ASSIGNMENT NO 6-Data Cleaning

Q1.

The extra-data json file provided contains a json document that describes the movies and some related information to it. It contains the box_office_currencyLabel which has 3 fields under it. The box_office_currencyLabel.xml:lang: contains the information in string format. The xml.lang has two values “en” and “type”. The box_office_currencyLabel.type gives the type of the currency. Box_office_currencyLabel.value gives the type of the of currency like United State Dollar, Hong Kong Dollar, Euro, Indian Rupees, Russian Ruble, 1 etc.

titleLable has type and value fields which describe the type of the title and the value of the title that is the name of the title. IMDb_ID.type describes the type which it belongs to, like literal.IMDb_ID.value gives the id of that particular title. titleLabel.value gives the name of the title and titleLabel.type gives its type like literal. Cost consists of 3 fields like datatype,type and value. Cost.value gives the cost for that particular title.

MPAA_film_ratingLabel contains xml:lang, its type and the value. MPAA_film_ratingLabel.value contains the type of rating like “PG-13”, ”R”, ”PG”, ”G” . It gives us the rating related to each title.

distributorLabel contains xml:lang,type,value. The distributorLabel.value gives the name of the distributor for the related title.

Q2.

First, loading the extra-data.json file into mongodb database imdbNew for seeing the structure of the document.

```
C:\Users\Revaa>mongoimport --db imdbNew --collection newData --file
C:\Users\Revaa\Desktop\jsonfiles\extra-data.json
```

Exporting the newData collection from mongoDb that contains the extra-data.json data in the form of csv file. Only the fields that are required like IMDb_ID.value, MPAA_film_ratingLabel.value,box_office.value,box_office_currencyLabel.value,cost.value,distributorLabel.value, titleLabel.value are selected while converting into extra-data.csv file.
The csv file is exported using the following command:

C:\Users\Revaa>mongoexport --host localhost:27017 --db imdbNew --collection newData --csv --out C:\Users\Revaa\Desktop\jsonfiles\extra-data.csv --fields IMDb_ID.value,MPAA_film_ratingLabel.value,box_office.value,box_office_currencyLabel.value,cost.value,distributorLabel.value,titleLabel.value

After the csv file is exported, the pre-processing part is done in postgresql.First, a table is created where all the data from the csv file is loaded into it.

Creating a temporary table

```
create table extraData(_id varchar,
                       MPAA_film_ratingLabel varchar,
                       box_office float,
                       box_office_currencyLabel varchar,
                       cost float,
                       distributorLabel varchar,
                       titleLabel text);
```

Copying data from csv file

```
copy extraData from 'C:/Users/Revaa/Desktop/jsonfiles/extra-datanew.csv' with(format csv, header true, delimiter ',',null '', force_null(MPAA_film_ratingLabel, box_office,box_office_currencyLabel, cost, distributorLabel,titleLabel));
```

The _ID contains the data in varchar type and it needs to be converted into integer type.

Removing the unwanted characters and converting its type to integer:

```
update extraData
set _id=replace(_id,'tt','');

update extraData
set _id=replace(_id,'nm','');

alter table extraDataTemp
alter column _id type integer
USING (_id::INTEGER);
```

The box office revenue needs to be only in United States Dollar. So selecting only the records that have the currency as United States Dollar and removing the null values from the records.

Creating a table that stores the records with only US dollars.

```
create table extraDataTemp(_id varchar,
                       MPAA_film_ratingLabel varchar,
                       box_office float,
                       box_office_currencyLabel varchar,
                       cost float,
```

```
                                    distributorLabel varchar,
                                    titleLabel text);
```

Removing null values in _id and selecting the values where box office currency is United States dollar

```
insert into extraDataTemp
select * from extraData where _id is not null and box_office_currencyLabel='United States
dollar';
```

It is found that there are many records which contain duplicate values.

Creating another table extraDataTe which contains the values with no duplicates.

```
create table extraDataTe(_id integer,
                                    MPAA_film_ratingLabel varchar,
                                    box_office float,
                                    box_office_currencyLabel varchar,
                                    cost float,
                                    distributorLabel varchar,
                                    titleLabel text);

insert into extraDataTe
select distinct on (_id) _id,
                                    MPAA_film_ratingLabel,
                                    box_office ,
                                    box_office_currencyLabel,
                                    cost ,
                                    distributorLabel,
                                    titleLabel
                                    from
                                    extraDataTemp;


update extraDataTe
set titleLabel=replace(titleLabel,'"','');
```

All the duplicate values, null values were removed and the json file obtained contains only the required information which is used for combining the two collections.
After the pre-processing on the data is completed the data is converted into a json file and then loaded into mongodb for merging it with the Movies collection.

Creating a json file

```
COPY (SELECT json_strip_nulls(row_to_json(r))FROM( SELECT _id, MPAA_film_ratingLabel,
box_office,box_office_currencyLabel, cost, distributorLabel, titleLabel from extraDataTe)r)
TO 'C:\Users\Revaa\Desktop\jsonfiles\extraDataTe.json' with (FORMAT TEXT, HEADER false);
```

Importing the json file into the imdbNew database in mongodb

C:\Users\Revaa>mongoimport --db imdbNew --collection extraDataTe --file C:\Users\Revaa\Desktop\jsonfiles\extraDataTe.json

Merging the collections on the basis of _id:

```
db.extraDataTe.aggregate({
  $merge:{into:"Movies",on:"_id"}})
```

After merging the collection the total successful updates were found to be 181.

Q3.

Assuming that the documents in the file do not contain IMDB ids, the matching process can be performed on the basis of titles. The two collections can be merged on the basis of matching titles. We cannot consider other fields like box_office.value, box_office_currencyLabel, cost, MPAA_film_ratingLabel for performing the matching process as they are not present in the Movies collection. To perform the matching process, the field on which the matching is to be performed needs to be present in both the collections. In this scenario, the only common field on which the matching can be performed is the titles. Both the collections have titles for each id.

This can be done by converting the json file into csv file by considering only the required columns. Here we are not considering the Imdb ids as they are not required.

The procedure here is same as question 2

First, exporting the collection as a csv file with only the required columns.

C:\Users\Revaa>mongoexport --host localhost:27017 --db imdbNew --collection newData --csv --out C:\Users\Revaa\Desktop\jsonfiles\extra-dataq3.csv --fields MPAA_film_ratingLabel.value,box_office.value,box_office_currencyLabel.value,cost.value,distributorLabel.value,titleLabel.value

Creating a table to copy the records from csv file

```
create table extraDataq3(
                                MPAA_film_ratingLabel varchar,
                                box_office float,
                                box_office_currencyLabel varchar,
                                cost float,
                                distributorLabel varchar,
                                titleLabel text);
```

Copying the data from csv file to the table

```
copy extraDataq3 from 'C:/Users/Revaa/Desktop/jsonfiles/extra-dataq3.csv'
                    with(format csv, header true, delimiter ',',null '',
                    force_null(MPAA_film_ratingLabel,box_office,
                            box_office_currencyLabel, cost,
                            distributorLabel,titleLabel));
```

Creating a new table to store the records with no duplicates

```
create table extraDataq3Te(
                    MPAA_film_ratingLabel varchar,
                     box_office float,
                     box_office_currencyLabel varchar,
                     cost float,
                     distributorLabel varchar,
                     titleLabel text);
```

Inserting the records into the new table

```
insert into extraDataq3Te
select distinct on (titleLabel)
                    MPAA_film_ratingLabel ,
                     box_office ,
                     box_office_currencyLabel,
                     cost ,
                     distributorLabel,
                     titleLabel
                    from
                    extraDataq3;
```

Creating a json file after completing the pre-processing in the data

```
COPY (SELECT json_strip_nulls(row_to_json(r))FROM(
SELECT MPAA_film_ratingLabel,box_office,box_office_currencyLabel, cost, distributorLabel,
titleLabel as _id from extraDataq3Te)r)
TO 'C:\Users\Revaa\Desktop\jsonfiles\extraDataq3.json' with (FORMAT TEXT, HEADER false);
```

Creating a table for Movies data with required columns

```
create table movieq3(actors varchar,
                    avgrating float,
                     directors varchar,
                     genres varchar,
                     numvotes integer,
                     originaltitle text,
                     producers varchar,
                     runtime integer,
```

```
                                        startyear integer,
                                       title text,
                                        type varchar,
                                        writers varchar
                                        );
```

Copying the data into the created table

```
copy movieq3 from 'C:\Users\Revaa\Desktop\jsonfiles\Movies.csv' with(format csv, header true,
delimiter ',',null '',force_null(actors,avgrating,directors,genres,numvotes,originaltitle, producers,
runtime,startyear,title ,type ,writers ));
```

Creating another table for movies to store the records without duplicates

```
create table movieq3Te(actors varchar,
                                       avgrating float,
                                        directors varchar,
                                        genres varchar,
                                        numvotes integer,
                                        originaltitle text,
                                        producers varchar,
                                        runtime integer,
                                        startyear integer,
                                       title text,
                                        type varchar,
                                        writers varchar
                                        );
```

Inserting the records into the new table

```
insert into movieq3Te
select distinct on (title)
                    actors,avgrating,directors,genres,numvotes,originaltitle,producers,runtime,
startyear,title ,type , writers from movieq3;

update movieq3Te
set actors=replace(actors,"","");

update movieq3Te
set genres=replace(genres,"","");
```

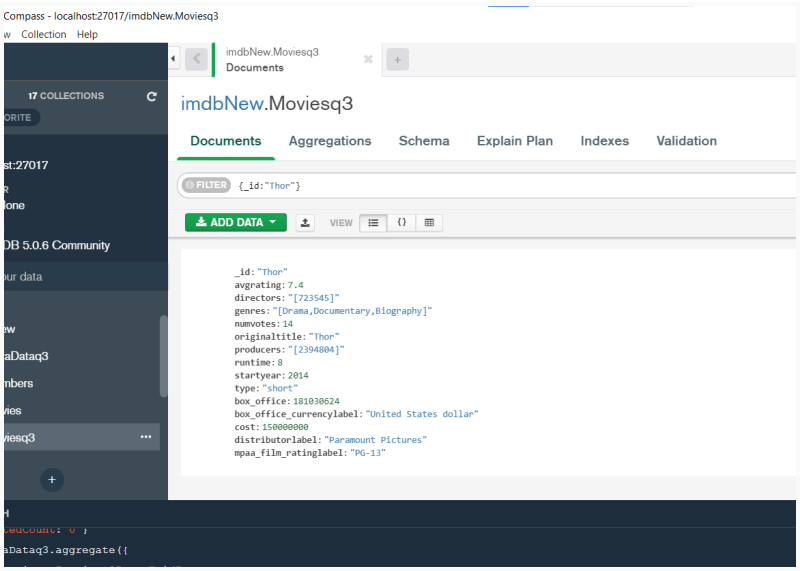
Creating a json file after completing the pre-processing

```
COPY (SELECT json_strip_nulls(row_to_json(r))FROM(
SELECT actors,avgrating,directors, genres, numvotes,originaltitle,producers,runtime,
startyear, title as _id ,type , writers from movieq3Te)r)
TO 'C:\Users\Revaa\Desktop\jsonfiles\Moviesq3.json' with (FORMAT TEXT, HEADER false);
```

Merging the two collections in mongodb to get the required output

```
db.ExtraDataq3.aggregate({
  $merge:{into:"Moviesq3",on:"_id"}}
)
```

The total successful updates were found to be 173

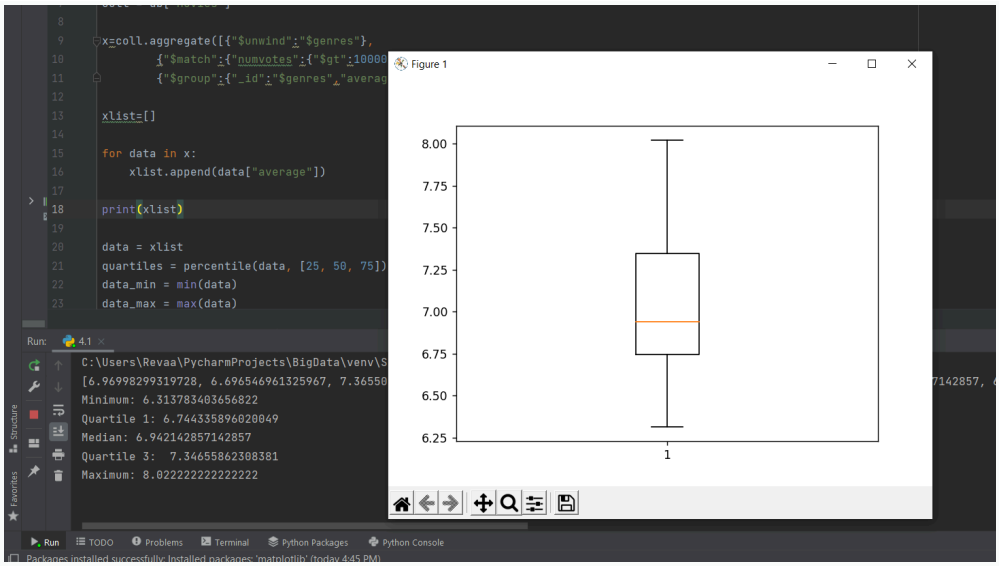


Q4.

4.1 For each genre, a five-number summary of the average ratings of movies with more than 10K votes.

```
from pymongo import MongoClient
from numpy import percentile
import matplotlib.pyplot as plt

client = MongoClient('localhost',27017)
db = client['imdbNew']
coll = db['Movies']

x=coll.aggregate([{"$unwind":"$genres"},
      {"$match":{"numvotes":{"$gt":10000}}},
      {"$group":{"_id":"$genres","average":{"$avg":"$avgrating"}}}])

xlist=[]

for data in x:
```

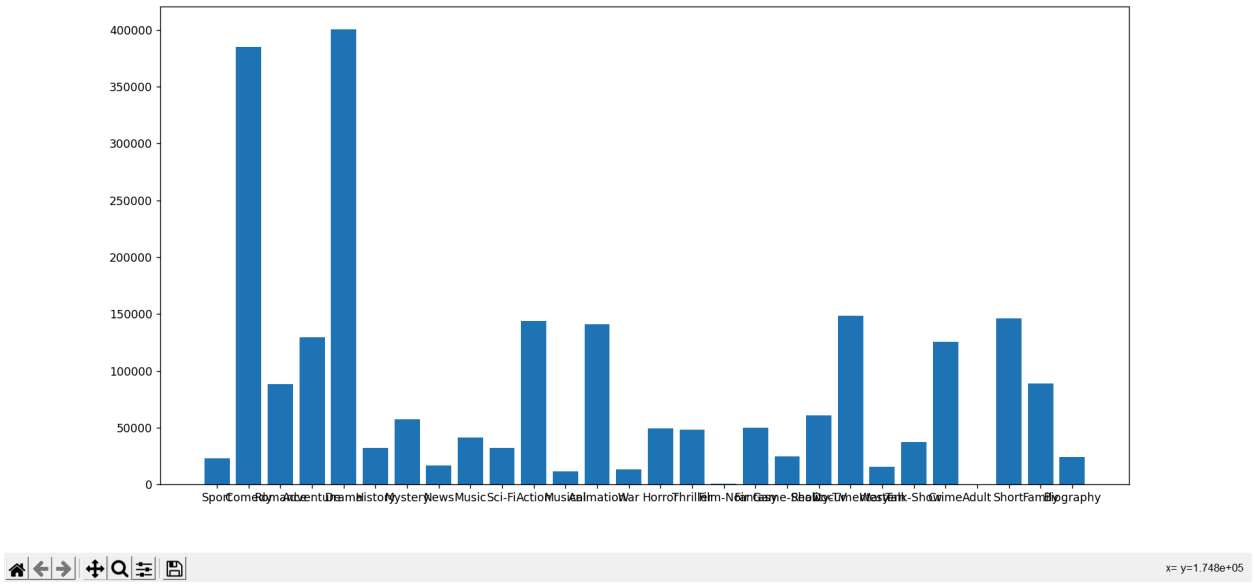
```
  xlist.append(data["average"])

print(xlist)

data = xlist
quartiles = percentile(data, [25, 50, 75])
data_min = min(data)
data_max = max(data)

print('Minimum:',data_min)
print('Quartile 1:',quartiles[0])
print('Median:',quartiles[1])
print('Quartile 3: ', quartiles[2])
print('Maximum:',data_max)

plt.boxplot(xlist)
plt.show()
```



4.2 Average number of actors per movie by genre as a bar chart for all movies with any actors (i.e. skip documents with no “actors” field).

```
from pymongo import MongoClient
import matplotlib.pyplot as plt

client = MongoClient('localhost', 27017)
db = client['imdbNew']
coll = db['Movies']

x = coll.aggregate([{"$unwind": "$genres"},
                    {"$group": {"_id": "$genres", "count": {"$sum": 1}}},
                    {"$group": {"_id": "$_id", "average": {"$avg": "$count"}}}
```

```
                ])

xlist = []
ylist = []

for data in x:
  xlist.append(data["_id"])
  ylist.append(data["average"])

print(xlist)
print(ylist)
plt.bar(xlist, ylist)
plt.show()
```



4.3 Number of movies produced each year (startYear) as a time series plot

```
from pymongo import MongoClient
import matplotlib.pyplot as plt

client = MongoClient('localhost', 27017)
db = client['imdbNew']
coll = db['Movies']
x = coll.aggregate([{"$match": {"producers": {"$exists": True}}},
                    {"$group": {"_id": "$startyear", "count": {"$count": {}}}}
                    ])
xlist = []
ylist = []
```

```
for data in x:
  xlist.append(data["_id"])
  ylist.append(data["count"])

print(xlist)
print(ylist)

plt.plot(xlist, ylist)
plt.show()
```