NoSQL_MongoDB_with_Python

September 6, 2023

1 Course Section : AIT614-005

1.1 Lab 2 : NoSQL MongoDB with Python

1.1.1 Student's Name : Rashmika Calve

[1]:

```
!pip install pymongo
```

```
Requirement already satisfied: pymongo in c:\users\rashmika\anaconda3\lib\site-
packages (4.3.3)
Requirement already satisfied: dnspython<3.0.0,>=1.16.0 in
c:\users\rashmika\anaconda3\lib\site-packages (from pymongo) (2.3.0)
```

Importing the required libraries

[2]:

```
import pymongo
import pandas as pd
import json
```

Connect to MongoDB

[3]:

```
client = pymongo.MongoClient("mongodb://localhost:27017/")
```

Load the csv file

[4]:

```
df = pd.read_csv("EmployeeAttrition.csv")
df.head(10)
```

[4]:

```
   Age Attrition     BusinessTravel  DailyRate              Department  \
0   41       Yes      Travel_Rarely       1102                   Sales
1   49        No  Travel_Frequently        279  Research & Development
2   37       Yes      Travel_Rarely       1373  Research & Development
3   33        No  Travel_Frequently       1392  Research & Development
4   27        No      Travel_Rarely        591  Research & Development
5   32        No  Travel_Frequently       1005  Research & Development
6   59        No      Travel_Rarely       1324  Research & Development
7   30        No      Travel_Rarely       1358  Research & Development
8   38        No  Travel_Frequently        216  Research & Development
9   36        No      Travel_Rarely       1299  Research & Development
```

```
   DistanceFromHome  Education EducationField  EmployeeCount  EmployeeNumber  \
0                 1          2  Life Sciences              1               1
1                 8          1  Life Sciences              1               2
2                 2          2          Other              1               4
3                 3          4  Life Sciences              1               5
4                 2          1        Medical              1               7
5                 2          2  Life Sciences              1               8
6                 3          3        Medical              1              10
7                24          1  Life Sciences              1              11
8                23          3  Life Sciences              1              12
9                27          3        Medical              1              13

   …  RelationshipSatisfaction StandardHours  StockOptionLevel  \
0  …                         1            80                 0
1  …                         4            80                 1
2  …                         2            80                 0
3  …                         3            80                 0
4  …                         4            80                 1
5  …                         3            80                 0
6  …                         1            80                 3
7  …                         2            80                 1
8  …                         2            80                 0
9  …                         2            80                 2

   TotalWorkingYears  TrainingTimesLastYear WorkLifeBalance  YearsAtCompany  \
0                  8                      0               1               6
1                 10                      3               3              10
2                  7                      3               3               0
3                  8                      3               3               8
4                  6                      3               3               2
5                  8                      2               2               7
6                 12                      3               2               1
7                  1                      2               3               1
8                 10                      2               3               9
9                 17                      3               2               7

  YearsInCurrentRole  YearsSinceLastPromotion  YearsWithCurrManager
0                  4                        0                     5
1                  7                        1                     7
2                  0                        0                     0
3                  7                        3                     0
4                  2                        2                     2
5                  7                        3                     6
6                  0                        0                     0
7                  0                        0                     0
8                  7                        1                     8
9                  7                        7                     7
```

```
[10 rows x 35 columns]
```

[5]:

```
df.shape
```

[5]:

```
(1470, 35)
```

Converting the dataframe to JSON format and Loading to MongoDB

[6]:

```
# Converting the df to json
emp_data = json.loads(df.to_json(orient='records'))
```

Creating a Database

[7]:

```
mongo_db = client["myDB"]
mongo_db
```

[7]:

```
Database(MongoClient(host=['localhost:27017'], document_class=dict,
tz_aware=False, connect=True), 'myDB')
```

Creating a collection

[8]:

```
collection_nm = "Empl_Attrition"
```

[9]:

```
collection_nm = mongo_db[collection_nm]
```

Insert the data into MongoDB Collection

[10]:

```
collection_nm.insert_many(emp_data)
```

[10]:

```
<pymongo.results.InsertManyResult at 0x2285f173100>
```

Query MongoDB

Count the total no of documents in the collection

[11]:

```
collection_nm.count_documents({})
```

[11]:

```
1470
```

1. Count the no of employees whose TotalWorkingYears are greater than 20.

[12]:

```
collection_nm.count_documents({
    "TotalWorkingYears" : {"$gt" : 20}  })
```

[12]:

```
207
```

2. Find EmployeeNumber, EducationField, JobRole for all the employees whose Age is between 25 and 30 and Education is 5. Display only EmployeeNumber, EducationField, and JobRobe in the output.

[13]:
```
res = collection_nm.find({'Age' : {'$gte' : 25, '$lte' : 30},
                         "Education" : 5},
                         {'EmployeeNumber', 'EducationField','JobRole'}
                        )
print('EmployeeNumber', '\t', 'EducationField', '\t\t', 'JobRole')
print('---------------------------------------------------------')
for r in res:
    print(r['EmployeeNumber'], '\t\t', r['JobRole'], '\t\t',␣
 ↪r['EducationField'])
```

```
EmployeeNumber    EducationField                 JobRole
--------------------------------------------------------
455               Laboratory Technician          Other
565               Research Scientist             Technical Degree
747               Sales Executive                Marketing
1094              Laboratory Technician          Life Sciences
```

3. For all the women employees having Age between 35 and 40 and TotalWorkingYears < 5, sort EmployeeNumber in an ascending order. Print only Department and EmployeeNumber in the output.

[14]:
```
# Adding conditions to the find function
emp_res= collection_nm.find(
    {"$and": [
        {"Gender" : 'Female'},
        {'Age' : {'$gte' : 35}},
        {'Age' : {'$lte' : 40}},
        {'TotalWorkingYears' : {'$lt':5}}
    ]},
    {'EmployeeNumber','Department'}
).sort('EmployeeNumber',1)
```

[15]:
```
# Converting the cursor to a list
emp_res_list = list(emp_res)
```

[16]:
```
# Converting the list to a dataframe
emp_df_3 = pd.DataFrame(emp_res_list)
emp_df_3.shape
```

[16]:
```
(9, 3)
```

[17]:
```
#Displaying the results
emp_df_3[['EmployeeNumber','Department']]
```

```
[17]:    EmployeeNumber              Department
      0              49                   Sales
      1              75  Research & Development
      2             245  Research & Development
      3             805                   Sales
      4            1569  Research & Development
      5            1662  Research & Development
      6            1675  Research & Development
      7            1886  Research & Development
      8            2052  Research & Development
```

4. Find employees whose HourlyRate is greater than or equal to 100 or DailyRate is greater than 1490. Display Age, HourlyRate, DailyRate, and Department only and sort DailyRate in an ascending order.

```
[18]: # Adding conditions to the find function
      emp_res4 = collection_nm.find(
          {'$or': [
              {'HourlyRate' : {'$gte':100}},
              {'DailyRate': {'$gt':1490}}
          ]},
          {'Age','HourlyRate','DailyRate','Department'}
      ).sort('DailyRate',1) # 1 means ascending order
```

```
[19]: # Converting the cursor to a list
      emp_res_list4 = list(emp_res4)
      emp_res_list4
```

```
[19]: [{'_id': ObjectId('63f5348794ccd61cbfa4e495'),
        'Age': 31,
        'DailyRate': 218,
        'Department': 'Sales',
        'HourlyRate': 100},
       {'_id': ObjectId('63f5348794ccd61cbfa4e79b'),
        'Age': 29,
        'DailyRate': 224,
        'Department': 'Research & Development',
        'HourlyRate': 100},
       {'_id': ObjectId('63f5348794ccd61cbfa4e55e'),
        'Age': 45,
        'DailyRate': 306,
        'Department': 'Sales',
        'HourlyRate': 100},
       {'_id': ObjectId('63f5348794ccd61cbfa4e911'),
        'Age': 38,
        'DailyRate': 345,
        'Department': 'Sales',
        'HourlyRate': 100},
       {'_id': ObjectId('63f5348794ccd61cbfa4e6a9'),
        'Age': 35,
        'DailyRate': 528,
        'Department': 'Human Resources',
        'HourlyRate': 100},
       {'_id': ObjectId('63f5348794ccd61cbfa4e3e6'),
        'Age': 22,
        'DailyRate': 594,
        'Department': 'Research & Development',
        'HourlyRate': 100},
       {'_id': ObjectId('63f5348794ccd61cbfa4e411'),
        'Age': 19,
        'DailyRate': 602,
        'Department': 'Sales',
        'HourlyRate': 100},
       {'_id': ObjectId('63f5348794ccd61cbfa4e752'),
        'Age': 26,
        'DailyRate': 652,
        'Department': 'Research & Development',
        'HourlyRate': 100},
       {'_id': ObjectId('63f5348794ccd61cbfa4e5c4'),
        'Age': 34,
        'DailyRate': 702,
        'Department': 'Research & Development',
        'HourlyRate': 100},
       {'_id': ObjectId('63f5348794ccd61cbfa4e457'),
        'Age': 32,
        'DailyRate': 976,
        'Department': 'Sales',
        'HourlyRate': 100},
       {'_id': ObjectId('63f5348794ccd61cbfa4e478'),
        'Age': 21,
        'DailyRate': 996,
        'Department': 'Research & Development',
        'HourlyRate': 100},
       {'_id': ObjectId('63f5348794ccd61cbfa4e3ce'),
        'Age': 37,
        'DailyRate': 1040,
        'Department': 'Research & Development',
        'HourlyRate': 100},
       {'_id': ObjectId('63f5348794ccd61cbfa4e4d5'),
        'Age': 50,
        'DailyRate': 1046,
        'Department': 'Research & Development',
        'HourlyRate': 100},
       {'_id': ObjectId('63f5348794ccd61cbfa4e913'),
        'Age': 36,
        'DailyRate': 1120,
        'Department': 'Sales',
        'HourlyRate': 100},
       {'_id': ObjectId('63f5348794ccd61cbfa4e607'),
        'Age': 33,
        'DailyRate': 1198,
        'Department': 'Research & Development',
        'HourlyRate': 100},
       {'_id': ObjectId('63f5348794ccd61cbfa4e46a'),
        'Age': 32,
        'DailyRate': 1311,
        'Department': 'Research & Development',
        'HourlyRate': 100},
       {'_id': ObjectId('63f5348794ccd61cbfa4e86f'),
        'Age': 38,
        'DailyRate': 1336,
        'Department': 'Human Resources',
        'HourlyRate': 100},
       {'_id': ObjectId('63f5348794ccd61cbfa4e76f'),
        'Age': 31,
        'DailyRate': 1445,
        'Department': 'Research & Development',
        'HourlyRate': 100},
       {'_id': ObjectId('63f5348794ccd61cbfa4e628'),
        'Age': 40,
        'DailyRate': 1479,
        'Department': 'Sales',
        'HourlyRate': 100},
       {'_id': ObjectId('63f5348794ccd61cbfa4e677'),
        'Age': 40,
        'DailyRate': 1492,
        'Department': 'Research & Development',
        'HourlyRate': 61},
       {'_id': ObjectId('63f5348794ccd61cbfa4e53d'),
        'Age': 38,
        'DailyRate': 1495,
        'Department': 'Research & Development',
        'HourlyRate': 76},
       {'_id': ObjectId('63f5348794ccd61cbfa4e77a'),
        'Age': 49,
        'DailyRate': 1495,
        'Department': 'Research & Development',
        'HourlyRate': 96},
       {'_id': ObjectId('63f5348794ccd61cbfa4e818'),
        'Age': 38,
        'DailyRate': 1495,
        'Department': 'Research & Development',
        'HourlyRate': 87},
       {'_id': ObjectId('63f5348794ccd61cbfa4e401'),
        'Age': 29,
        'DailyRate': 1496,
        'Department': 'Research & Development',
        'HourlyRate': 41},
       {'_id': ObjectId('63f5348794ccd61cbfa4e786'),
        'Age': 28,
        'DailyRate': 1496,
        'Department': 'Sales',
        'HourlyRate': 92},
       {'_id': ObjectId('63f5348794ccd61cbfa4e740'),
        'Age': 39,
        'DailyRate': 1498,
        'Department': 'Sales',
        'HourlyRate': 44},
       {'_id': ObjectId('63f5348794ccd61cbfa4e511'),
        'Age': 60,
        'DailyRate': 1499,
        'Department': 'Sales',
        'HourlyRate': 80}]
```

[20]:

```
# Converting the list to a dataframe
emp_df_4 = pd.DataFrame(emp_res_list4)
emp_df_4.shape
```

[20]:

```
(27, 5)
```

[21]:

```
#Displaying the results
emp_df_4.loc[ : , emp_df_4.columns != '_id']
```

[21]:

```
    Age  DailyRate              Department  HourlyRate
0    31        218                   Sales         100
1    29        224  Research & Development         100
2    45        306                   Sales         100
3    38        345                   Sales         100
4    35        528         Human Resources         100
5    22        594  Research & Development         100
6    19        602                   Sales         100
7    26        652  Research & Development         100
8    34        702  Research & Development         100
9    32        976                   Sales         100
10   21        996  Research & Development         100
11   37       1040  Research & Development         100
12   50       1046  Research & Development         100
13   36       1120                   Sales         100
```

```
14   33       1198  Research & Development         100
15   32       1311  Research & Development         100
16   38       1336         Human Resources         100
17   31       1445  Research & Development         100
18   40       1479                   Sales         100
19   40       1492  Research & Development          61
20   38       1495  Research & Development          76
21   49       1495  Research & Development          96
22   38       1495  Research & Development          87
23   29       1496  Research & Development          41
24   28       1496                   Sales          92
25   39       1498                   Sales          44
26   60       1499                   Sales          80
```

5. For each JobRole, find the average MonthlyIncome. Print out the formatted monthly incomes in hundredth and arrange them in descending order.

[22]:

```
emp_res5 = collection_nm.aggregate([
    {"$group": {
        "_id" : "$JobRole",
        "avg_monthly_income" : {"$avg" : '$MonthlyIncome'}
    }},
    {"$sort" : {
        "avg_monthly_income" : -1 }
    }
])


#Printing the results
print('Job Role', '\t\t\t\t', 'Average Monthly Income')
print('-----------------------------------------------------------------')
for r in emp_res5:
    print(f"{r['_id'] : <25}{'{:.2f}' : >30}".format(r['avg_monthly_income']))
```

```
Job Role                                 Average Monthly Income
-----------------------------------------------------------------
Manager                                          17181.68
Research Director                                16033.55
Healthcare Representative                        7528.76
Manufacturing Director                           7295.14
Sales Executive                                  6924.28
Human Resources                                  4235.75
Research Scientist                               3239.97
Laboratory Technician                            3237.17
Sales Representative                             2626.00
```

6. Count the different MaritalStatus when Attrition is YES and AGE is greater than 35 in the dataset. Arrange the count in descending order.

[23]:

```
emp_res6 = collection_nm.aggregate([
    {
        '$match' : {
            '$and': [
                {"Attrition":'Yes'},
                {'Age': {'$gt' : 35}}
            ]
        }
    },
    {
        "$group" : {
            "_id" : "$MaritalStatus",
            "count_emp" : {"$sum" : 1}
        }
    },
    {"$sort" : {
        "count_emp" : -1 }
    }
])
```

[24]:

```
list(emp_res6)
```

[24]:

```
[{'_id': 'Married', 'count_emp': 33},
 {'_id': 'Single', 'count_emp': 30},
 {'_id': 'Divorced', 'count_emp': 14}]
```

Delete All Documents in a Collection

[25]:

```
collection_del = collection_nm.delete_many({})
```

Delete the Collection

[26]:

```
collection_nm.drop()
```

References [1] Dr. Liao's lab tutorials and code examples: Blackboard/Liao_PyMongo.html

[2] Python MongoDB - https://www.w3schools.com/python/python_mongodb_getstarted.asp

[3] PyMongoDB Documentation - https://pymongo.readthedocs.io/en/stable/