Database Query Execution and Data Extraction: Oracle SQL and MongoDB

Abstract

This document outlines a process for querying and extracting data from two different database systems: Oracle SQL and MongoDB. The procedure involves executing predefined queries on each database and exporting the results to Excel files. Users have to select the desired database, executing relevant queries, and seamlessly extracting data for further analysis.

Prepared By

Vaishnavi Kulkarni, under the guidance of Mohit Vaishnav.

Contents

Problem Statement

Given two different databases for SQL (Oracle) and NoSQL (MongoDB) , the requirement was to connect to the respective databases , perform some queries on the datasets and design a python program that can extract these databases from Oracle and MongoDB respectively and export only the resultant data obtained from executing the queries into a excel sheet, that can be used as a report for further analysis.

Data Needed

1. A dataset for Oracle and a dataset for MongoDB.

2. Credentials for connecting to the Oracle SQL database, including the username, password, host, port, and service name.

3. Credentials for connecting to the MongoDB database, including the database name and connection URL.

4. The SQL query to be executed on the Oracle SQL database to retrieve specific information.

5. The MongoDB aggregation pipeline query to be executed on the MongoDB database to retrieve specific information.

6. The filename and path for the Excel file where the extracted data will be saved.

Solution

The solution proposed involves two functions, each tailored to interact with a specific type of database:

1. Oracle SQL Database Interaction Function:

This function, fetch_from_oracle, connects to the Oracle SQL database using credentials stored in a JSON file. It executes a predefined SQL query to retrieve specific information from the database. The fetched data is processed and saved into an Excel file using the Pandas library for further analysis.

The ‘fetch_from_oracle’ function is as follows -

```
def fetch_from_oracle(sql_query, excel_filename):
    with open('db_creds.json') as f:
        db_creds = json.load(f)
    oracle_username = db_creds["oracle_username"]
    oracle_password = db_creds["oracle_password"]
    oracle_host = db_creds["oracle_host"]
    oracle_port = db_creds["oracle_port"]
    oracle_service_name = db_creds["oracle_service_name"]
    connection = cx_Oracle.connect(
        user=oracle_username,
        password=oracle_password,
        dsn=f"{oracle_host}:{oracle_port}/{oracle_service_name}"
    )
    cursor = connection.cursor()
    cursor.execute(sql_query)
    columns = [col[0] for col in cursor.description]
    result = cursor.fetchall()
    df = pd.DataFrame(result, columns=columns)
    cursor.close()
    connection.close()
    df.to_excel(excel_filename, index=False)
    print(f"Data from Oracle saved to {excel_filename}")
```

2. MongoDB Database Interaction Function:

The function ‘fetch_data’ connects to the MongoDB database using the specified database name. It executes a MongoDB aggregation pipeline query to retrieve specific information from the database. Similar to the Oracle SQL function, the fetched data is processed and saved into an Excel file using Pandas.

The function ‘fetch_data’ is as follows -

```
def fetch_data(database_name, query, output_file):
    client = pymongo.MongoClient("mongodb://localhost:27017/")
    db = client[database_name]
    result = db.movies.aggregate(query)
    data = list(result)
    df = pd.DataFrame(data)
    df['wins'] = df['awards'].apply(lambda x: x.get('wins', None))
    df['nominations'] = df['awards'].apply(lambda x: x.get('nominations', None))
    df['imdb rating'] = df['imdb'].apply(lambda x: x.get('rating', None))
    df['cast'] = df['cast'].apply(lambda x: ', '.join(x) if isinstance(x, list) else x)
    df['directors'] = df['directors'].apply(lambda x: ', '.join(x) if isinstance(x, list) else x)
    df.drop(['awards', 'imdb'], axis=1, inplace=True)
    df.to_excel(output_file, index=False)
    print("Data from MongoDB exported to Excel successfully!")
```

Users are prompted to choose the database they want to interact with, and based on their selection, the respective function is called to retrieve and export the data into an Excel file.

Team/Members Involved

K V Gopalakrishna

Deepak K Dubey

Mohit Vaishnav

Vaishnavi Kulkarni

Data Collection

1. The sample database used in Oracle is a ‘Sales_History’ database.

2. The sample database used in MongoDB is a ‘MOVIE’ database.

Sample input SQL query for Oracle database

```
SELECT C.CUST_ID, C.CUST_FIRST_NAME, C.CUST_LAST_NAME, SUM(S.AMOUNT_SOLD) AS
TOTAL_SALES
  FROM CUSTOMER C
  JOIN SALES S ON C.CUST_ID = S.CUST_ID
  GROUP BY C.CUST_ID, C.CUST_FIRST_NAME, C.CUST_LAST_NAME
  ORDER BY TOTAL_SALES DESC
  FETCH FIRST 10 ROWS ONLY
```

Sample output

CUST_ID	CUST_FIRST_NAME	CUST_LAST_NAME	TOTAL_SALES
11407	Dora	Rice	103412.66
10747	Lolita	Katz	99578.09
42167	Tesia	Eppling	98585.96
4974	Xerxes	Abbassi	98006.16
12783	Rose	Lehman	97573.55
6395	Mitchel	Polk	97010.48
2994	Ronald	Adams	94862.61
429	Meriel	Fairman	94819.41
1743	Harold	Allis	94786.13
4759	Roderica	Lavin	93644.32

Sample input MongoDB query for MongoDB database

```
[
    {
      "$match": {
        "year": { "$gte": 2000 },
        "imdb.rating": { "$gte": 6.0 }
      }
    },
    {
      "$project": {
        "_id": 1,
        "plot": 1,
        "cast": 1,
        "directors": 1,
        "awards": 1,
        "imdb": 1
      }
    }
  ]
```

Sample output

_id	plot	cast	directors	wins	nominations	imdb
573a	Kate and …	Meg Ryan, Hugh Jackman	James Mangold	2	4	6.3
573b	A modern day..	Crispin Glover, Vanessa	Menahem Golan	2	0	6.4

Observations

- The use of a JSON file to store database credentials ensures secure connectivity to Oracle SQL and MongoDB databases.
- Credentials are programmatically accessed, reducing the risk of exposure and enhancing security.