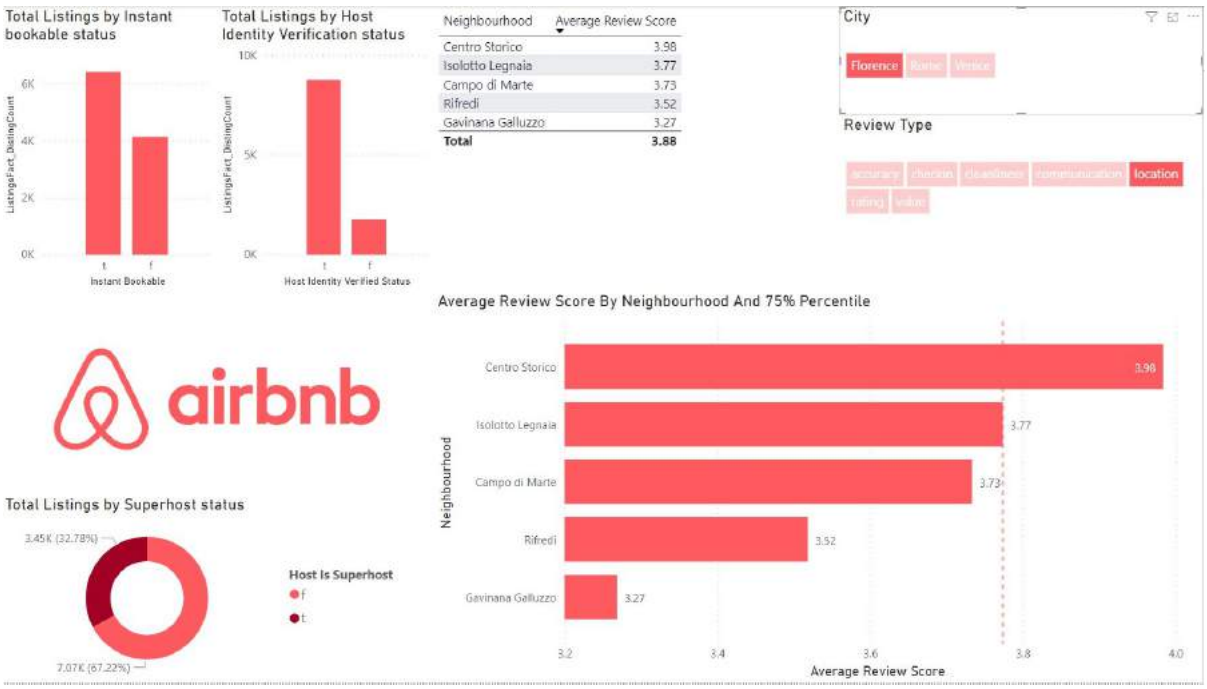
General observations

1. We observe that the biggest portion of hosts are not superhosts, but they are verified.
2. The most hosts offer the "Instant Book" option.

- **Drill-down by city and a specific review type**



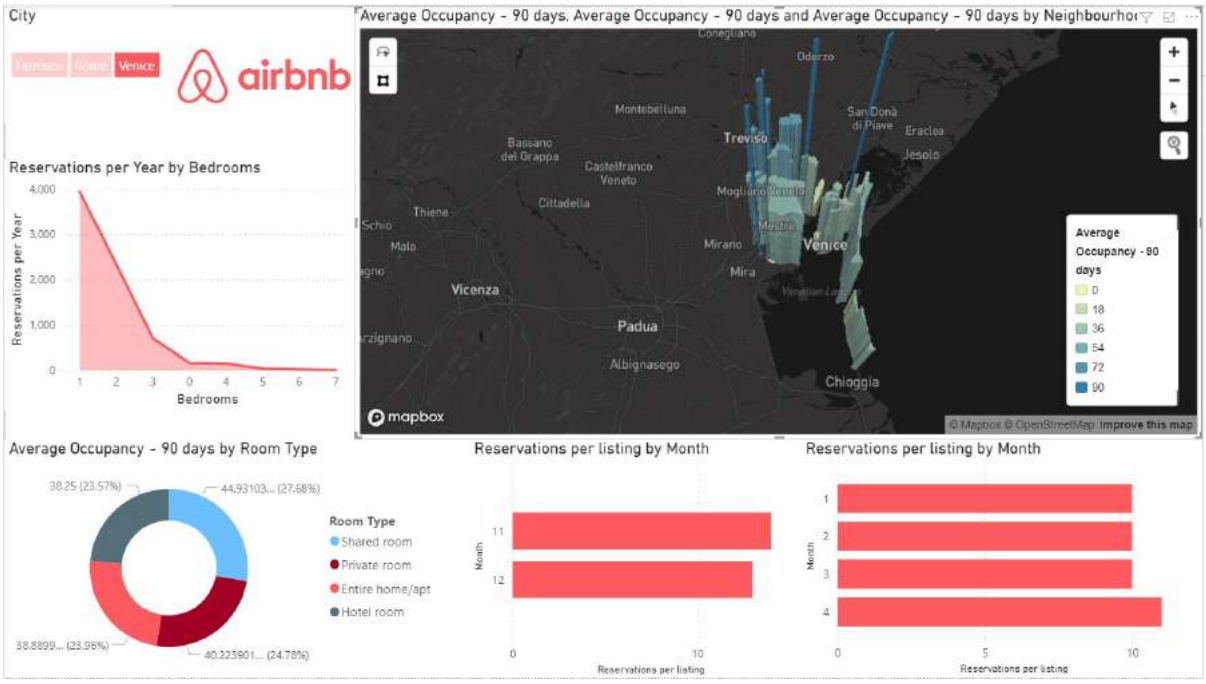
- Superhosts have hight average communication score compared to the total set of hosts.
- The majority of them have their identity identified.

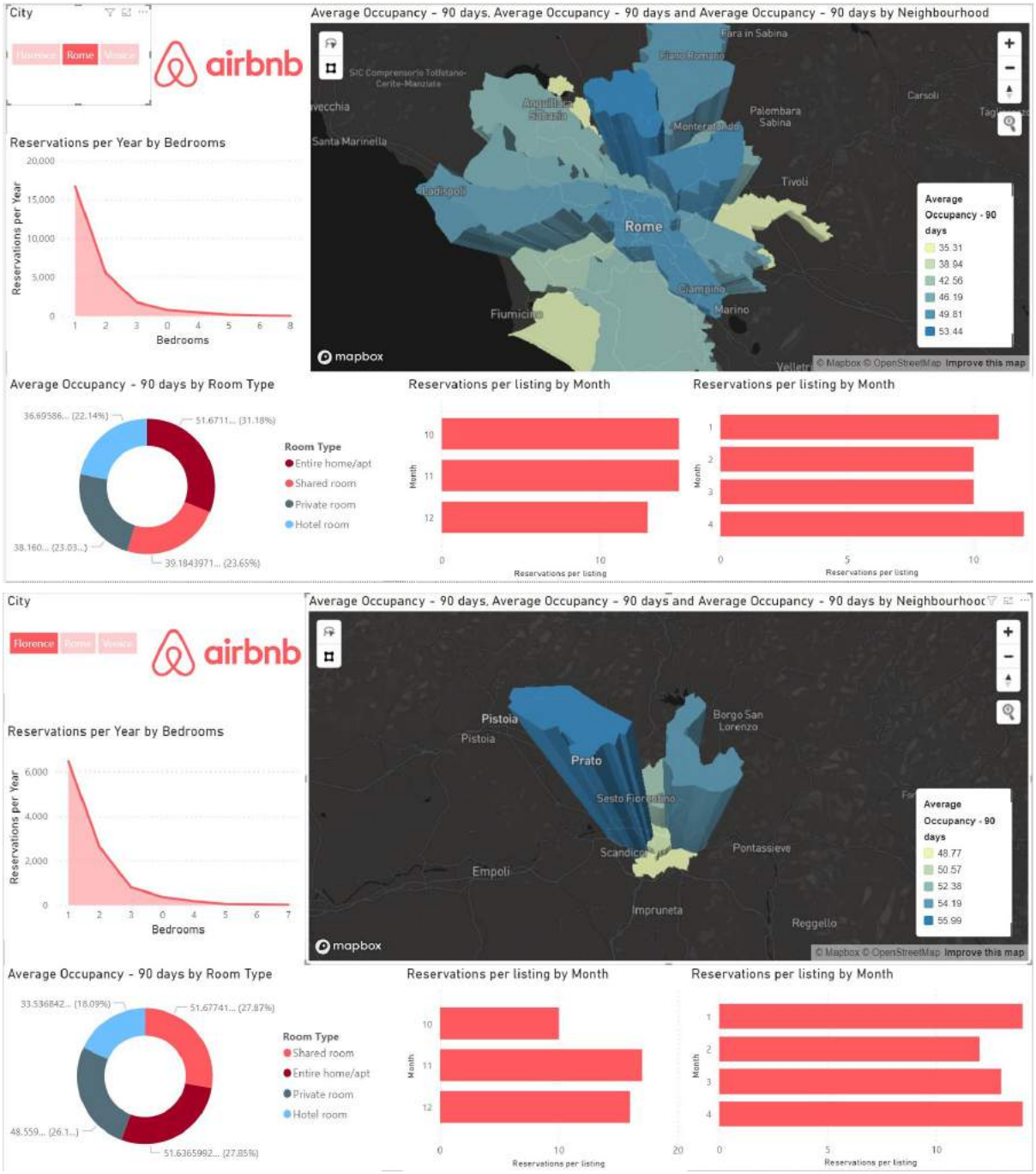


- We notice that the best average location score is observed in the Historical Centre with Isolotto Legnaia following.

c. **Occupancy/Score**

- Occupancy score by city

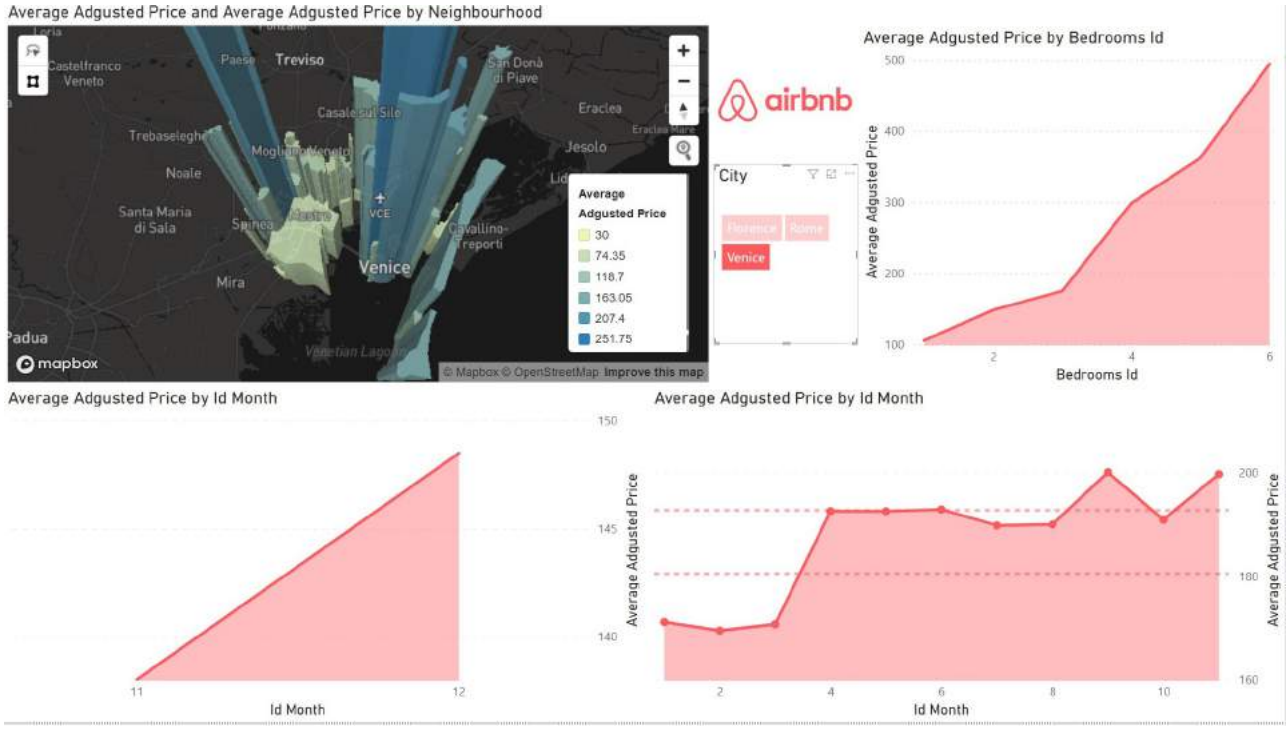


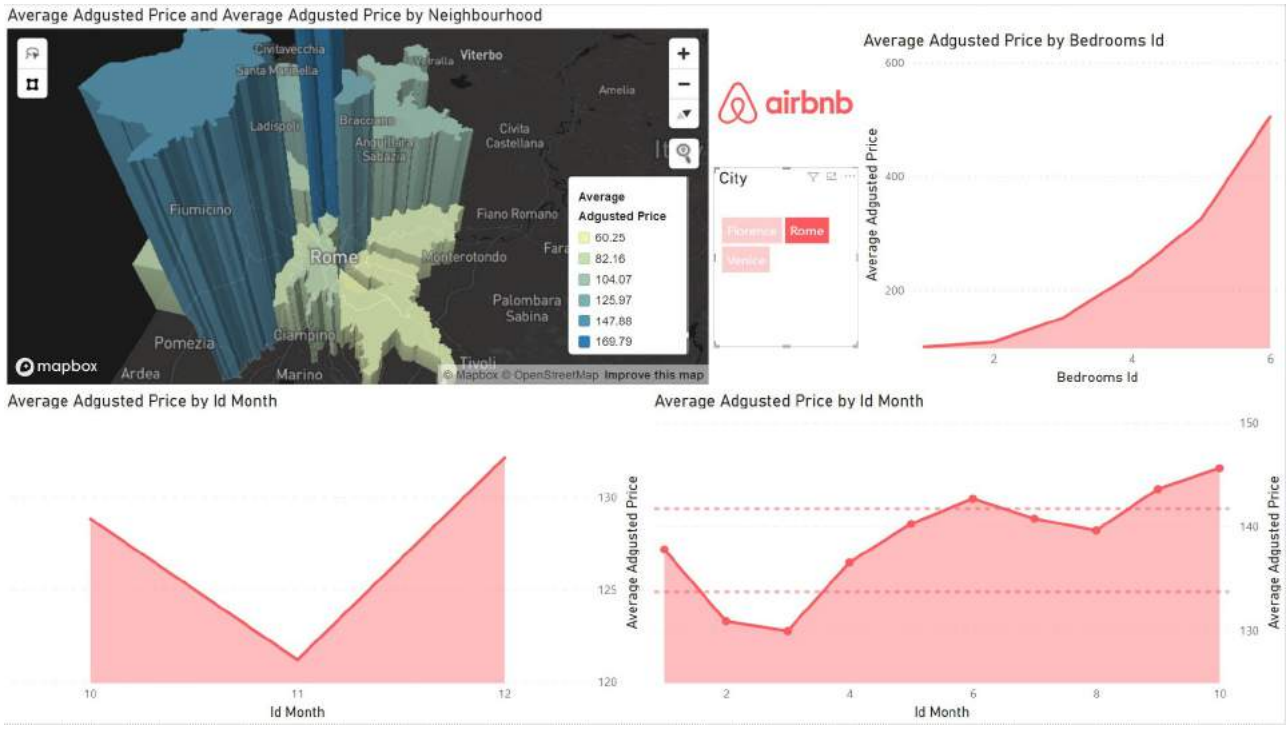


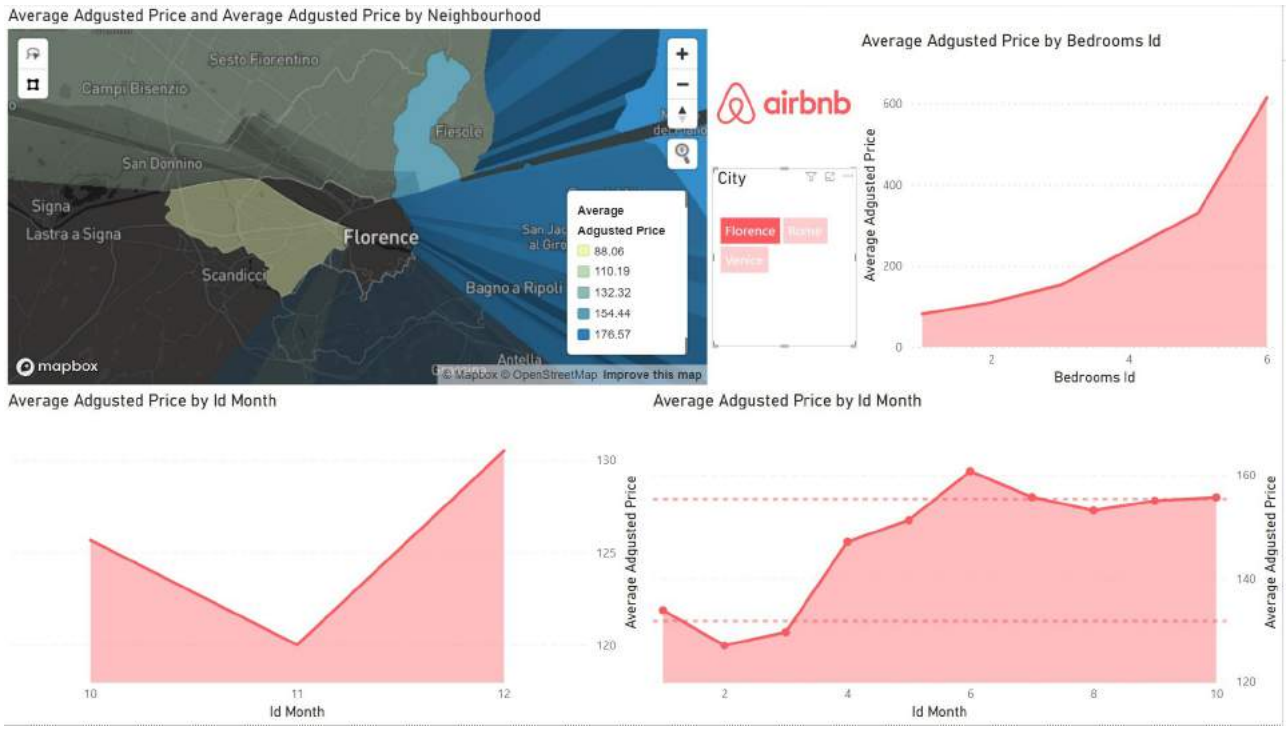
- It seems that most reservations were made for the listings with bedrooms in range one to three and then we notice that studios(listings with zero bedrooms) are following. It's logical that the 1-bedroom listings have the biggest demand, because it's the most common type.
- For 2021, we notice that for Venice the occupancy rate is bigger for November than December. The same is observed for Rome. And Florence's has its biggest occupancy rate in November.
- There is a noticeable increase in reservations for Florence in November and December. Which makes sense because Florence is a winter destination.
- Also, we can see from the map that the most booked regions are usually in or around the center. We can drilldown by selecting a specific region on the map and observe the value of that.

d. **Price Analysis**

We use the adjusted price because this is the real price with which the listings is reserved.



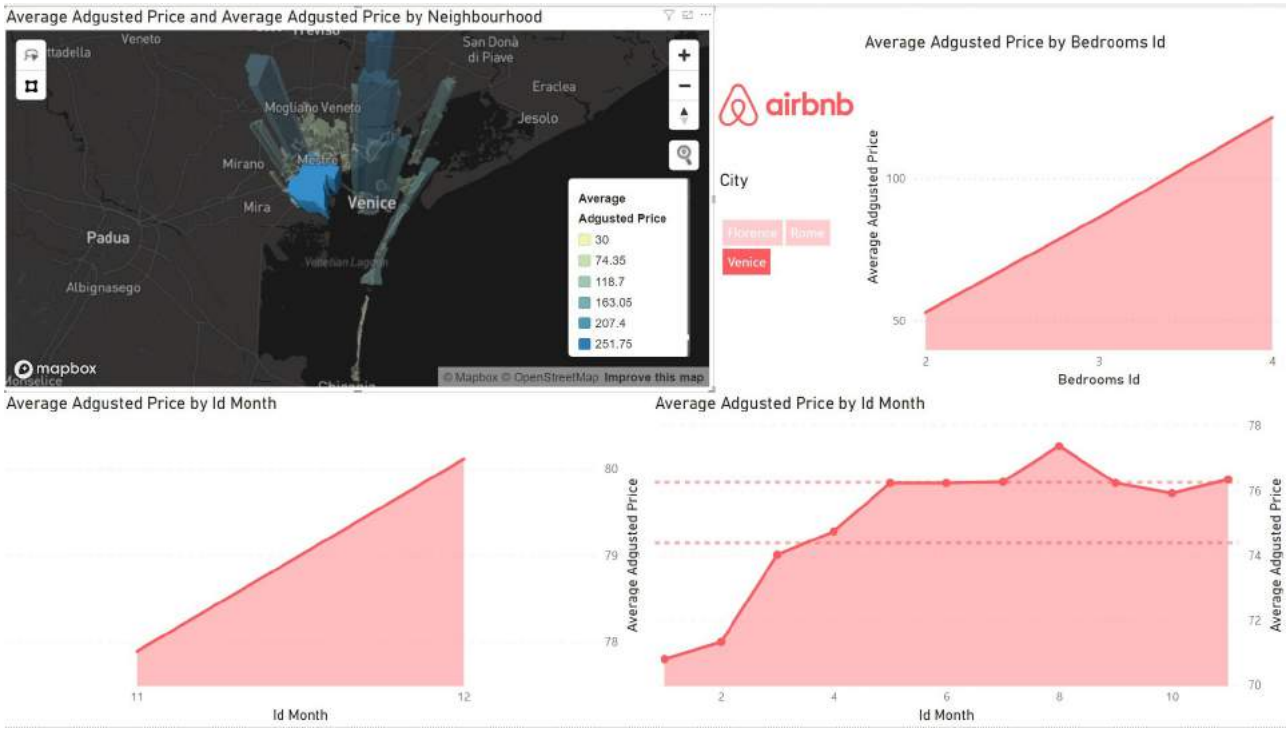




Generally:

- For 2021, we observe a significant decrease of the price on November and an increase on December because of the demand on the Christmas holidays.
- For 2022, we observe that there is an increase in price from March and it is steadily improve until it reaches the 75th percentile on June.

- **Drilldown by region**



- We can reach the same conclusion with other filters selected.