top-10-largest-banks_by-market-capitalization-USD

February 17, 2024

1 Scrapping data for the top 10 largest bank

1.0.1 import requirement

[ ]:
```
import pandas as pd
from bs4 import BeautifulSoup
import requests as req
from datetime import datetime
```

1.0.2 log function that write in a log text file

[144]:
```
def log_progress(txt):
    now = datetime.now()
    with open('code_log.txt ' ,'a') as f:
        f.write(str(now)+' : '+txt+'\n')
```

1.0.3 extract function that scrap data from a website and store it in a dataframe

[144]:
```
def extract():
    l=[]
    #url= 'https://en.wikipedia.org/wiki/List_of_largest_banks'
    url = 'https://web.archive.org/web/20230908091635/https://en.wikipedia.org/
 ↪wiki/List_of_largest_banks'
    content = req.get(url)

    tree = BeautifulSoup(content.content , 'html.parser')
    table = tree.find('tbody')
    lines = table.find_all('tr')[1:]
    cols = []
    for line in lines:
        cols.append([td.get_text(strip=True) for td in line.find_all('td')])
    df = pd.DataFrame(cols , columns=['Rank' , 'Bank name' , 'Market cap $'])
    df['Rank'] = df['Rank'].transform(lambda x : int(x))
    df['Market cap $'] = df['Market cap $'].transform(lambda x: float(x))
    return df
```

1.0.4 transform function that add 3 column for echange based on a csv file

[145]:
```
#Market Capitalization in GBP, EUR and INR
exch = pd.read_csv('exchange_rate.csv' , index_col =0)
rateEUR = exch.loc['EUR',:]
rateGBP = exch.loc['GBP',:]
rateINR = exch.loc['INR',:]
def transform(df):
    df['Market cap in GBP'] = df['Market cap $'].transform(lambda x :␣
 ↪round(x*rateGBP,2))
    df['Market cap in EUR'] = df['Market cap $'].transform(lambda x :␣
 ↪round(x*rateEUR,2))
    df['Market cap in INR'] = df['Market cap $'].transform(lambda x :␣
 ↪round(x*rateINR,2))
    return df
```

1.0.5 load to csv function

[146]:
```
def load_to_csv(df):
    df.to_csv('./Largest_banks_data.csv' , index=False)
```

1.0.6 load to data base function

[147]:
```
#Banks.db-Largest_banks
import sqlite3 as sql
def load_to_db(df):

    df.to_sql(name='Largest_banks' , index=False , con=conn ,␣
 ↪if_exists='replace')
```

1.0.7 run query function

[148]:
```
def run_query(query_statement, sql_connection):
    dic ={}
    cur = sql_connection.cursor()
    res = cur.execute(query_statement).fetchall()
    cur.close()
    for b in res:
        dic[b[0]]=b[1]
    return dic
```

1.0.8 Main

[149]:
```
conn = sql.connect('Banks.db')
log_progress('_____--START--_____')
log_progress('Start Extracting')
df = extract()
log_progress('End Extracting')
log_progress('Start Transformation')
transform(df)
log_progress('End Transformation')
log_progress('Load to csv')
load_to_csv(df)
log_progress('Load to database')
load_to_db(df)
log_progress('--_____END_____--')
```

[150]:
```
q0 = "select `Bank name` , `Market cap in GBP` as MC_GBP_Billion from␣
 ↪Largest_banks "
q1 = "select `Bank name` , `Market cap in EUR` as MC_EUR_Billion from␣
 ↪Largest_banks "
q2 = "select `Bank name` , `Market cap in INR` as MC_INR_Billion from␣
 ↪Largest_banks "
print(run_query(q0 , conn) , end='\n')
```

```
{'JPMorgan Chase': 346.34, 'Bank of America': 185.22, 'Industrial and Commercial
Bank of China': 155.65, 'Agricultural Bank of China': 128.54, 'HDFC Bank':
126.33, 'Wells Fargo': 124.7, 'HSBC Holdings PLC': 119.12, 'Morgan Stanley':
112.66, 'China Construction Bank': 111.86, 'Bank of China': 109.45}
```

[151]:
```
print(run_query(q1 , conn) , end='\n')
```

```
{'JPMorgan Chase': 402.62, 'Bank of America': 215.31, 'Industrial and Commercial
Bank of China': 180.94, 'Agricultural Bank of China': 149.43, 'HDFC Bank':
146.86, 'Wells Fargo': 144.96, 'HSBC Holdings PLC': 138.48, 'Morgan Stanley':
130.97, 'China Construction Bank': 130.03, 'Bank of China': 127.23}
```

[152]:
```
print(run_query(q2 , conn), end='\n')
```

```
{'JPMorgan Chase': 35910.71, 'Bank of America': 19204.58, 'Industrial and
Commercial Bank of China': 16138.75, 'Agricultural Bank of China': 13328.41,
'HDFC Bank': 13098.63, 'Wells Fargo': 12929.42, 'HSBC Holdings PLC': 12351.26,
'Morgan Stanley': 11681.85, 'China Construction Bank': 11598.07, 'Bank of
China': 11348.39}
```

[153]:
```
conn.close()
```