**DAMG 6210: DMDD Assignment 4 - Normalization**

**Project Title:** Realty to Reality – The search for your dream home ends now!

**Team Members:**

* Shashank Dongre, 002747740, [dongre.s@northeastern.edu](mailto:dongre.s@northeastern.edu), <https://github.com/ShashankDongre>
* Riya Virani, 002747048, [virani.r@northeastern.edu](mailto:virani.r@northeastern.edu),  <https://github.com/riya-virani>
* Ojasvi Patel, 002793770, [patel.oj@northeastern.edu](mailto:patel.oj@northeastern.edu), <https://github.com/Ojasvi1329>
* GitHub URL: <https://github.com/ShashankDongre/Realty-to-Reality---Find-your-dream-home-here->

**Assignment 4:**

We created a database which answers questions pertaining to house listings, selling, and buying specific to our use cases. We framed and structured SQL queries using MySQL relational database management system. We drafted an object model/UML diagram representing relationships between entities and their attributes. We then scraped the popular house listing web portal named Zillow specific to our area of focus using python’s web scraping capabilities and libraries. We then performed data cleaning, data wrangling, data munging, etc. using powerful libraries of python. We used Python’s Seaborn library for our specific data visualization purposes. The python notebook contains the code and analysis pertaining to the data visualization as well. The resultant dataset was then imported to the MySQL server and used further our queries and analysis.

**Technologies used:**

* MySQL – Relational Database Management System
* Python - Web Scraping, Data wrangling, Data cleaning, Data auditing
* Libraries – pandas, NumPy, seaborn, regular expression
* MS Excel – Data extraction, basic formatting
* Kaggle: https://www.kaggle.com/datasets/polartech/real-estate-information-in-boston
* MS Office - Project operation and execution

**Physical ER Diagram of the model:**

**Diagram

Description automatically generated**

**Explanation on some of the design decisions:**

● The ‘House\_Listing’ table contains details extracted from Zillow’s webpages of various house listings either by a seller or an agency. In this table the unique attribute i.e., ‘property\_id’ is the primary key whereas ‘agency\_id’ is the foreign key. This table has data which is essential while buying a house. We carefully extracted the limited number of attributes as data is the most valuable asset of Zillow and hence, they try hard to hide as much data as possible on a single page using various drop-down arrangements and hidden UI features of a webpage. This table contains the following desired attributes: property\_id, agency\_id, bedroom\_number, bathroom\_number, living\_space, land\_space, property\_status, and property\_type.

● Zestimate: Zillow's assessment of a house's market worth is based on the Zestimate® home valuation methodology. A Zestimate considers the specifics of the house, the neighborhood, and market trends in addition to data from the MLS, public sources, and user submissions. It cannot be used in place of an appraisal since it is not an appraisal. The Zestimate for on-market properties has a median error rate of 3.2% throughout the country, while the Zestimate for off-market homes has a median error rate of 7.52%. The regional data accessibility will determine how accurate the Zestimate is for a certain residence. Some places have access to more specific housing data, such square footage and the number of bedrooms or bathrooms, whereas others do not. The Zestimate number will be more precise the more data there are.

● The ‘Agency’ table is a simple table containing the details regarding the agency name and it has a unique ‘agency\_id’ as its primary key. Since this table is linking the agency name to the other tables, it need not have to have a foreign key for this purpose.

●. The ‘Price\_table’ contains the extracted details regarding the price and Zestimate of the listed property. In this table, ‘price\_id’ is assigned as a primary key. The unique attribute ‘property\_id’ is referenced in the address table using the foreign key. This table contains price details such as property\_id, price, price per unit of the land, and Zillow’s propriety attribute named ‘Zestimate’.

●. The ‘Address\_table’ contains the extracted details regarding the address of the listed property. In this table, ‘address\_id’ is assigned as a primary key. The unique attribute ‘property\_id’ is referenced in the address table using the foreign key. This table contains address details such as street\_name, rstate, city, and postcode.

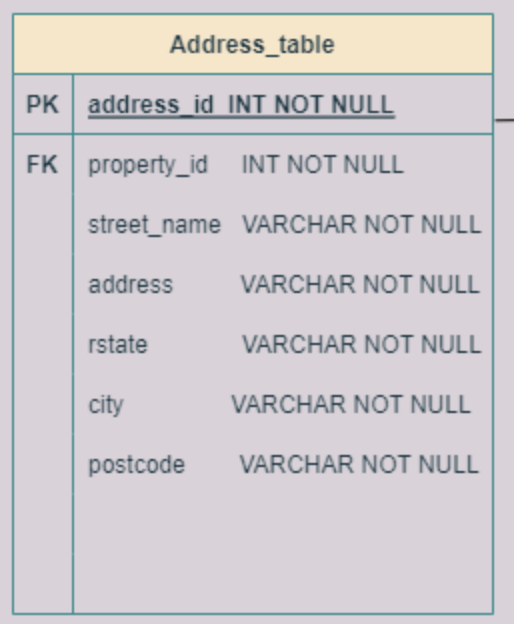
**Audit Validity/ Accuracy/ Audit Completeness/ Audit Consistency/Uniformity:**

The data that we scraped from Zillow’s webpage is specific to Boston city. Kindly note that this data belongs to the time when we scraped it from Zillow’s website. This may contain some old listings as well. Some listings may not be present during the time of the review. The scrapper was able to scrape more than two thousand records before it was stopped by Zillow’s webpage security mechanisms. The extraction of this dataset into a csv file implied that some records had to be deleted. The basic formatting of the data was conducted in MS Excel, which again filtered data for Boston city in several other states such as there are 16 cities in the United States named as ‘Boston’. This implied that the number of records drastically got reduced and hence we were reduced to almost 750 records for Boston, Massachusetts.

All data cleaning, munging, wrangling was performed in google collab notebook using python and its powerful libraries. The detail code can be found in the python notebook.

**Database Normalization:**

Table 1 – Address\_Table



**1st Normal Form:**

1. It has primary key with minimal attributes. (PK = address\_id)
2. The values in each column of a table are NOT atomic (Multi-value attributes are present).
3. There are no repeating groups.

The address\_table is not in 1st Normal Form.

To convert this table to 1NF, we dropped the address column, because the multi-value data in that column was already present in 3 different columns – street\_name, city, rstate, and postcode.

**2nd Normal Form:**

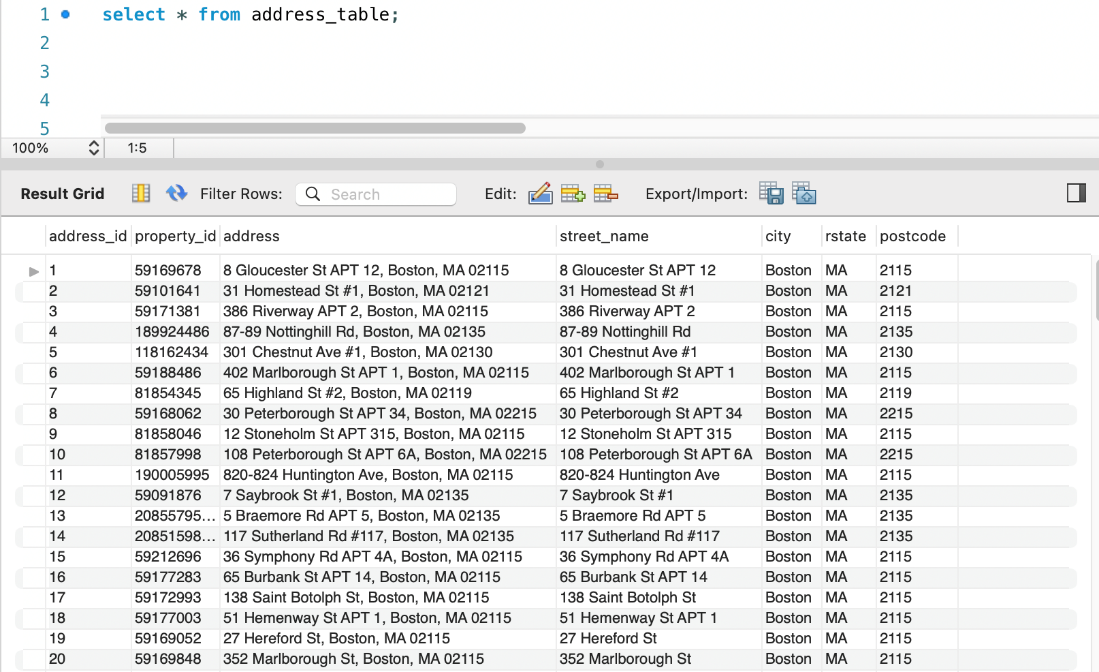
1. After the conversion, the table will be in 1st Normal form.
2. All non-key attributes are fully dependent on Primary key.

The address\_table table is in 2nd Normal form.

**3rd Normal Form:**

1. It is in the 2nd Normal Form.
2. It has no transitive dependencies.

The address\_table is in 3rd Normal form. Therefore, the address\_table table is now Normalized.



**Data before 1NF is shown above**

Graphical user interface, text, application

Description automatically generated

**Data after 3NF is shown above**

**Table structure after normalization of address\_table**

Table

Description automatically generated

Table 2 – House\_Listing

Table

Description automatically generated

**1st Normal Form:**

1. It has primary key with minimal attributes. (PK = property\_id)
2. The values in each column of a table are atomic (No multi-value attributes are present).
3. There are no repeating groups.

The house\_listing table is in 1st Normal Form.

**2nd Normal Form:**

1. After the conversion, the table will be in 1st Normal form.
2. All non-key attributes are fully dependent on Primary key.

The house\_listing table is in 2nd Normal form.

**3rd Normal Form:**

1. It is in the 2nd Normal Form.
2. It has no transitive dependencies.

The house\_listing is in 3rd Normal form. Therefore, the house\_listing table is Normalized.

Table

Description automatically generated with medium confidence

**Data of house\_listing table is shown above**

Table 3 – Agency

Graphical user interface, table

Description automatically generated with medium confidence

**1st Normal Form:**

1. It has primary key with minimal attributes. (PK = agency\_id)
2. The values in each column of a table are atomic (No multi-value attributes are present).
3. There are no repeating groups.

The agency table is in 1st Normal Form.

**2nd Normal Form:**

1. After the conversion, the table will be in 1st Normal form.
2. All non-key attributes are fully dependent on Primary key.

The agency table is in 2nd Normal form.

**3rd Normal Form:**

1. It is in the 2nd Normal Form.
2. It has no transitive dependencies.

The agency is in 3rd Normal form. Therefore, the agency table is Normalized.

Graphical user interface

Description automatically generated with medium confidence

**Data of agency table is shown above**

Table 4 – Price\_Table

Table

Description automatically generated

**1st Normal Form:**

1. It has primary key with minimal attributes. (PK = price\_id)
2. The values in each column of a table are atomic (No multi-value attributes are present).
3. There are no repeating groups.

The price\_table is in 1st Normal Form.

**2nd Normal Form:**

1. After the conversion, the table will be in 1st Normal form.
2. All non-key attributes are fully dependent on Primary key.

The price\_table is in 2nd Normal form.

**3rd Normal Form:**

1. It is in the 2nd Normal Form.
2. It has no transitive dependencies.

The price\_table is in 3rd Normal form.

Therefore, the agency table is Normalized.

Table

Description automatically generated

**Data of price\_table is shown above**

**Updated ER Diagram of the model:**

Diagram

Description automatically generated

**ER Diagram after normalization of all tables is shown above**

**SQL - CREATE VIEW STATEMENTS**

**Use Case 1**: Finding properties pertaining to our specific search criteria

Precondition: The listings should be listed on Zillow’s website.

Steps: We select all listings from our master house listing table. We select prices and post code from two tables and perform JOIN operation on the unique attribute ‘property\_id’. We then specify the postcode criteria using the WHERE clause. Please note that these query conditions can be slightly altered producing search results for houses and their prices for various zip code and locations within Boston.

System Responses: The system instantly returns all such results that meet our search criteria.

View Statement Query:

CREATE VIEW House\_Search AS

select \* from house\_listing

where bedroom\_number<4 and living\_space>2000;

select \* from House\_Search;

**Use Case 2:** Find the properties that are owned by a “single-family”

Precondition: The listings should be listed on Zillow’s website.

Steps: We select all listings from our master house listing table. We select prices and post code from two tables and perform JOIN operation on the unique attribute ‘property\_id’. We then specify the postcode criteria using the WHERE clause. Please note that these query conditions can be slightly altered producing search results for houses and their prices for various zip code and locations within Boston.

System Responses: The system instantly returns all such results that meet our search criteria.

View Statement Query:

CREATE VIEW Family\_Type AS

select \* from house\_listing

where property\_type like '%single\_family%';

**Use Case 3**: Find the status of properties that are “pending”

Precondition: The listings should be listed on Zillow’s website.

Steps: We select all listings from our master house listing table. We select prices and post code from two tables and perform JOIN operation on the unique attribute ‘property\_id’. We then specify the postcode criteria using the WHERE clause. Please note that these query conditions can be slightly altered producing search results for houses and their prices for various zip code and locations within Boston.

System Responses: The system instantly returns all such results that meet our search criteria.

View Statement Query:

CREATE VIEW Pending\_Properties AS

select \* from house\_listing

where property\_status like '%pending%';

**Use Case 4**: Find the prices of all listed houses in a particular postcode

Precondition: The listings should be listed on Zillow’s website.

Steps: We select all listings from our master house listing table. We select prices and post code from two tables and perform JOIN operation on the unique attribute ‘property\_id’. We then specify the postcode criteria using the WHERE clause. Please note that these query conditions can be slightly altered producing search results for houses and their prices for various zip code and locations within Boston.

System Responses: The system instantly returns all such results that meet our search criteria.

View Statement Query:

CREATE VIEW PostCode\_Search AS

select p.price, a.postcode from price\_table p

join address\_table a on a.property\_id = p.property\_id

where postcode = '2215';

**Use Case 5**: Find all condos with exactly 2 bathrooms.

Precondition: The listings should be listed on Zillow’s website.

Steps: We select prices, number of bathrooms from two tables and perform JOIN operation on the unique attribute ‘property\_id’. We then specify the number of bathroom and house type criteria using the WHERE clause. Please note that these query conditions can be slightly altered producing a combination of search results for various types of houses and bathroom numbers.

System Responses: The system instantly returns all such results that meet our search criteria.

View Statement Query:

CREATE VIEW House\_Details AS

select p.price, h.bathroom\_number, h.property\_type from price\_table p

join house\_listing h on p.property\_id = h.property\_id

where h.bathroom\_number=2 and h.property\_type like '%condo%';

**Use Case 6**: Find the Zestimate of multi-family houses.

Precondition: The listings should be listed on Zillow’s website with reasonable Zestimate score.

Steps: We select Zestimate, property type from two tables and perform JOIN operation on the unique attribute ‘property\_id’. We then specify the property type criteria using the WHERE clause. Please note that these query conditions can be slightly altered producing a combination of search results for various house types.

System Responses: The system instantly returns all such results that meet our search criteria.

View Statement Query:

CREATE VIEW Property\_type AS

select p.zestimate, h.property\_type from price\_table p

join house\_listing h on p.property\_id = h.property\_id

where h.property\_type like '%multi\_family%';

**Use Case 7**: Find the addresses of the listed houses with area greater than 1300 sqft.

Precondition: The listings should be listed on Zillow’s website.

Steps: We select street name, postcode, and area attributes from two tables and perform JOIN operation on the unique attribute ‘property\_id’. We then specify ‘land\_space’ attribute with suitable value using the WHERE clause. Please note that these query conditions can be slightly altered producing a combination of search results for various areas and locations.

System Responses: The system instantly returns all such results that meet our search criteria.

View Statement Query:

CREATE VIEW Area\_Less\_than AS

select a.street\_name, a.postcode, h.land\_space from address\_table a

join house\_listing h on h.property\_id = a.property\_id

where h.land\_space<1300;

**Use Case 8**: Find the addresses of the listed houses which are under-valued according to Zillow.

Precondition: The listings should be listed on Zillow’s website.

Steps: We select the unique attribute named ‘property\_id’, address, price and Zestimate attributes from two tables and perform JOIN operation on the unique attribute ‘property\_id’. We then specify the condition of Zestimate being less than the listed price using the WHERE clause. Please note that these query conditions can be slightly altered producing a combination of search results for various areas and locations. This would also return the list of over-valued properties as well.

System Responses: The system instantly returns all such results that meet our search criteria.

View Statement Query:

CREATE VIEW Undervalued\_Properties AS

select p.property\_id, a.street\_name, a.city, a.rstate, a.postcode, p.price, p.zestimate from price\_table p

join address\_table a on a.property\_id = p.property\_id

where p.zestimate < p.price;

**Use Case 9**: Find the average Zestimate for a given zip code.

Precondition: The listings should be listed on Zillow’s website and have a reasonable Zestimate score.

Steps: In this query we use the aggregate function named ‘AVG’.

We select the Zestimate and apply the average aggregate function as average\_zestimate, postcode from two tables and perform JOIN operation on the unique attribute ‘property\_id’. We then use the ‘GROUP BY’ aggregate function on postcode. Please note that these query conditions can be slightly altered producing using other aggregate functions such as ‘MIN’, ‘MAX’, ‘COUNT’, etc.

System Responses: The system instantly returns all such results that meet our search criteria.

View Statement Query:

CREATE VIEW Average\_Zestimate AS

select avg(p.zestimate) as average\_zestimate, a.postcode from price\_table p

join address\_table a on a.property\_id = p.property\_id

group by a.postcode;

**Use Case 10**: Find the addresses of the listed houses which are over-valued according to Zillow.

Precondition: The listings should be listed on Zillow’s website with a reasonable Zestimate score.

Steps: We select the unique attribute named ‘property\_id’, address, price and Zestimate attributes from two tables and perform JOIN operation on the unique attribute ‘property\_id’. We then specify the condition of Zestimate being greater than the listed price using the WHERE clause. Please note that these query conditions can be slightly altered producing a combination of search results for various areas and locations. This would also return the list of under-valued properties as well.

System Responses: The system instantly returns all such results that meet our search criteria.

View Statement Query:

CREATE VIEW Overvalued\_Properties AS

select a.street\_name, a.postcode, p.zestimate, p.price from price\_table p

join address\_table a on a.property\_id = p.property\_id

where p.zestimate > p.price;

**Use Case 11**: Find all listed houses within the specified area and conditions.

Precondition: The listings should be listed on Zillow’s website.

Steps: We select the unique attribute named ‘property\_id’, bedroom\_number, bathroom\_number, living\_space, and land\_space attributes two tables and perform JOIN operation on the unique attribute ‘property\_id’. We then specify the condition of land price per unit area between $300 and $1500 per square feet using the ‘WHERE’ clause. Please note that these query conditions can be slightly altered producing a combination of search results for various locations and parameters.

System Responses: The system instantly returns all such results that meet our search criteria.

View Statement Query:

CREATE VIEW Price\_Range AS

select h.property\_id, bedroom\_number, bathroom\_number, living\_space, land\_space,

property\_status, property\_type, price\_per\_unit from house\_listing h

join price\_table p on p.property\_id = h.property\_id

where price\_per\_unit between 300 and 1500;

**Use Case 12**: Find top 3 most expensive properties under a specific postcode.

Precondition: The listings should be listed on Zillow’s website.

Steps: We select the unique attribute named ‘property\_id’, postcode, and price attributes two tables and perform JOIN operation on the unique attribute ‘property\_id’. We then specify the condition of the specific postcode using the ‘WHERE’ clause. We then use the ‘ORDER BY’ statement to arrange these in descending order. We then limit the results to 3 to view the top three most expensive house listings available. Please note that these query conditions can be slightly altered producing a combination of search results for various locations and parameters.

System Responses: The system instantly returns all such results that meet our search criteria.

View Statement Query:

CREATE VIEW Top\_Three\_Properties AS

select price, postcode from price\_table p

join address\_table a on a.property\_id = p.property\_id

where postcode = '2115'

order by price desc

limit 3;

**Use Case 13**: Find address and other details of the listed houses which are neither under-valued nor over-valued, according to Zillow’s index ‘Zestimate’.

Precondition: The listings should be listed on Zillow’s website and have a reasonable Zestimate score.

Steps: We select the unique attribute named ‘property\_id’, address, bedroom\_number, bathroom\_number, living\_space, Zestimate, price attributes from three tables and perform JOIN operation on the unique attribute ‘property\_id’ in both join operations. The first join is between ‘house\_listing’ and ‘address’ tables whereas the second join is between ‘price\_table’ and ‘address’ table. We then specify the condition of the specific in which Zestimate equals the price of the property using the ‘WHERE’ clause. Please note that these query conditions can be slightly altered producing a combination of search results for various locations and parameters.

System Responses: The system instantly returns all such results that meet our search criteria.

View Statement Query:

CREATE VIEW Same\_Price AS

select street\_name, city, rstate, postcode, bedroom\_number, bathroom\_number, living\_space, zestimate, price from address\_table a

join house\_listing h on h.property\_id = a.property\_id

join price\_table p on p.property\_id = a.property\_id

where zestimate = price;

**Use Case 14**: Find top 3 popular agencies with highest number of listings.

Precondition: The listings should be listed on Zillow’s website. The listings must have at least one agency associated to it.

Steps: We select the name of the agency and use aggregate function ‘COUNT’ on the attribute ‘agency\_id’ as ‘famous\_agency’ from two tables and perform JOIN operation on the attribute ‘agency\_id’. We then use the aggregate function named ‘GROUP BY’ to group all agencies with their unique ids. The query then performs the descending arrangement of the names of the agencies using the ‘ORDER BY’ statement. We limit the number of results to three as we have initially asked to view these three most popular agencies. Please note that these query conditions can be slightly altered producing a combination of search results for other parameters and arrangements.

System Responses: The system instantly returns all such results that meet our search criteria.

View Statement Query:

CREATE VIEW Famous\_Agency AS

select count(h.agency\_id) as famous\_agency, agency\_name from house\_listing h

join agency a on a.agency\_id = h.agency\_id

group by h.agency\_id

order by famous\_agency desc

limit 3;

**Use Case 15**: Find property and agency details of the desired properties.

Precondition: The listings should be listed on Zillow’s website.

Steps: We select the unique attribute named ‘property\_id’, bedroom\_number, bathroom\_number, living\_space, agency\_name from two tables and perform JOIN operation on the attribute ‘agency\_id’. We then specify the condition of number of bedrooms greater than 3 using the ‘WHERE’ clause. Please note that these query conditions can be slightly altered producing a combination of search results for various number of bedrooms or bathrooms among many other possible combinations.

System Responses: The system instantly returns all such results that meet our search criteria.

View Statement Query:

CREATE VIEW Greater\_Than\_Three AS

select h.property\_id, bedroom\_number, bathroom\_number, living\_space, agency\_name from house\_listing h

join agency a on a.agency\_id = h.agency\_id

where bedroom\_number>3;

**Use Case 16**: Find property and agency details of the pending properties (not yet sold or bought).

Precondition: The listings should be listed on Zillow’s website.

Steps: We select the attribute named ‘agency\_id’, the unique attribute named ‘property\_id’, bedroom\_number, bathroom\_number, living\_space, property\_status, agency\_name from two tables and perform JOIN operation on the attribute ‘agency\_id’. We then specify the condition of the status of property as pending using the ‘WHERE’ clause.

System Responses: The system instantly returns all such results that meet our search criteria.

View Statement Query:

CREATE VIEW Pending\_Properties\_of\_an\_agency AS

select h.property\_id, bedroom\_number, bathroom\_number, living\_space, property\_status, agency\_name from house\_listing h

join agency a on a.agency\_id = h.agency\_id

where property\_status = 'PENDING';

**Use Case 17**: Find property which costs more than $7000000

Precondition: The listings should be listed on Zillow’s website.

Steps: We select the unique attribute named ‘property\_id’, address, price from two tables and perform JOIN operation on the attribute ‘property\_id’. We then specify the condition of the price more than $7000000 using the ‘WHERE’ clause.

System Responses: The system instantly returns all such results that meet our search criteria.

View Statement Query:

CREATE VIEW Expensive\_Properties AS

select a.street\_name, a.city, a.rstate, a.postcode, p.price from address\_table a

join price\_table p on p.property\_id = a.property\_id

where p.price>7000000;

**Use Case 18**: Find all properties which has living area greater than 500 sqft.

Precondition: The listings should be listed on Zillow’s website and have a reasonable Zestimate score.

Steps: We select the unique attribute named ‘property\_id’, price, Zestimate, living\_space, land\_space from two tables and perform JOIN operation on the attribute ‘property\_id’. We then specify the condition of living space greater than 500 sqft using the ‘WHERE’ clause.

System Responses: The system instantly returns all such results that meet our search criteria.

View Statement Query:

CREATE VIEW Property\_Details AS

select p.price, p.zestimate, h.living\_space, h.land\_space from price\_table p

join house\_listing h on h.property\_id = p.property\_id

where h.living\_space>500;