# Lab Assignment 4

# Report On

# CIS 612 Big Data Processing

# November 4, 2022

1. **Part One:** Building a Knowledge Base in Inverted Index for Big Data Applications such as web Page Categorization or Google Search Engine.

   We build inverted index in two MySQL tables.

   - First Level Dictionary Table(Term, DocFreq, CollectionFreq)
   - Second Level Posting Table(Term, Doc_Id, TermFreq)

   A Python script created in the root folder (part_1.py) is run to generate predefined tables as shown in the figure below.

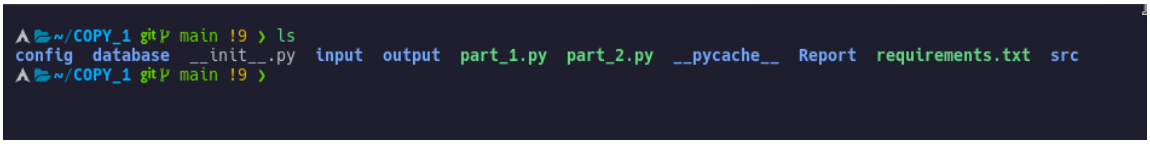



Fig 1.1: Files required.

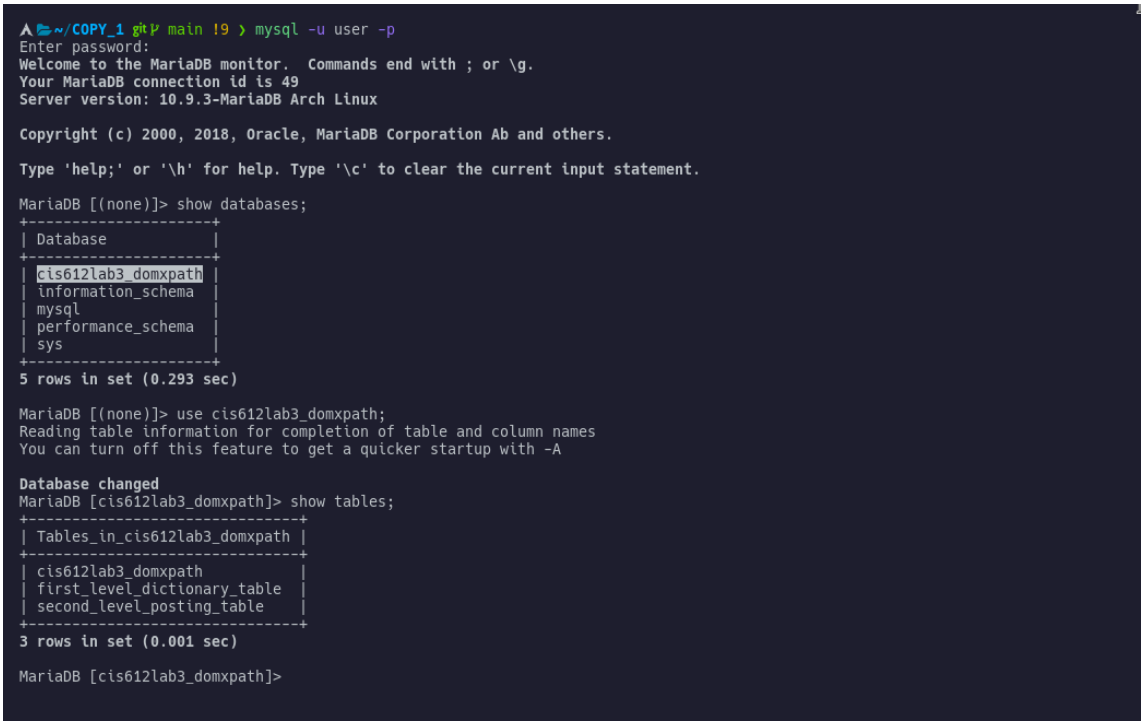



Fig 1.2: Login MySQL, Database and Tables.

```
MariaDB [cis612lab3_domxpath]> show tables;
+-------------------------------+
| Tables_in_cis612lab3_domxpath |
+-------------------------------+
| cis612lab3_domxpath           |
| first_level_dictionary_table  |
| second_level_posting_table    |
+-------------------------------+
3 rows in set (0.000 sec)

MariaDB [cis612lab3_domxpath]> describe first_level_dictionary_table;
+---------------------+-------------+------+-----+---------+----------------+
| Field               | Type        | Null | Key | Default | Extra          |
+---------------------+-------------+------+-----+---------+----------------+
| id                  | int(11)     | NO   | PRI | NULL    | auto_increment |
| term                | varchar(255)| NO   |     | NULL    |                |
| documentFrequency   | varchar(255)| NO   |     | NULL    |                |
| collectionFrequency | int(11)     | YES  |     | NULL    |                |
+---------------------+-------------+------+-----+---------+----------------+
4 rows in set (0.293 sec)

MariaDB [cis612lab3_domxpath]> select id,term,documentFrequency,collectionFrequency from first_level_dictionary_table limit 10;
+----+--------------+---------------------------------------+---------------------+
| id | term         | documentFrequency                     | collectionFrequency |
+----+--------------+---------------------------------------+---------------------+
|  1 | fellowcitizen | Theodore Roosevelt  December 8, 1908 |                  72 |
|  2 | of           | George W. Bush  January 28th, 2008    |                 217 |
|  3 | the          | George W. Bush  January 28th, 2008    |                 217 |
|  4 | senat        | George W. Bush  January 28th, 2008    |                 183 |
|  5 | and          | George W. Bush  January 28th, 2008    |                 217 |
|  6 | hous         | George W. Bush  January 28th, 2008    |                 201 |
|  7 | repres       | George W. Bush  January 23rd, 2007    |                 195 |
|  8 | i            | George W. Bush  January 28th, 2008    |                 217 |
|  9 | embrac       | George W. Bush  January 29, 2002      |                  86 |
| 10 | with         | George W. Bush  January 28th, 2008    |                 217 |
+----+--------------+---------------------------------------+---------------------+
10 rows in set (0.089 sec)

MariaDB [cis612lab3_domxpath]>
```

Fig 1.3: First Level dictionary table

```
MariaDB [cis612lab3_domxpath]> show tables;
+-------------------------------+
| Tables_in_cis612lab3_domxpath |
+-------------------------------+
| cis612lab3_domxpath           |
| first_level_dictionary_table  |
| second_level_posting_table    |
+-------------------------------+
3 rows in set (0.000 sec)

MariaDB [cis612lab3_domxpath]> describe second_level_posting_table;
+---------------+-------------+------+-----+---------+----------------+
| Field         | Type        | Null | Key | Default | Extra          |
+---------------+-------------+------+-----+---------+----------------+
| id            | int(11)     | NO   | PRI | NULL    | auto_increment |
| term          | varchar(255)| NO   |     | NULL    |                |
| documentId    | varchar(255)| NO   |     | NULL    |                |
| termFrequency | int(11)     | YES  |     | NULL    |                |
+---------------+-------------+------+-----+---------+----------------+
4 rows in set (0.001 sec)

MariaDB [cis612lab3_domxpath]> select id,term,documentId,termFrequency from second_level_posting_table limit 10;
+----+--------------+------------+---------------+
| id | term         | documentId | termFrequency |
+----+--------------+------------+---------------+
|  1 | fellowcitizen | 133        |            72 |
|  2 | of           | 78         |           217 |
|  3 | the          | 78         |           217 |
|  4 | senat        | 78         |           183 |
|  5 | and          | 78         |           217 |
|  6 | hous         | 78         |           201 |
|  7 | repres       | 119        |           195 |
|  8 | i            | 78         |           217 |
|  9 | embrac       | 15         |            86 |
| 10 | with         | 78         |           217 |
+----+--------------+------------+---------------+
10 rows in set (0.000 sec)

MariaDB [cis612lab3_domxpath]>
```

Fig 1.4: Posting File in SQL table

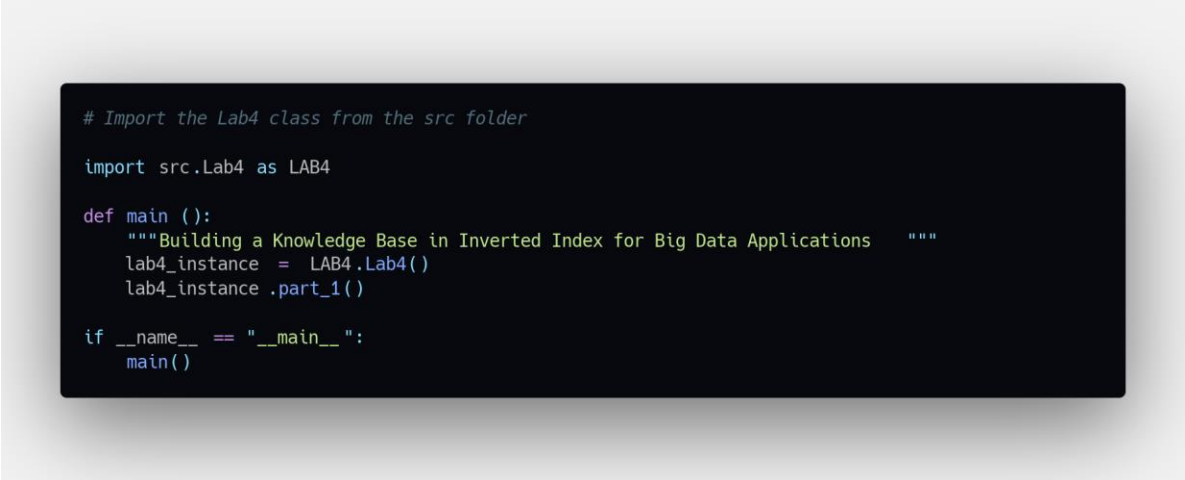
Documents are preprocessed in a better way to enhance a word count in a systematic way,

Below are the NLP methods used in the project :-

- Text punctuation
- Tokenization

- Convert numbers to words
- Suffix trimming
- Remove stop words
- Lowercasing

The python script **part_1.py** contains the code below.



```
# Import the Lab4 class from the src folder

import src.Lab4 as LAB4

def main ():
    """Building a Knowledge Base in Inverted Index for Big Data Applications    """
    lab4_instance  =  LAB4.Lab4()
    lab4_instance .part_1()

if __name__  == "__main__":
    main()
```

Fig 1.5 Python script part_1.py

## 2. **Part Two:** Building Document Vectors for Web Page Content Analysis.

We are cheking for text similarities between documents in cosine similarity metric.

Using cosine similarity, we consider matrix vectors use. Part Two mostly consistes of document vectorization and cosine similarity computatios.

We use the TfidfVectorizer() class module to perform document vectorization

TfidfVectorizer() - Transform a count matrix to a normalized tf or tf-idf representation.

Tf means term-frequency while tf-idf means term-frequency times inverse

document-frequency. This is a common term weighting scheme in information retrieval, that has also found good use in document classification.

This is the vectorize function.

```
def vectorize_document (self, processed_text_document ):
    """ Vectorize the document  """

    vectorizer  = TfidfVectorizer ()
    for doc_name , doc_content  in processed_text_document .items():
        temp_cs_dict  = {}
        train_corpus  = [" " . join(doc_content )]
        try:
            X  = vectorizer .fit_transform (train_corpus )
            X  = X.toarray()
            for doc_name_2 , doc_content_2  in processed_text_document .items():
                temp_cs_dict [doc_name_2 ] = self.find_cosine_similarity (X, vectorizer .transform([" ".join(doc_content_2 )]).toarray())
        except ValueError :
            continue
    self.cosine_similarity_result [doc_name ] = temp_cs_dict
```

Fig 1.6: Vectorization

Next step is to find similarity between text, we use the dot method of numpy computation functions.

```
def find_cosine_similarity (self, a, b):
    """ Find the cosine similarity  """
    a = numpy.array(a)
    b = numpy.array(b)
    dot_product  = numpy.dot(a, b)
    magnitude  = numpy.linalg.norm(a) * numpy.linalg.norm(b)
    if not magnitude :
        return 0
    return dot_product  / magnitude
```

Fig 1.7: Numpy dot method computation

Below is the code for python script **part_2.py**

```python
# Import the Lab4 class from the src folder

import src.Lab4 as LAB4

def main ():
    """Building Document Vectors for Web Page Content Analysis   """
    lab4_instance  =  LAB4.Lab4()
    lab4_instance .part_2()

if __name__  == "__main__":
    main()
```

Fig 1.8: Python script part_2.py

**Running and configuration**

Requirements.

Python3, pip modules in requirements.txt, MySQL server

1. The server credentials are located in .env file which is loaded and assigned to necessary variables during connection.

Edit this file to suit your local configuration.

2. Open your terminal and start your MySQL server, e.g. In Linux you would use this command

sudo systemctl start mysql.service

Note: Service names may be different, please confirm with your installation.

3. Unzip your project folder and change into the COPY directory.
4. Now we need pip modules installed

   pip3 install –r requirements.txt

5. Now run the script part_1.py and part_2.py respectively

   python3 part_1.py and python3 part_2.py

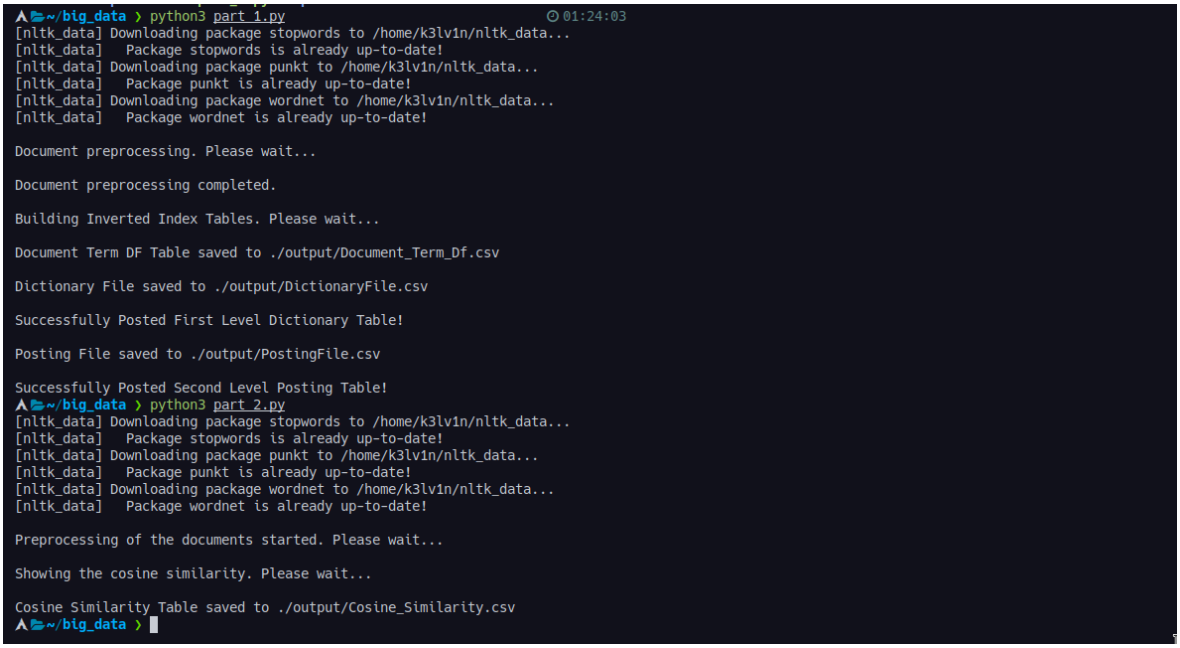



Fig 1.9: Output part_1.py and part_2.py