**DAMG 6210: DMDD Assignment 3 - Gathering, Scraping, Munging and Cleaning Data**

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

**Assignment 3:** 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. The seaborn library for data visualization. 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
* Source of data - <https://tinyurl.com/y5pjwy67> (Zillow Homepage)
* 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 ‘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 details regarding the addresses of the listed properties. This table has ‘address\_id’ as its primary key. The unique attribute ‘property\_id’ is referenced in the address table using the foreign key. This table contains address details such as property\_id, street\_name, address, city, rstate, and postcode.

● 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.

**Downloading and reformatting of the data:** 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.

Audit Validity/ Accuracy

Audit Completeness

Audit Consistency/Uniformity (10 points)

**SQL - CREATE TABLE STATEMENTS**

CREATE TABLE `address\_table` (

`address\_id` int NOT NULL AUTO\_INCREMENT,

`property\_id` int DEFAULT NULL,

`address` varchar(1000) DEFAULT NULL,

`street\_name` varchar(500) DEFAULT NULL,

`city` varchar(45) DEFAULT NULL,

`rstate` varchar(45) DEFAULT NULL,

`postcode` varchar(10) DEFAULT NULL,

PRIMARY KEY (`address\_id`),

);

CREATE TABLE `agency` (

`agency\_id` int NOT NULL AUTO\_INCREMENT,

`agency\_name` varchar(500) DEFAULT NULL,

PRIMARY KEY (`agency\_id`)

);

CREATE TABLE `house\_listing` (

`property\_id` int NOT NULL,

`agency\_id` int DEFAULT NULL,

`bedroom\_number` int DEFAULT NULL,

`bathroom\_number` int DEFAULT NULL,

`living\_space` int DEFAULT NULL,

`land\_space` decimal(9,2) DEFAULT NULL,

`property\_status` varchar(45) DEFAULT NULL,

`property\_type` varchar(45) DEFAULT NULL,

PRIMARY KEY (`property\_id`),

);

CREATE TABLE `price\_table` (

`price\_id` int NOT NULL AUTO\_INCREMENT,

`property\_id` int DEFAULT NULL,

`zestimate` int DEFAULT NULL,

`price` int DEFAULT NULL,

`price\_per\_unit` int DEFAULT NULL,

PRIMARY KEY (`price\_id`),

);

**SQL - INSERT DATA STATEMENTS**

INSERT INTO house\_listing (property\_id, bedroom\_number, bathroom\_number, living\_space, land\_space, property\_status, property\_type)

SELECT property\_id, bedroom\_number, bathroom\_number, living\_space, land\_space, property\_status, property\_type FROM df;

INSERT INTO price\_table(property\_id, zestimate, price, price\_per\_unit)

SELECT property\_id, zestimate, price, price\_per\_unit from df;

INSERT INTO address\_table(property\_id, address, street\_name, city, rstate, postcode)

SELECT property\_id, address, street\_name, city, rstate, postcode from df;

INSERT INTO agency (agency\_name)

SELECT distinct agency\_name from df;

UPDATE house\_listing h, df d

SET h.agency\_id =

(SELECT a.agency\_id from agency a

JOIN df d ON d.agency\_name=a.agency\_name

WHERE h.property\_id = d.property\_id)

WHERE h.property\_id = d.property\_id;

**FOREIGN KEY CONSTRAINTS**

alter table address\_table

add constraint FK\_address\_house

foreign key (property\_id) references house\_listing(property\_id);

alter table price\_table

add constraint FK\_price\_house

foreign key (property\_id) references house\_listing(property\_id);

alter table house\_listing

add constraint FK\_house\_agency

foreign key (agency\_id) references agency(agency\_id);

**Use Cases:**

**Use Case 1**: 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.

**Use Case 2**: 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.

**Use Case 3**: 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.

**Use Case 4**: 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.

**Use Case 5**: 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.

**Use Case 6**: 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.

**Use Case 7**: 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.

**Use Case 8**: 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.

**Use Case 9**: 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.

**Use Case 10**: 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.

**Use Case 11**: 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.

**Use Case 12**: 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.

**Use Case 13**: 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.

**Use Case 14**: 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.

**Use Case 15**: 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.

**Response to any feedback received in Assignment 2:**

The Assignment Two - Scraping Twitter was based on creating a functioning bot in python to scrape specific data from the specific Twitter pages. The model and requirements needed to adhere to the assignment rubric to the fullest. This includes the creation of our R2R company which required each user to have a unique twitter account and interact on twitter to do some action. The basic working of the model is as follows: In assignment 2, in order to get house recommendations, each user had to post a tweet with his/her requirements on twitter. This tweet would then be used to provide results/recommendations to him/her. The purpose of R2R company was to gather the data for themselves and provide recommendations to the user. The reason user table cannot be connected to the house\_listing table is that he goes via the company R2R and follows the twitter model.

In assignment 3, we have optimized our database and have created a new schema and a corresponding E-R diagram. Therefore, we cannot implement the given feedback in assignment 3.

**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-