EMSE 6586

Data Translation of Yelp Data to Arango DB

Anantha Narayanan - G46252520

Pranav Parthasarathy - G25917423

INTRODUCTION

Yelp, as a leading platform for business reviews and recommendations, generates vast amounts of JSON-formatted data capturing valuable insights from users worldwide. However, to unleash the full potential of this data, efficient storage, querying, and analysis are essential.

Our Project Focus on the process of translating Yelp's JSON data into ArangoDB, a powerful multi-model database, known for its flexibility, scalability, and query capabilities. By leveraging ArangoDB's features, we can transform raw JSON data into a structured, queryable format, enabling businesses to extract actionable insights, enhance user experiences, and make data-driven decisions.

Python is a go-to language for a wide array of data tasks, including data pre-processing, ETL (Extraction, Transformation, Loading) development, and scripting. Its vast library ecosystem empowers developers with powerful tools for handling JSON data and seamlessly interfacing with databases

PROJECT WORKFLOW

01 - Data Collection

Yelp dataset in json format was collected from kaggle

02 - DB Creation

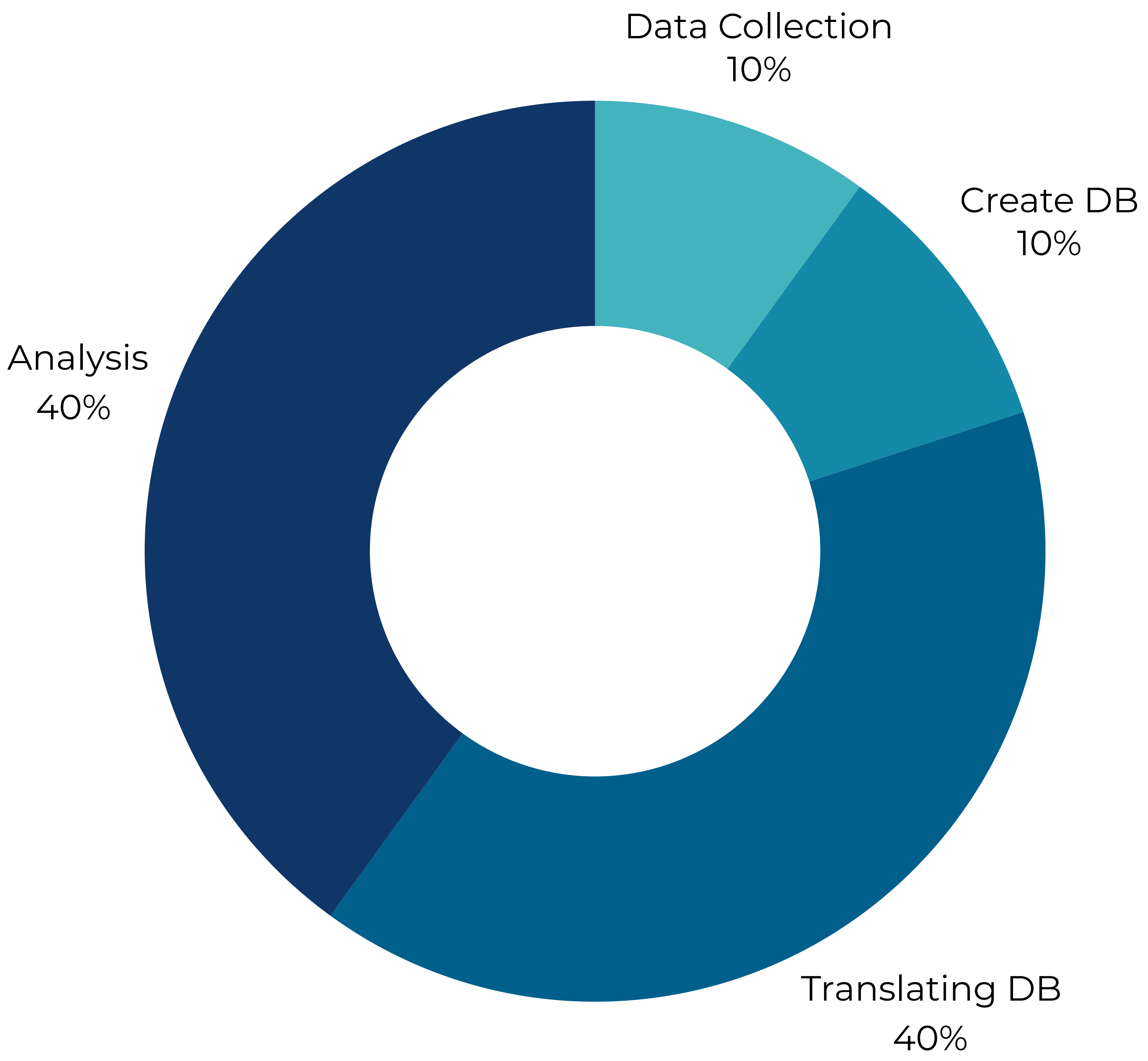
ArangoDB was hosted on the local Windows Machine

03 - DB Translation

Data from json was transformed and loaded into the ArangoDB

04 - Analysis

Analysis on the loaded was performed and interpreted

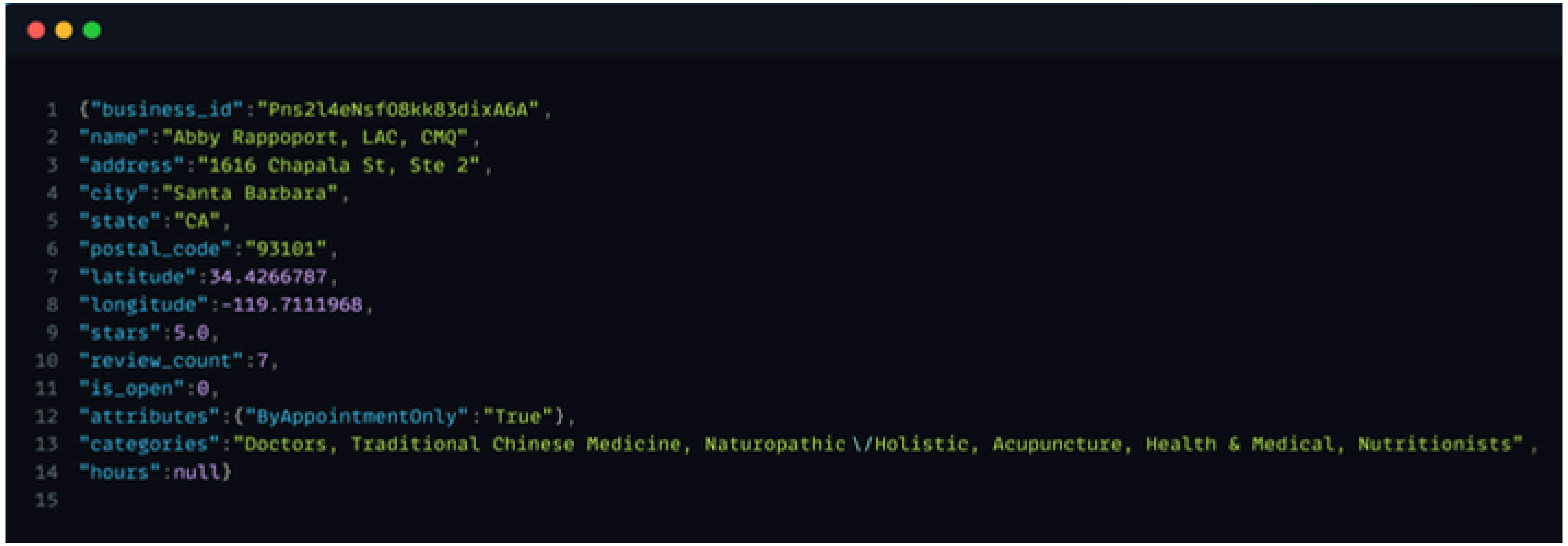


COLLECTED DATA: YELP JSON DATASET

- The **Yelp JSON dataset** is a comprehensive collection of data from the popular **business review platform**, Yelp.
- It encompasses a vast array of information, providing valuable **insights into businesses, user, reviews, checkins, and tips**.
- The Yelp JSON dataset is substantial, totaling approximately **10 GB of data** across its various files.
- This size underscores the richness and depth of information available for analysis, offering ample opportunities for extracting actionable insights and deriving value.

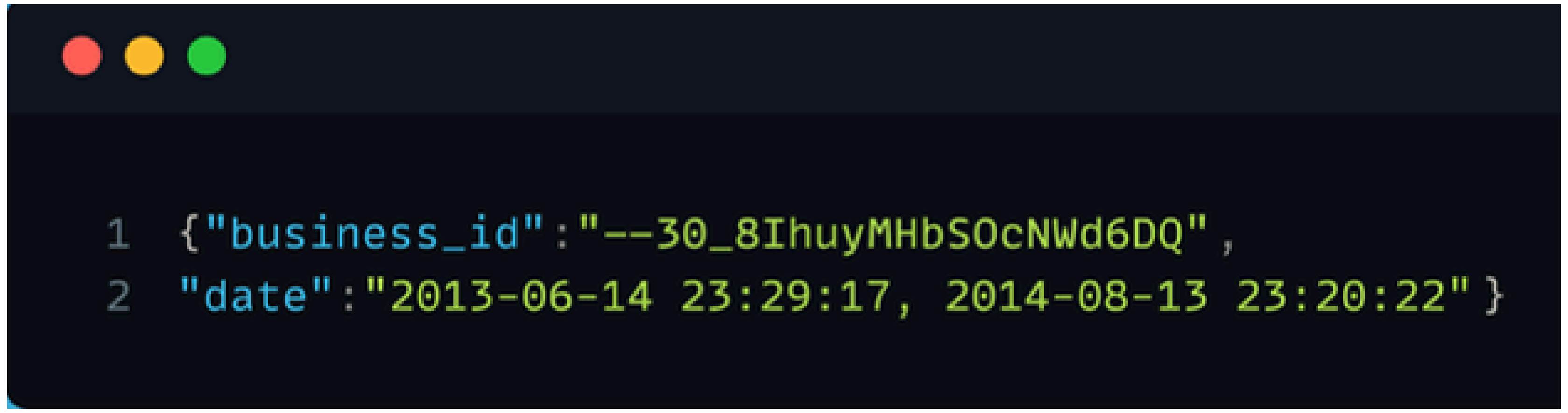
SAMPLE JSON DATA

business data. json



```
{"business_id":"Pns2l4eNsfO8kk83dixA6A",
"name":"Abby Rappoport, LAC, CMQ",
"address":"1616 Chapala St, Ste 2",
"city":"Santa Barbara",
"state":"CA",
"postal_code":"93101",
"latitude":34.4266787,
"longitude":-119.7111968,
"stars":5.0,
"review_count":7,
"is_open":0,
"attributes":{"ByAppointmentOnly":"True"},
"categories":"Doctors, Traditional Chinese Medicine, Naturopathic\/Holistic, Acupuncture, Health & Medical, Nutritionists",
"hours":null}
```

checking data.json

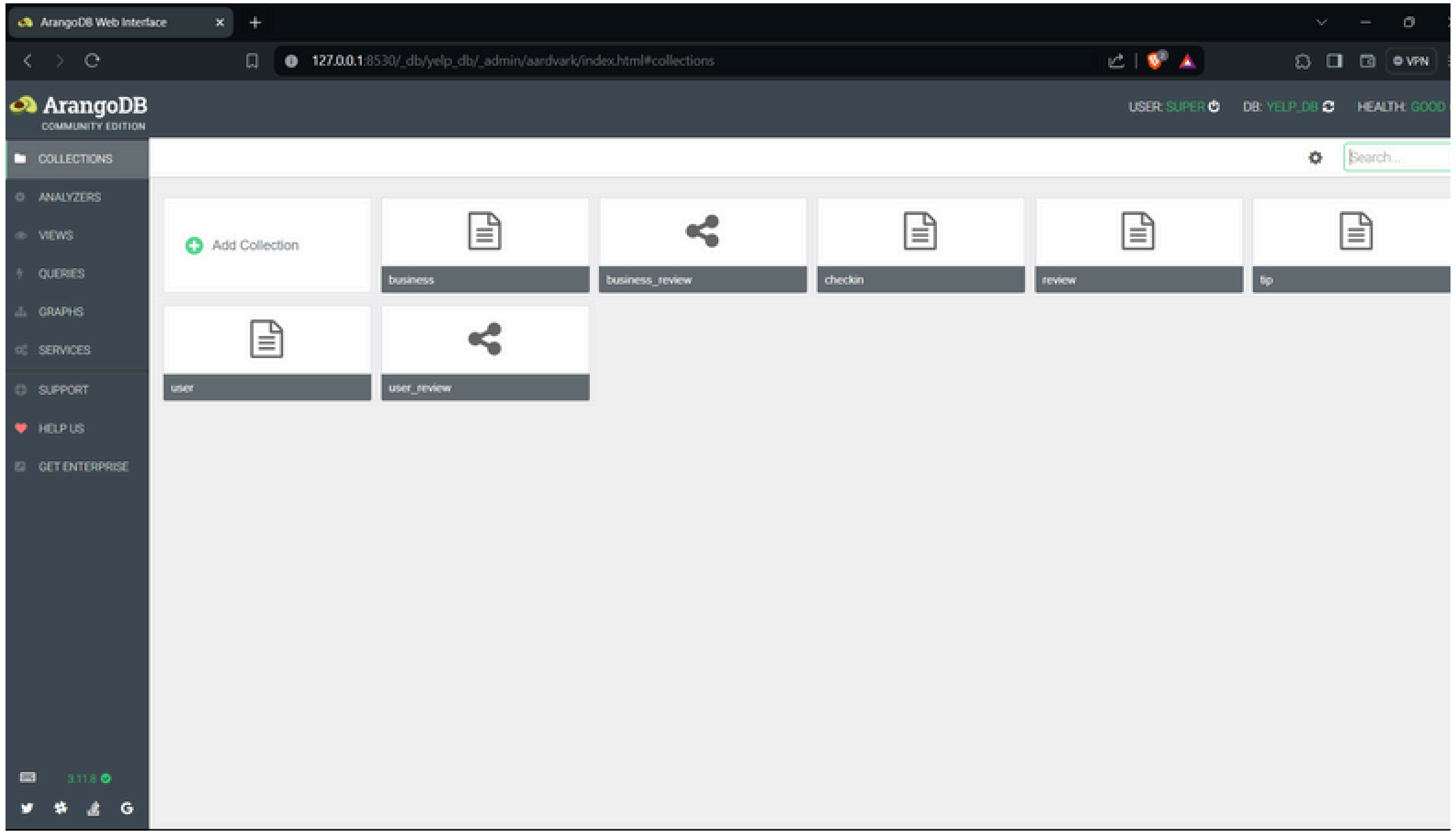


```
{"business_id":"--30_8IhuyMHbSOcNWd6DQ",
"date":"2013-06-14 23:29:17, 2014-08-13 23:20:22"}
```

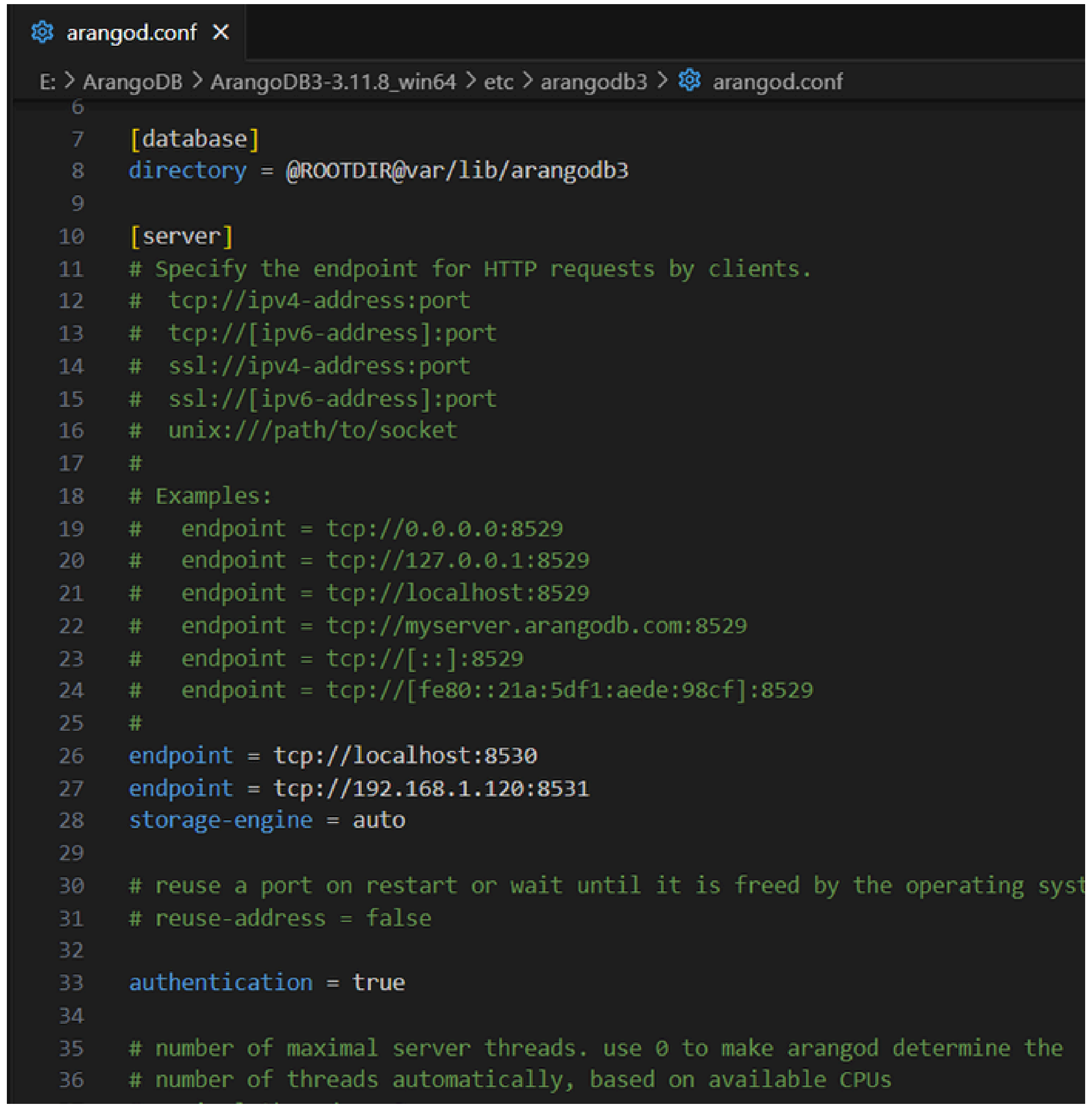
SETTING UP ARANGO DB

1. **Download ArangoDB:**
   - Visit the ArangoDB download page and select the appropriate version for your Windows system (32-bit or 64-bit).
   - Download version 3.11 (Support for native WIndows and macOS removed in v3.12)
2. **Install ArangoDB:**
   - Once the download is complete, run the installer.
   - Follow the installation wizard instructions. You can generally accept the default settings unless you have specific preferences.
3. **Start ArangoDB:**
   - Open Command shell and type arangod , to start arango server
4. **Configure ArangoDB :**
   - ArangoDB comes with default configurations that work for most cases. But for safety reasons , changed default port to 8530 and also added another endpoint http://192.168.1.120:8531 , to access the server on the local network.
5. **Access ArangoDB Web Interface:**
   - Open a web browser and go to http://127.0.0.1:8530.
   - You should see the ArangoDB web interface where you can manage your databases, collections, and more.

ARANGO DB INTERFACE AND CONF FILE



Arango DB Interface


arangod.conf
E: > ArangoDB > ArangoDB3-3.11.8_win64 > etc > arangodb3 > arangod.conf

```
[database]
directory = @ROOTDIR@var/lib/arangodb3

[server]
# Specify the endpoint for HTTP requests by clients.
#  tcp://ipv4-address:port
#  tcp://[ipv6-address]:port
#  ssl://ipv4-address:port
#  ssl://[ipv6-address]:port
#  unix:///path/to/socket
#
# Examples:
#   endpoint = tcp://0.0.0.0:8529
#   endpoint = tcp://127.0.0.1:8529
#   endpoint = tcp://localhost:8529
#   endpoint = tcp://myserver.arangodb.com:8529
#   endpoint = tcp://[::]:8529
#   endpoint = tcp://[fe80::21a:5df1:aede:98cf]:8529
#
endpoint = tcp://localhost:8530
endpoint = tcp://192.168.1.120:8531
storage-engine = auto

# reuse a port on restart or wait until it is freed by the operating syst
# reuse-address = false

authentication = true

# number of maximal server threads. use 0 to make arangod determine the
# number of threads automatically, based on available CPUs
```

Arango DB Conf File

ARANGO DB COLLECTIONS

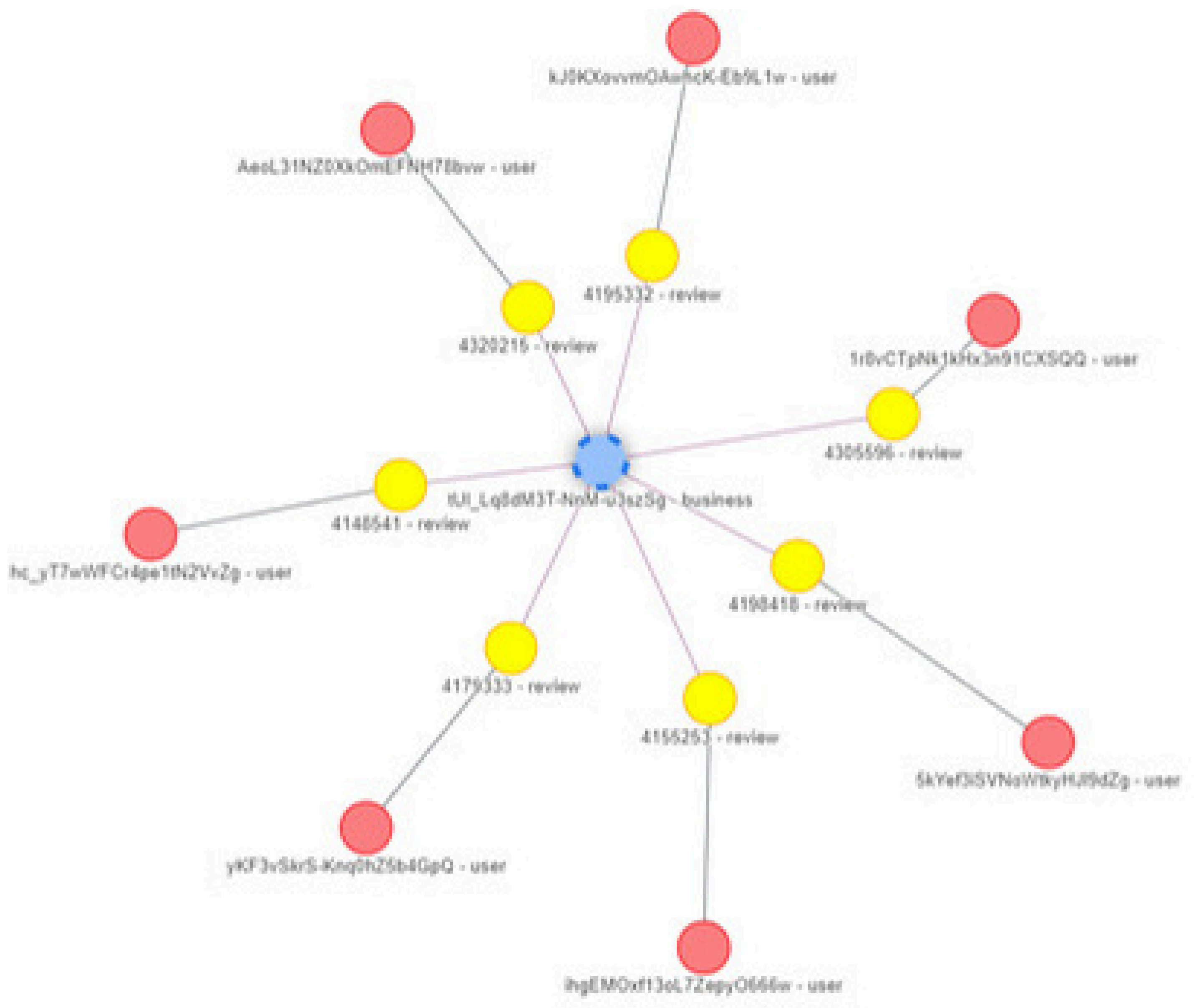
VERTEX COLLECTIONS:

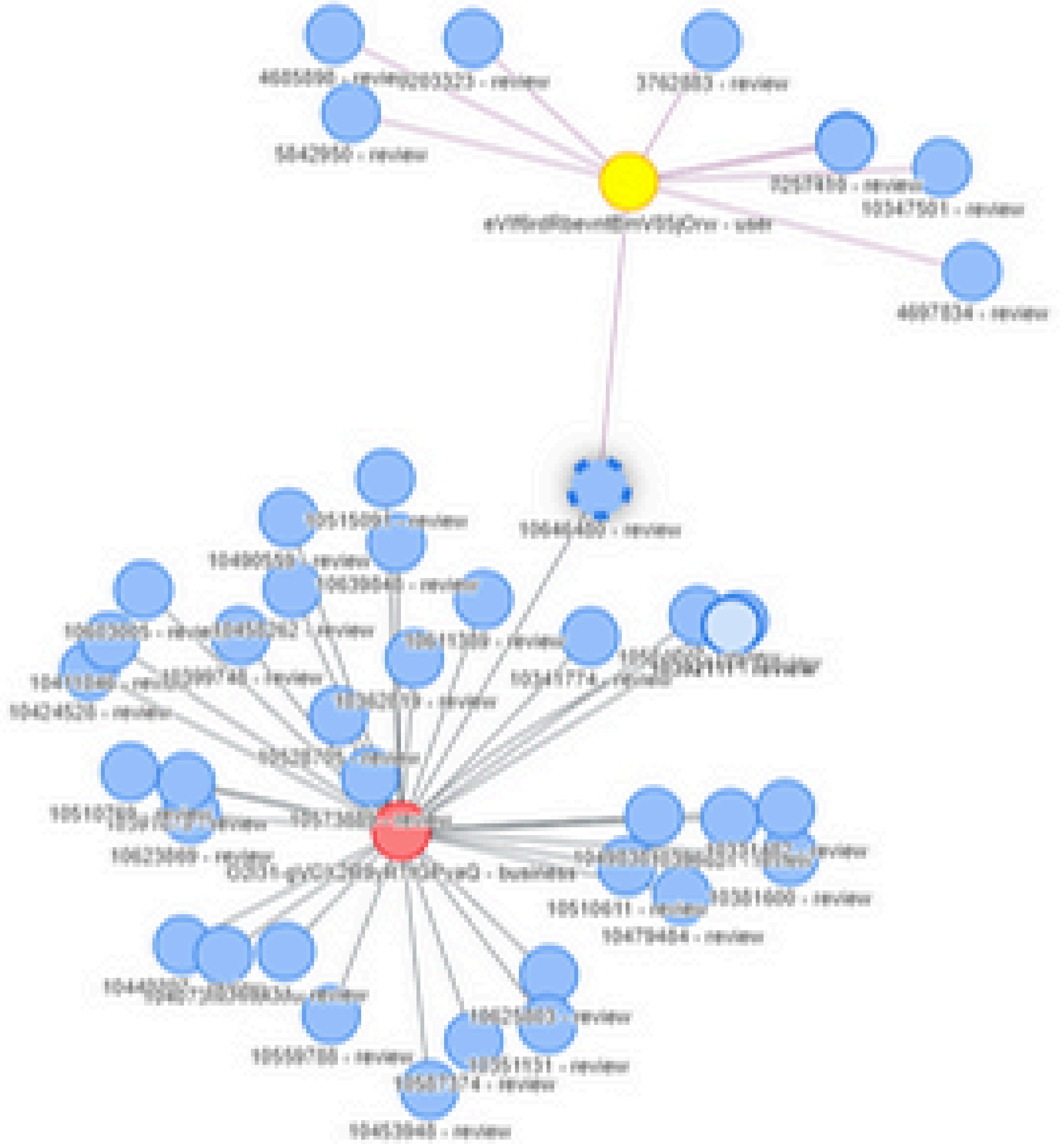
1. Businesses: Contains Business names, locations, categories, ratings etc
2. Users: Contains User’s name, review counts, and other relevant information.
3. Reviews: Reviews are the lifeblood of Yelp's platform .The review collection captures detailed feedback of ratings, textual content, and timestamps, enabling sentiment analysis and trend identification.
4. Tips: Tips provide very concise recommendations and insights shared by users about specific businesses.
5. Checkin: It contains only various check in times of businesses.

EDGE COLLECTIONS:

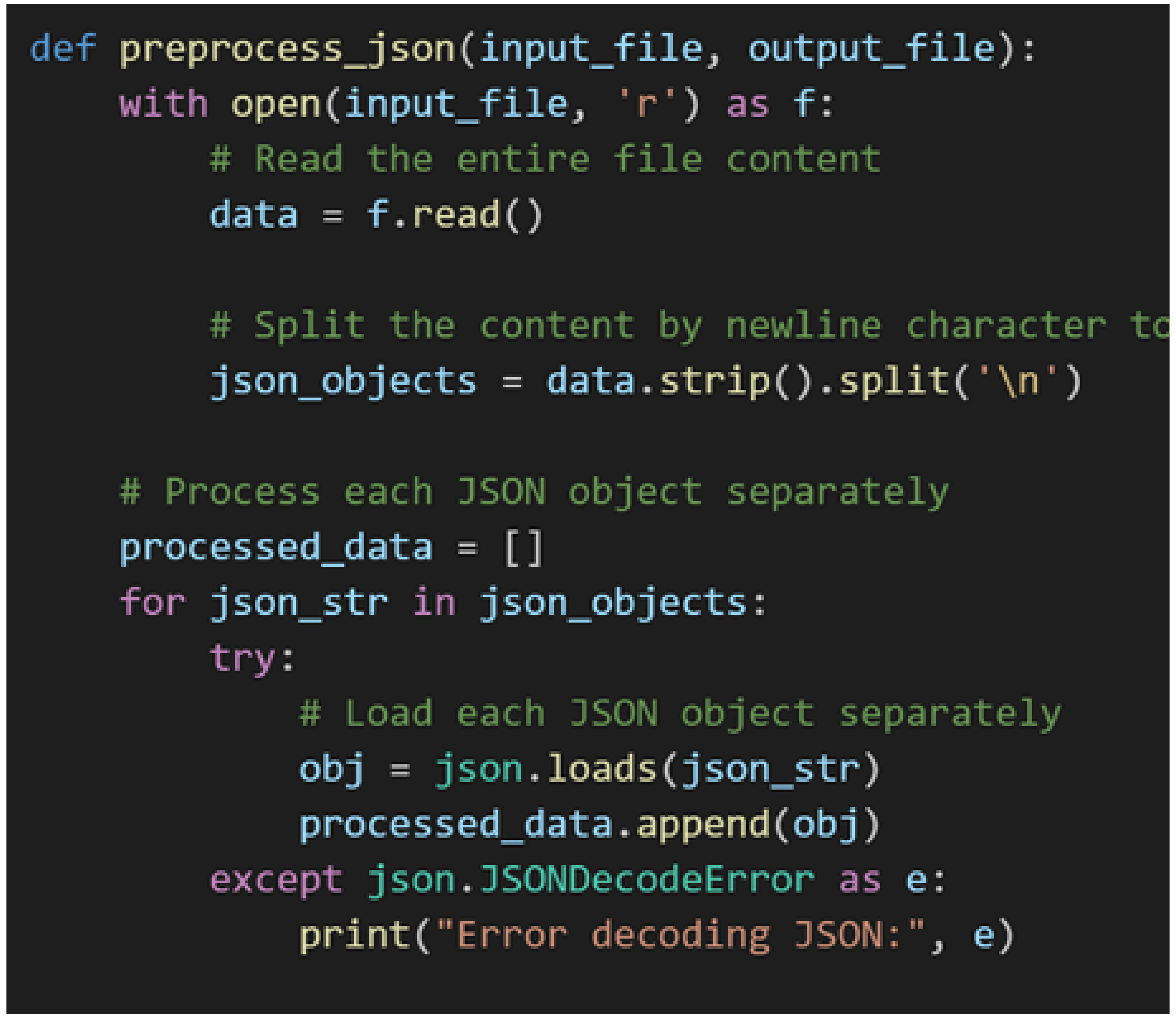
1. Business_Review: Connects businesses with reviews.
2. User_Review: Connects users with reviews.

WHY GRAPH DB IS USEFUL


kJ0KXovvmOAwhcK-Eb9L1w - user
AeoL31NZ0XkOmEFNH78bvw - user
4195332 - review
4320215 - review
1r8vCTpNk1kHx3n91CXSQQ - user
4305596 - review
tUI_Lq8dM3T-NnM-u3sz5g - business
4148541 - review
hc_yT7wWFCr4pe1tN2VvZg - user
4198418 - review
4179333 - review
4155253 - review
5kYef3iSVNoWtkyHJl9dZg - user
yKF3vSkrS-Knq0hZ5b4GpQ - user
ihgEMOxf13oL7ZepyO666w - user



PREPARING DATA & INGESTION

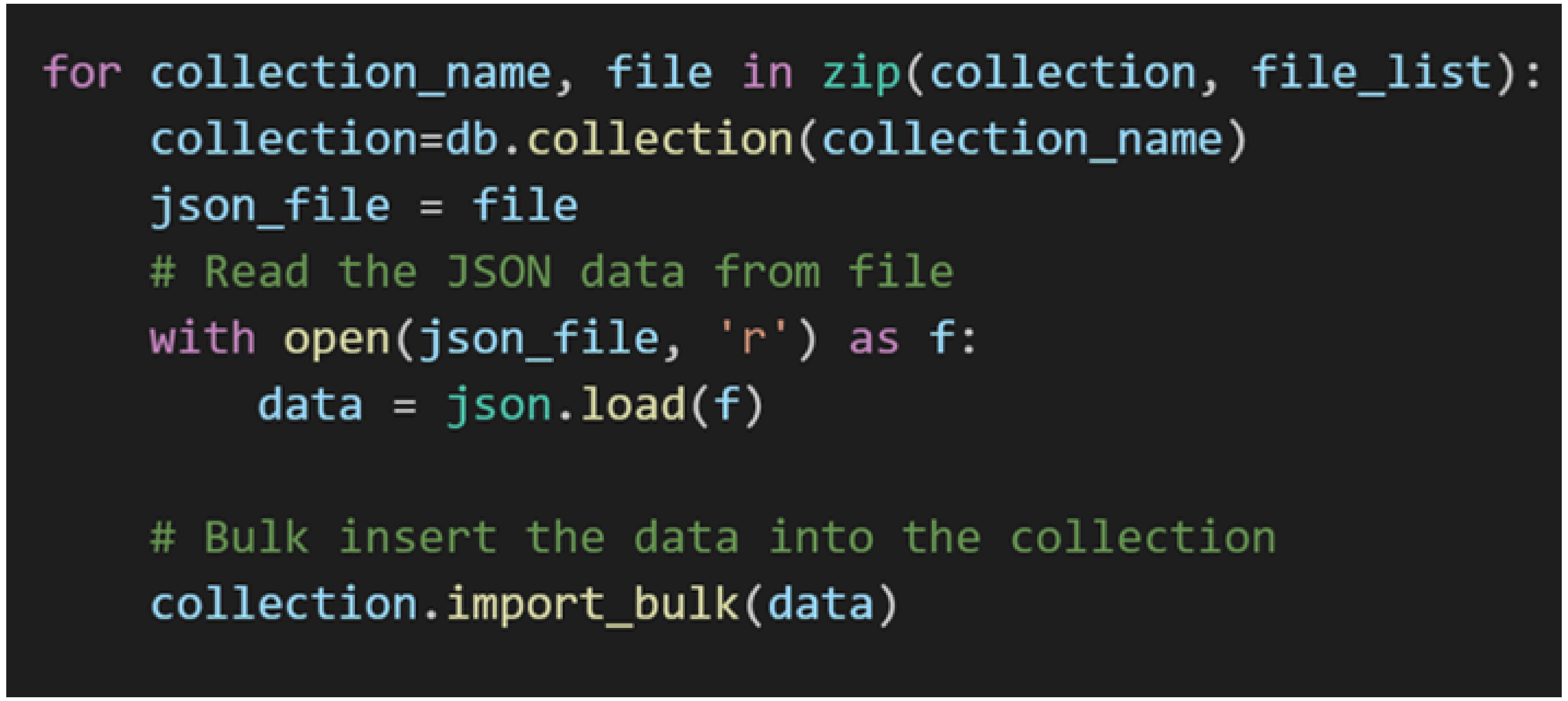


```python
def preprocess_json(input_file, output_file):
    with open(input_file, 'r') as f:
        # Read the entire file content
        data = f.read()

        # Split the content by newline character to
        json_objects = data.strip().split('\n')

    # Process each JSON object separately
    processed_data = []
    for json_str in json_objects:
        try:
            # Load each JSON object separately
            obj = json.loads(json_str)
            processed_data.append(obj)
        except json.JSONDecodeError as e:
            print("Error decoding JSON:", e)
```

Preparing Json Data For easy loading

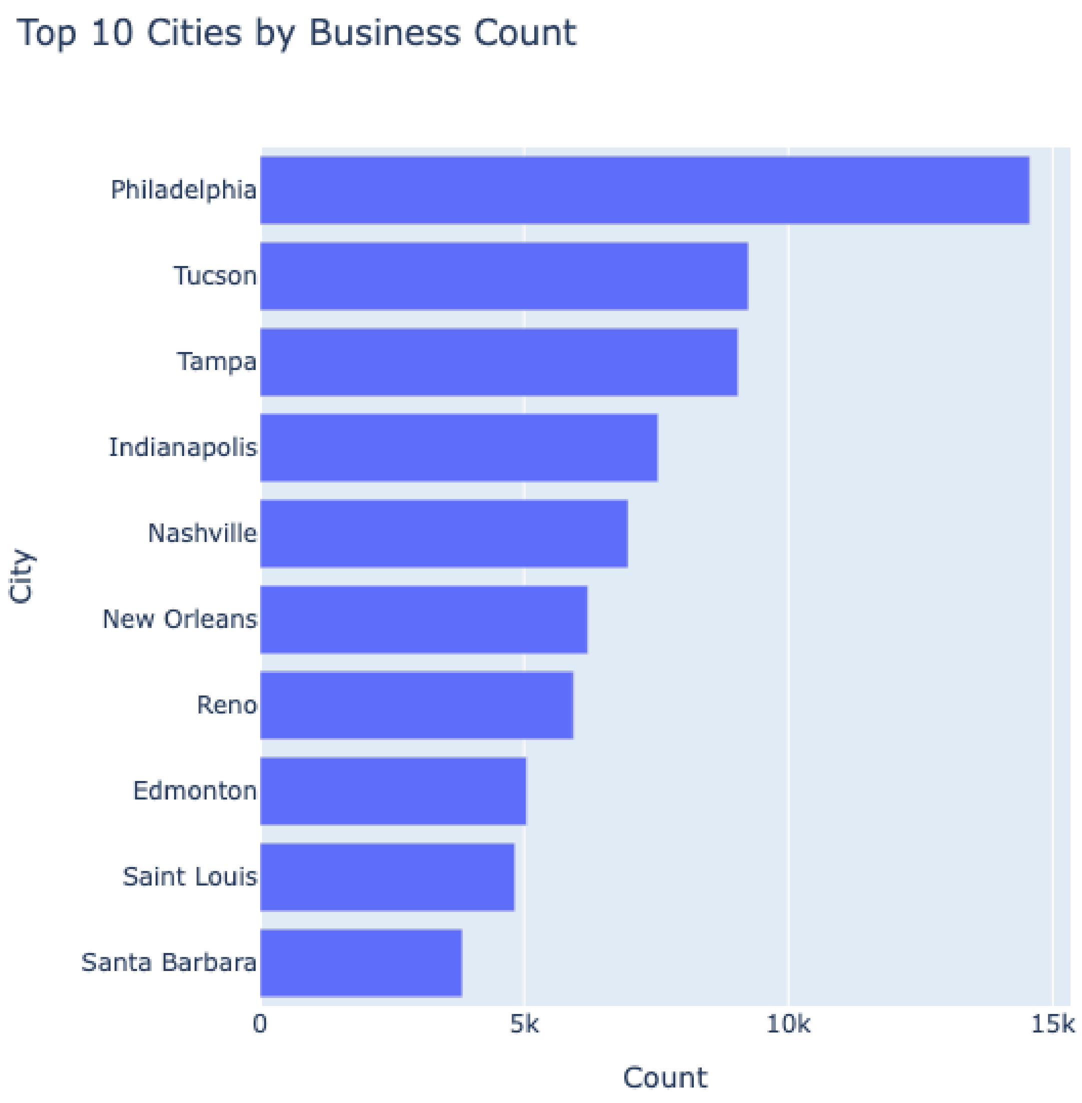


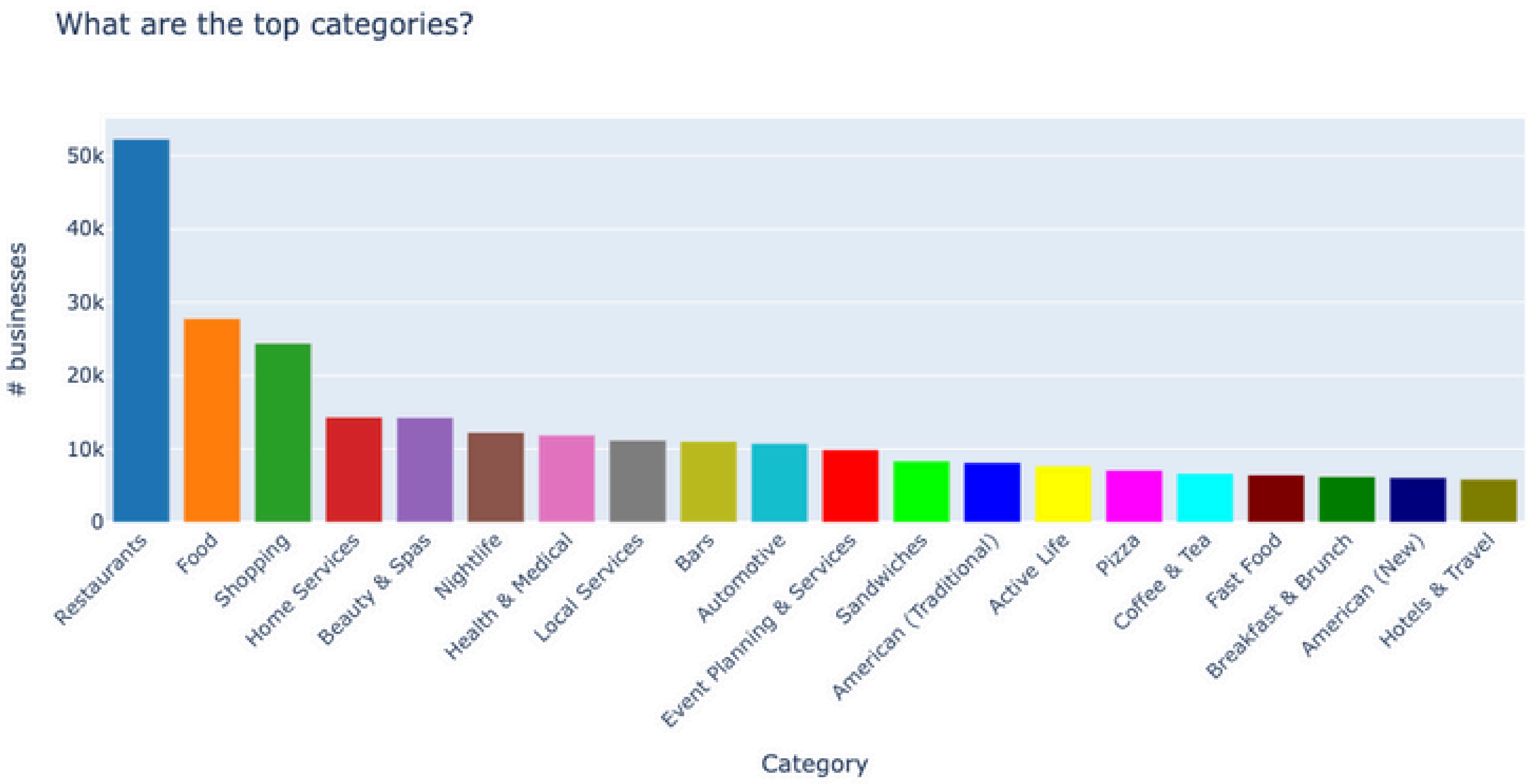
```python
for collection_name, file in zip(collection, file_list):
    collection=db.collection(collection_name)
    json_file = file
    # Read the JSON data from file
    with open(json_file, 'r') as f:
        data = json.load(f)

    # Bulk insert the data into the collection
    collection.import_bulk(data)
```

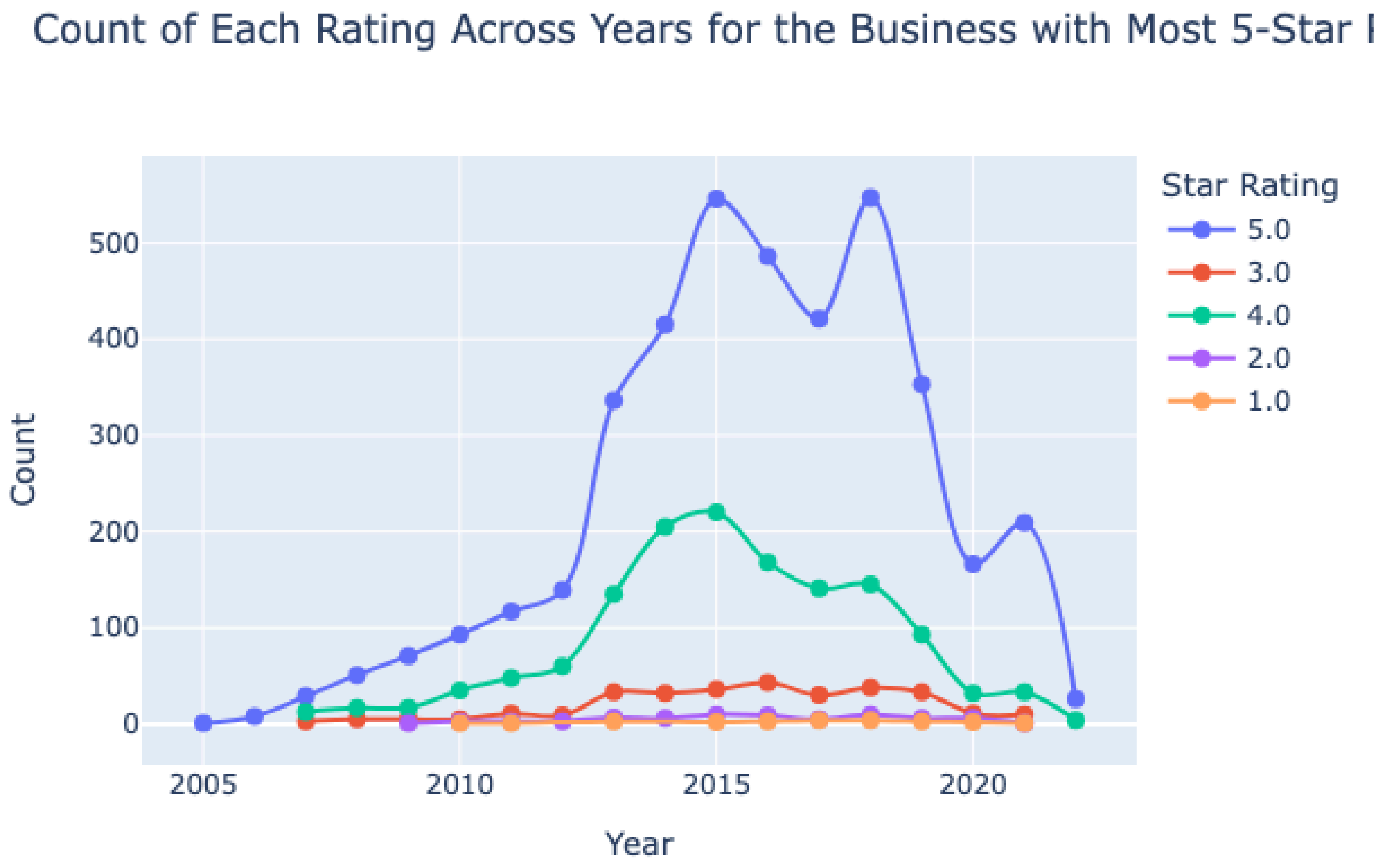
Loading Data into Collections

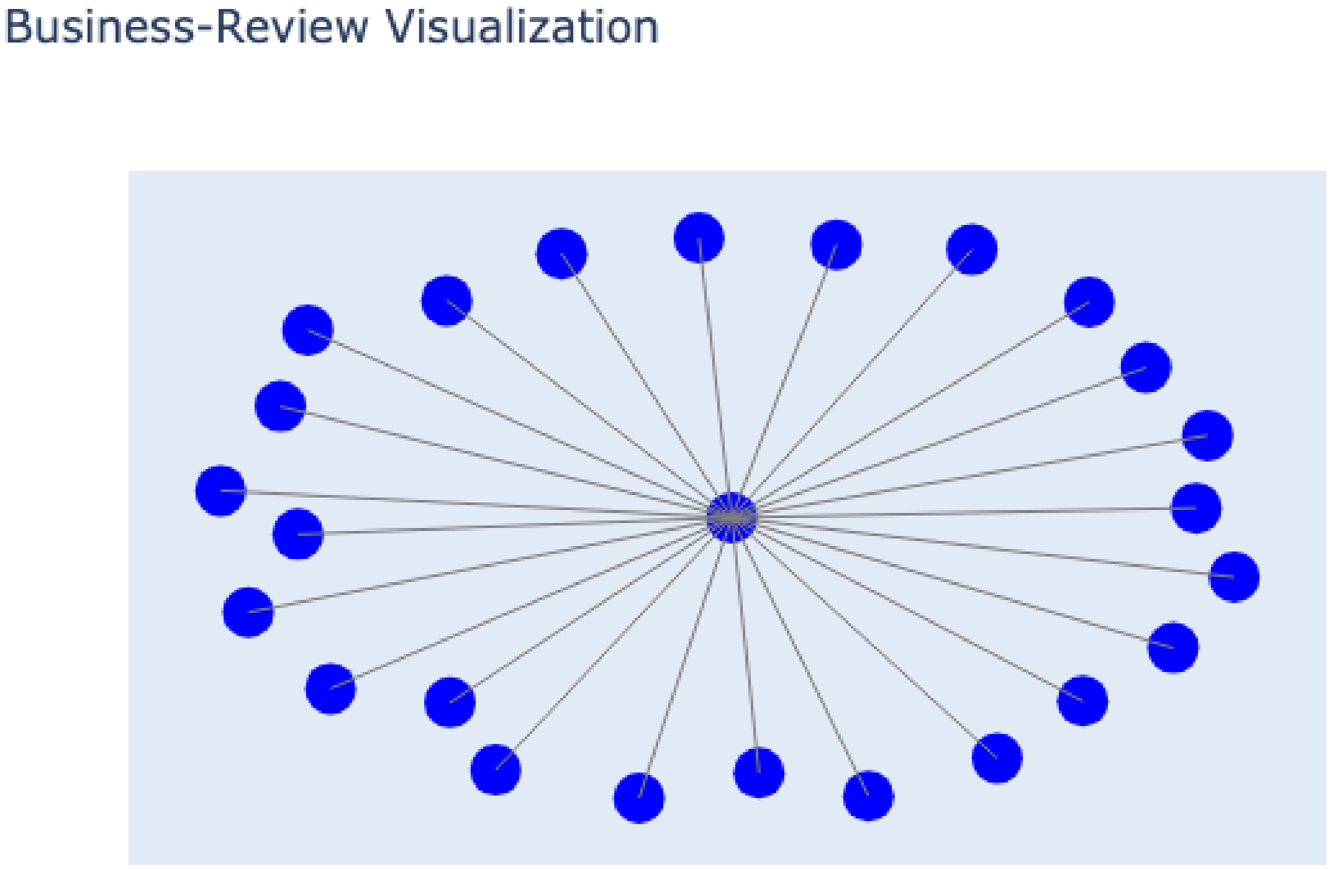
SAMPLE QUERY RESULTS





SAMPLE QUERY RESULTS





ISSUES AND CHALLENGES

- The challenges encountered during edge creation in the Yelp dataset's representation in Arango DB encompass issues related to , complex relationship patterns, schema design flexibility, data validation and edge cardinality.
- Loading data to the database due to large collections poses challenges in performance, resource utilization, data transfer speed, data integrity, transaction management, indexing overhead, and scalability

OVERCOMING THE CHALLENGES

- To overcome challenges in edge creation for the Yelp dataset in Arango DB, meticulously went through the collection data and made edge collections that made sense avoided unnecessary and redundant edges.
- To overcome challenges in loading data to the database due to large collections, optimize data loading processes through efficient resource management and loading data in batches

CONCLUSION

- We successfully preprocessed the Yelp dataset from JSON format to Arango database format, efficiently loading it into the database.
- Subsequently, we established essential edge relations within the collection, facilitating deeper insights into data relationships.
- Furthermore, leveraging visualization techniques and sentiment analysis, we gained valuable insights, enhancing our understanding of the dataset's nuances.
- Our comprehensive approach, from preprocessing to analysis, underscores our ability to unlock the dataset's potential for informed decision-making.

THANK YOU

Questions