QUBIT DOCUMENTATION

**Problem Statement :**

Objective: Create a workflow that extracts company data from an SQL database,

enriches the data using the LinkedIn Bulk Data Scraper API, and stores the enriched

data in a new table in the database.

**File Structure :**

* **Qubit**
  + **DatabaseQueries.py**
  + **execution.py**
  + **getAPIdata.py**
  + **queries.py**
  + **serializers.py**
  + **Output\_test.json**
  + **Requirements.txt**
  + **.env**

**DatabaseQueries.py :**

Having a class named DatabaseManager, the following operations were carried out.

1. **Initialize**
   1. A connection (named **conn**) to the database with it’s credentials.
   2. A cursor (named **cur**) that can do operation on the database.
2. **CreateTables**
   1. Taking the query info from queries.py file and executing it.
   2. No arguments needed.
3. **Insert (tableName, values)**
   1. Takes **tableName, values** as arguments.
   2. tableName : string value
   3. values : list of tuples of the data need to be inserted
   4. Based on tableName given we take the corresponding query from queries.py file and using **executemany function**, we can insert many values by a single instruction.
4. **delete\_table(tableName)**
   1. Takes **tableName** as argument.
   2. tableName: string value.
   3. Executing the DROP function, giving the tableName to it.
5. **showTabel(tableName)**
   1. Takes **tableName** as argument.
   2. tableName : string value.
   3. Based on tableName given we take the corresponding query from queries.py file and using **execute function** and **fetchall()** we get the rows of the table data.
   4. Returns rows of the table.
6. **Close**
   1. No arguments needed.
   2. Closes the opened connection(conn) and cursor(curr) in initialization.

**getAPIdata.py :**

Using requests, a python library, we can call a POST request to API, provided the URL.

1. As credentials are in the .env file, using dotenv library we can retrieve them.
2. Takes **payload** as argument.
3. payload : dict
   1. format : {“links” : [list of LinkedIn links in string format] }
4. initializing headers : headers = {

        'x-rapidapi-key': os.getenv("RAPID\_API\_KEY"),

        'x-rapidapi-host': os.getenv("RAPID\_API\_HOST"),

        'Content-Type': "application/json",

        'x-rapidapi-user': os.getenv("RAPID\_API\_USER"),

        }

1. data = requests.post(url, json=payload, headers=headers)
2. Returning the data obtained in JSON format

**Serializers.py :**

Built with 2 functions, one for deserializing the data obtained from the API, other is for serializing the data obtained from the database operations and converting it into JSON format.

1. deserializer(data)
   1. Takes data as argument.
   2. data : dict 🡪 contains all the necessary info about the Company.
   3. Function filters the required data for inserting in database, and assigns them to corresponding lists.
   4. Lists 🡪 enrichedData, locations\_list, similar\_organizations\_list, affiliate\_organizations\_list
   5. Returns Lists
2. serializer(tableName, data)
   1. Takes data, tableName as argument.
   2. tableName : string value
   3. data : list 🡪 list of tuples of the data present in database
   4. Returns JSON data

**queries.py**

Contains different queries used in insert, createTables, showTable methods.

**execution.py**

Contains all imports from different files and executing each of the function as required.

**Output\_test.json**

The output printed on terminal can be copied and pasted here to view the data clearly in JSON format.

**requirements.txt**

Contains all the requirements that needs to be installed in order to run the programs. The below the requirements needs to be installed to run the project.

1. certifi==2024.7.4
2. charset-normalizer==3.3.2
3. idna==3.7
4. psycopg2==2.9.9
5. python-dotenv==1.0.1
6. requests==2.32.3
7. urllib3==2.2.2

**.env**

Contains all the confidential information of the project. Here in this project it contains :

1. DB\_NAME
2. DB\_USER
3. DB\_PASSWORD
4. DB\_HOST
5. DB\_PORT
6. RAPID\_API\_URL
7. RAPID\_API\_KEY
8. RAPID\_API\_USER
9. RAPID\_API\_HOST

**Format of values list for inserting data :**

1. CompanyData Table
   1. Format : [(company\_name, company\_linkedin\_url), … ]
2. EnrichedCompanyData Table
   1. Format : [(company\_name, company\_url, company\_id), … ]
3. similarOrganizations Table
   1. Format : [(related\_company\_name, similar\_organization\_name), … ]
4. affiliatedOrganizations Table
   1. Format : [(related\_company\_name, affiliatedOrganization\_name), … ]
5. Locations Table
   1. Format : [(company\_name, location\_country, location\_city, location\_pin, location\_headquarters), … ]

**Operations Allowed :**

1. Database Operations
   1. createTables()
   2. insert(tableName, values)
   3. delete\_table(tableName)
   4. showTable(tableName)
   5. close()
2. deserializer(api\_data)
3. create\_payload(companyData)
4. get\_data\_from\_api(payload)
5. serializer(tableName, database\_data)

GitHub Link :