**Step 1: Connect to the Snowflake SQL**

* Download SNOWSQL : <https://developers.snowflake.com/snowsql/>
* Open cmd
* snowsql -a xsymtcm-em56590 -u amritaneogi
* Password: \*\*\*\*\*\*\*\*\*\*\*\*\*\*

Here, account\_name = xsymtcm-em56590

login\_name = amritaneogi

**Step 2:** **Create a virtual Schema or Database**

* CREATE DATABASE

Default schema available for each database

HOUSE\_PRICE;



* [amritaneogi#COMPUTE\_WH@HOUSE\_PRICE.PUBLIC](mailto:amritaneogi#COMPUTE_WH@HOUSE_PRICE.PUBLIC) >



Connected to the new database



**Step 3:** **Create a virtual Data Warehouse**; COMPUTE\_WH is the default warehouse provided.

Having a data warehouse in Snowflake is crucial, unlike other databases. The warehouse serves as the essential resource allocator for various computations, including running queries and performing SQL operations. This necessity arises from Snowflake's nature as a cloud-based data warehouse, where every operation consumes resources. The resources are used from the warehouse select, and based on this the cost is determined.

* CREATE WAREHOUSE PRICE\_WH;
* [amritaneogi#PRICE\_WH@HOUSE\_PRICE.PUBLIC](mailto:amritaneogi#PRICE_WH@HOUSE_PRICE.PUBLIC) >

**Step 4: Create a ‘Stage’.**

In other databases we load the file directly into the table. However, in Snowflake, we have to load it into a stage. A stage is nothing but a location which stores data files.

While creating stage we also need to mention the mention the format of the data files that it will store.

* CREATE OR REPLACE FILE FORMAT JSON\_FORMAT

TYPE = ‘JSON’

STRIP\_OUTER\_ARRAY = TRUE;

This is important to mention. This strips or eliminates the ‘[’ and ‘]’ at the beginning and end of the JSON file. If we skip this command and try to export a file, it will throw an error since it will treat the entire file as one single record.



# creating stage

* CREATE OR REPLACE STAGE OTODOM\_STAGE

FILE\_FORMAT = JSON\_FORMAT;

**Step 5: Export file in the Stage.**

We can directly export the file (either in .csv or JASON format) from the BrightData; but we need to purchase the file for that.

Instead, we will directly upload the file in the Snowflake 'Stage' that we created earlier.

* PUT file:///C:\Users\amrit\OneDrive\Documents\GitHub\Data\_Analytics\_Project-Hou

sing\_Price\_Profiler\Data\_Set\_Backup\Otodom\_Poland.json @OTODOM\_STAGE;

A screenshot of a computer program

Description automatically generated

This is now the Snowflake server look once all the changes are made:

A screenshot of a computer

Description automatically generated



A screenshot of a computer

Description automatically generated

**Step 6: Once the data is loaded into the stage, create a table.**

We are creating a table with only one column here ‘json\_data’.

*# writing codes on the snowflake console directly and not using cmd*.

Create table OTODOM\_DATA

(

json\_data VARIANT -- creating table with one column

);

‘VARIANT’ 🡪 this is a special kind of data type. It is used to store semi-structured data from JSON file.

**Step 7: Load data from stage to table.**

--copying data from stage to the table

copy into OTODOM\_DATA

from @OTODOM\_STAGE

on\_error= 'skip\_file'; -- skip any kind of error that could abandon the entire file.

***NOTE****: If the data is exported directly from BrightData, then it will have 2 extra files of one record each. This are the files that BrightData sends for the purpose of testing the connection.*

*Also, I am using the sample data of 1000 records that I got free from the BrightData. The original data count is approximately 500k.*

**Step 8: Flattening of file.**

The data we loaded is in JSON format, it is semi structured and not apt for any kind of analysis. So, we need to transform this into appropriate table columns. This process is called flattening of files.

Actual record (semi-structured):

{ "advertiser\_type": "agency", "description": "Chcesz mieszkać w ścisłym centrum Głogowa? A może poszukujesz nieruchomości, pod kątem inwestycyjnym? Świetnie! Mam dla Ciebie idealne rozwiązanie. Mieszkanie zlokalizowane jest na Placu Jana z Głogowa, nieopodal skweru z fontanną oraz Parku Słowiańskiego i Parku Sapera, zatem w otoczeniu zieleni.  W pobliżu odnajdziesz wszystko co jest niezbędne do funkcjonowania – sklepy spożywcze, piekarnię, cukiernię, punkty usługowe oraz gastronomiczne.  Blok posiada dodatkową bramę wejściową zabezpieczoną domofonem.  Mieszkanie charakteryzuje się 36,5 m2 powierzchni użytkowej. Nieruchomość posadowiona jest na parterze pięcio-kondygnacyjnego budynku i składa się z:     ● Salonu z bardzo słonecznym, dużym oknem oraz balkonem,     ● Aneksu kuchennego,  który jest oddzielony ścianką lecz bez problemu można go otworzyć na salon uzyskując większą przestrzeń,     ● Sypialni      ● Łazienki, wyposażonej w wannę,  umywalkę oraz WC,     ● Przedpokoju.  Mieszkanie jest utrzymane w bardzo dobrej kondycji, stolarka okienna pcv biała. Ściany zostały odświeżone, a nieruchomość nadaje się do zamieszkania.  Jest to również nieruchomość gotowa pod inwestycję, bowiem całe wyposażenie pozostaje do dyspozycji nowego nabywcy.  Ogrzewanie miejskie, mieszkanie z zasobów Spółdzielni Mieszkaniowej. Do nieruchomości przynależy również piwnica.   Chcesz dowiedzieć się więcej? Zadzwoń i umów się na bezpłatną prezentację nieruchomości.Istnieje możliwość negocjacji ceny.   Biuro Nieruchomości CUPRUM HOUSE Anna Wernik – Kaniewska Specjalista ds. nieruchomości tel. 723 411 712  \r\n\r\nOferta wysłana z programu dla biur nieruchomości ASARI CRM ()\r\n", " ": "pełna własność", "is\_for\_sale": true, "lighting": null, "location": "Głogów, głogowski, dolnośląskie", "market": [ "market", "secondary" ], "no\_of\_rooms": 2, "posting\_id": "4mSyJ", "price": "210000", "remote\_support": null, "surface": "36.5", "timestamp": "2023-09-20", "title": "Dwa pokoje w ścisłym centrum do zamieszkania", "url": "https://otodom.pl/pl/oferta/dwa-pokoje-w-scislym-centrum-do-zamieszkania-ID4mSyJ" }

For example:  
  
A screenshot of a data analysis

Description automatically generated

Remove the double quotes:

A screenshot of a computer

Description automatically generated

We will do the same for all other columns:

CREATE OR REPLACE table OTODOM\_DATA\_FLATTEN

as

select row\_number() over (order by title) as ID -- for creating a unique identifier for each row

, X.\*

from(

select

replace (JSON\_DATA: posting\_id,'"')::string as posting\_id,

replace (JSON\_DATA: advertiser\_type,'"')::string as advertiser\_type,

--replace (JSON\_DATA: balcony\_garden\_terrace,'"')::string as balcony\_garden\_terrace,

regexp\_replace(replace(JSON\_DATA:description, '"'), '<[^>]+>')::string as description,

--replace (JSON\_DATA: heating,'"')::string as heating,

replace (JSON\_DATA: is\_for\_sale,'"')::string as is\_for\_sale,

replace (JSON\_DATA: lighting,'"')::string as lighting,

replace (JSON\_DATA: location,'"')::string as location,

replace (JSON\_DATA: price,'"')::string as price,

replace (JSON\_DATA: remote\_support,'"')::string as remote\_support,

replace (JSON\_DATA: surface,'"')::string as surface,

replace (JSON\_DATA: timestamp,'"')::string as Date,

replace (JSON\_DATA: title,'"')::string as title,

replace (JSON\_DATA: url,'"')::string as url,

replace (JSON\_DATA: form\_of\_property,'"')::string as form\_of\_property,

replace (JSON\_DATA: no\_of\_rooms,'"')::string as no\_of\_rooms

--replace (JSON\_DATA: parking\_space,'"')::string as parking\_space

from OTODOM\_DATA

) X;

***Note****:*

*The code uses regexp\_replace to remove HTML tags from the "description" column. This is done by replacing anything that matches the regular expression '<[^>]+>' (which matches HTML tags) with an empty string.*

**Step 9: Transformation of data for further analysis**

1. Locations contains longitudinal and latitudinal values
2. Change the language from Polish to English

We will use Python and Google sheet for the same.

**Step 9.1: Create Python Virtual Environment**

Open Terminal from Jupyter Notebook

* conda create --name data\_analytics\_project python=3.11

All the required packages are installed with it at the same time

Changes done specific to my code ( ran code on Jupyter Notebook):

* conda active data\_analytics\_project
* conda install -n data\_analytics\_project jupyter

Open a new Jupyter notebook

* jupyter notebook



From the newly created Jupyter notebook

* conda install -n data\_analytics\_project -c conda-forge snowflake-sqlalchemy



* !pip install snowflake-sqlalchemy
* # import all the packages and libraries (refer the .ipynb file)