# StockETL-Databricks-AzureSQL

November 29, 2023

[ ]:
```
import pandas as pd
from bsedata.bse import BSE  ## if not installed use pip install bsedata
import time
from pyspark.sql import SparkSession

pd.set_option('display.max_rows', 500)
pd.set_option('display.max_columns', 500)
pd.set_option('display.width', 1000)

github_excel_url = "https://raw.githubusercontent.com/jangid6/Stock-ETL-Project/
 ↪main/Equity.xlsx"
engine = 'openpyxl'  ## if not installed use pip install openpyxl
EquityDF = pd.read_excel(github_excel_url, engine = engine)
EquityDF.head(n=2) # Get the list of stocks in Nifty50
```

[ ]:
```
   Security Code           Issuer Name Security Id         Security Name  Status
Group  Face Value       ISIN No                    Industry Instrument  Sector
Name            Industry New Name           Igroup Name              ISubgroup
Name
0         500002     ABB India Limited         ABB     ABB India Limited  Active
A          2.0  INE117A01022  Heavy Electrical Equipment     Equity  Industrials
Capital Goods  Electrical Equipment  Heavy Electrical Equipment
1         500003  Aegis Logistics Ltd.    AEGISLOG  AEGIS LOGISTICS LTD.  Active
A          1.0  INE208C01025               Trading - Gas     Equity       Energy
Oil, Gas & Consumable Fuels                   Gas               Trading - Gas
```

[ ]:
```
listOf_Nifty50_StockIDs = [
    "ADANIENT", "ADANIPORTS", "APOLLOHOSP", "ASIANPAINT", "AXISBANK",
    "BAJAJ-AUTO", "BAJFINANCE", "BAJAJFINSV", "BPCL", "BHARTIARTL",
    "BRITANNIA", "CIPLA", "COALINDIA", "DIVISLAB", "DRREDDY", "EICHERMOT",
    "GRASIM", "HCLTECH", "HDFCBANK", "HDFCLIFE", "HEROMOTOCO", "HINDALCO",
    "HINDUNILVR", "ICICIBANK", "ITC", "INDUSINDBK", "INFY", "JSWSTEEL",
    "KOTAKBANK", "LTIM", "LT", "M&M", "MARUTI", "NTPC", "NESTLEIND",
    "ONGC", "POWERGRID", "RELIANCE", "SBILIFE", "SBIN", "SUNPHARMA",
    "TCS", "TATACONSUM", "TATAMOTORS", "TATASTEEL", "TECHM", "TITAN",
    "UPL", "ULTRACEMCO", "WIPRO"
]
```

```
## We Want to filter EquityDF by looking up 'Security Id' col values exist in␣
 ↪listOf_Nifty50_StockIDs
EquityDF['Security Code'] = EquityDF['Security Code'].astype(str) ##coverting␣
 ↪into str as getQuote() accepts str input
nifty50_OnlyDF= EquityDF[EquityDF['Security Id'].isin(listOf_Nifty50_StockIDs)].
 ↪reset_index(drop=True)
nifty50_OnlyDF.columns = nifty50_OnlyDF.columns.str.replace(' ', '')

bseObject = BSE(update_codes=True)  ##creating Bse Lib Object
listof_stockDicts = []
sqcode_ListNifty50 = nifty50_OnlyDF['SecurityCode'].values
for sqCode in sqcode_ListNifty50:
    try:
        stockDict = bseObject.getQuote(sqCode)
        stockDict.pop("buy", None)
        stockDict.pop("sell", None)
        listof_stockDicts.append(stockDict)
        time.sleep(0.5)
    except IndexError:
        print(f"IndexError for {sqCode}: Data not available")

nifty50DailyTable = pd.DataFrame(listof_stockDicts)
nifty50DailyTable.head()
```

[ ]:
```
                    companyName currentValue  change pChange
updatedOn  securityID scripCode                group faceValue
industry previousClose previousOpen  dayHigh   dayLow 52weekHigh 52weekLow
weightedAvgPrice totalTradedValue totalTradedQuantity 2WeekAvgQuantity
marketCapFull marketCapFreeFloat
0         Bajaj Finance Limited      7129.75    3.60    0.05  29 Nov 23 | 04:01
PM  BAJFINANCE    500034  A  / S&P BSE SENSEX      2.00  Financial Services
7126.15      7130.05  7169.90  7115.00    8190.00   5487.25          7133.20
8.93 Cr.           0.13 Lakh        0.52 Lakh  4,40,635.00 Cr.    1,93,879.40
Cr.
1                   CIPLA LTD.      1203.55   11.50    0.96  29 Nov 23 | 04:01
PM       CIPLA    500087     A  / S&P BSE 100      2.00          Healthcare
1192.05      1194.55  1205.90  1191.75    1283.00    852.00          1201.07
8.66 Cr.           0.72 Lakh        0.81 Lakh    97,165.65 Cr.      64,129.33
Cr.
2           STATE BANK OF INDIA       568.50    3.95    0.70  29 Nov 23 | 04:01
PM        SBIN    500112  A  / S&P BSE SENSEX      1.00  Financial Services
564.55       567.30   569.00   565.05     629.65    499.35           567.51
36.83 Cr.          6.49 Lakh        5.75 Lakh  5,07,364.19 Cr.    2,18,166.60
Cr.
3         Titan Company Limited      3425.05  -17.00   -0.49  29 Nov 23 | 04:01
PM       TITAN    500114  A  / S&P BSE SENSEX      1.00   Consumer Durables
3442.05      3455.00  3470.00  3423.60    3470.00   2268.90          3438.65
```

```
3.33 Cr.           0.10 Lakh        0.28 Lakh  3,04,071.20 Cr.    1,42,913.46
Cr.
4  DR.REDDY'S LABORATORIES LTD.      5719.85   48.60    0.86  29 Nov 23 | 04:01
PM     DRREDDY    500124     A  / S&P BSE 100      5.00          Healthcare
5671.25      5700.00  5742.45  5654.55    5986.20   4176.85          5704.21
3.83 Cr.           0.07 Lakh        0.16 Lakh    95,400.83 Cr.      69,642.61
Cr.
```

[ ]:
```
nifty50DailyTable.rename(columns={'group': 'sharegroup'}, inplace=True)
nifty50DailyTable.rename(columns={'52weekHigh': 'fiftytwoweekHigh'},␣
 ↪inplace=True)
nifty50DailyTable.rename(columns={'52weekLow': 'fiftytwoweekLow'}, inplace=True)
nifty50DailyTable.rename(columns={'2WeekAvgQuantity': 'twoWeekAvgQuantity'},␣
 ↪inplace=True)
# Convert 'updatedOn' column to datetime and extract date
nifty50DailyTable['updatedOn'] = pd.to_datetime(nifty50DailyTable['updatedOn'],␣
 ↪format='%d %b %y | %I:%M %p', errors='coerce')

# Check if there are any invalid or missing date values
if pd.isna(nifty50DailyTable['updatedOn']).any():
    print("There are invalid or missing date values in the 'updatedOn' column.")
else:
    # Extract date from 'updatedOn' column and convert the column to datetime
    nifty50DailyTable['updatedOn'] = pd.
 ↪to_datetime(nifty50DailyTable['updatedOn'].dt.date)

if 'totalTradedValueCr' not in nifty50DailyTable.columns:
   # Assuming nifty50DailyTable is your DataFrame
    nifty50DailyTable['totalTradedValueCr'] = pd.
 ↪to_numeric(nifty50DailyTable['totalTradedValue'].str.replace(',', '').str.
 ↪replace(' Cr.', '', regex=True), errors='coerce')  # Convert to numeric and␣
 ↪handle 'Cr.'
    nifty50DailyTable['totalTradedQuantityLakh'] = pd.
 ↪to_numeric(nifty50DailyTable['totalTradedQuantity'].str.replace(',', '').str.
 ↪replace(' Lakh', '', regex=True), errors='coerce')  # Convert to numeric and␣
 ↪handle 'Lakh'
    nifty50DailyTable['twoWeekAvgQuantityLakh'] = pd.
 ↪to_numeric(nifty50DailyTable['twoWeekAvgQuantity'].str.replace(',', '').str.
 ↪replace(' Lakh', '', regex=True), errors='coerce')  # Convert to numeric and␣
 ↪handle 'Lakh'
    nifty50DailyTable['marketCapFullCr'] = pd.
 ↪to_numeric(nifty50DailyTable['marketCapFull'].str.replace(',', '').str.
 ↪replace(' Cr.', '', regex=True), errors='coerce')  # Convert to numeric and␣
 ↪handle 'Cr.'
```

```
    nifty50DailyTable['marketCapFreeFloatCr'] = pd.
 ↪to_numeric(nifty50DailyTable['marketCapFreeFloat'].str.replace(',', '').str.
 ↪replace(' Cr.', '', regex=True), errors='coerce')  # Convert to numeric and␣
 ↪handle 'Cr.'

    # Drop original columns
    nifty50DailyTable.drop(['totalTradedValue',␣
 ↪'totalTradedQuantity','twoWeekAvgQuantity', 'marketCapFull',␣
 ↪'marketCapFreeFloat'], axis=1, inplace=True)

nifty50DailyTable.head(n=2)
```

[ ]:
```
             companyName currentValue change pChange  updatedOn  securityID
scripCode           sharegroup faceValue            industry previousClose
previousOpen  dayHigh   dayLow fiftytwoweekHigh fiftytwoweekLow weightedAvgPrice
totalTradedValueCr  totalTradedQuantityLakh  twoWeekAvgQuantityLakh
marketCapFullCr  marketCapFreeFloatCr
0  Bajaj Finance Limited      7129.75   3.60    0.05 2023-11-29  BAJFINANCE
500034  A  / S&P BSE SENSEX      2.00  Financial Services       7126.15
7130.05  7169.90  7115.00          8190.00         5487.25          7133.20
8.93                    0.13                     0.52        440635.00
193879.40
1             CIPLA LTD.      1203.55  11.50    0.96 2023-11-29       CIPLA
500087     A  / S&P BSE 100      2.00          Healthcare       1192.05
1194.55  1205.90  1191.75          1283.00          852.00          1201.07
8.66                    0.72                     0.81         97165.65
64129.33
```

[ ]:
```
# Convert 'updatedOn' column to datetime and extract date
nifty50DailyTable['updatedOn'] = pd.to_datetime(nifty50DailyTable['updatedOn'],␣
 ↪format='%d %b %y | %I:%M %p', errors='coerce')

# Check if there are any invalid or missing date values
if pd.isna(nifty50DailyTable['updatedOn']).any():
    print("There are invalid or missing date values in the 'updatedOn' column.")
else:
    # Extract date from 'updatedOn' column and convert the column to datetime
    nifty50DailyTable['updatedOn'] = pd.
 ↪to_datetime(nifty50DailyTable['updatedOn'].dt.date)

if 'totalTradedValueCr' not in nifty50DailyTable.columns:
   # Assuming nifty50DailyTable is your DataFrame
    nifty50DailyTable['totalTradedValueCr'] = pd.
 ↪to_numeric(nifty50DailyTable['totalTradedValue'].str.replace(',', '').str.
 ↪replace(' Cr.', '', regex=True), errors='coerce')  # Convert to numeric and␣
 ↪handle 'Cr.'
```

```
    nifty50DailyTable['totalTradedQuantityLakh'] = pd.
↪to_numeric(nifty50DailyTable['totalTradedQuantity'].str.replace(',', '').str.
↪replace(' Lakh', '', regex=True), errors='coerce')  # Convert to numeric and␣
↪handle 'Lakh'
    nifty50DailyTable['twoWeekAvgQuantityLakh'] = pd.
↪to_numeric(nifty50DailyTable['twoWeekAvgQuantity'].str.replace(',', '').str.
↪replace(' Lakh', '', regex=True), errors='coerce')  # Convert to numeric and␣
↪handle 'Lakh'
    nifty50DailyTable['marketCapFullCr'] = pd.
↪to_numeric(nifty50DailyTable['marketCapFull'].str.replace(',', '').str.
↪replace(' Cr.', '', regex=True), errors='coerce')  # Convert to numeric and␣
↪handle 'Cr.'
    nifty50DailyTable['marketCapFreeFloatCr'] = pd.
↪to_numeric(nifty50DailyTable['marketCapFreeFloat'].str.replace(',', '').str.
↪replace(' Cr.', '', regex=True), errors='coerce')  # Convert to numeric and␣
↪handle 'Cr.'

    # Drop original columns
    nifty50DailyTable.drop(['totalTradedValue',␣
↪'totalTradedQuantity','twoWeekAvgQuantity', 'marketCapFull',␣
↪'marketCapFreeFloat'], axis=1, inplace=True)

# Create a Spark session
spark = SparkSession.builder.appName("Nifty50DailyData").getOrCreate()
nifty50DailyDailyData_spark_df = spark.createDataFrame(nifty50DailyTable)
```

[ ]:
```
nifty50DailyDailyData_spark_df.toPandas().head() #You can also use SparkDF.
↪show(), I prefer in .toPandas()
```

[ ]:
```
                    companyName currentValue  change pChange  updatedOn
securityID scripCode           sharegroup faceValue            industry
previousClose previousOpen  dayHigh   dayLow fiftytwoweekHigh fiftytwoweekLow
weightedAvgPrice  totalTradedValueCr  totalTradedQuantityLakh
twoWeekAvgQuantityLakh  marketCapFullCr  marketCapFreeFloatCr
0         Bajaj Finance Limited      7129.75    3.60    0.05 2023-11-29
BAJFINANCE    500034  A  / S&P BSE SENSEX      2.00  Financial Services
7126.15      7130.05  7169.90  7115.00          8190.00         5487.25
7133.20                8.93                      0.13                    0.52
440635.00             193879.40
1                   CIPLA LTD.      1203.55   11.50    0.96 2023-11-29
CIPLA    500087     A  / S&P BSE 100      2.00          Healthcare       1192.05
1194.55  1205.90  1191.75          1283.00          852.00          1201.07
8.66                     0.72                    0.81         97165.65
64129.33
2           STATE BANK OF INDIA       568.50    3.95    0.70 2023-11-29
SBIN    500112  A  / S&P BSE SENSEX      1.00  Financial Services        564.55
567.30   569.00   565.05           629.65          499.35           567.51
```

```
36.83                    6.49                    5.75        507364.19
218166.60
3         Titan Company Limited      3425.05  -17.00   -0.49 2023-11-29
TITAN    500114  A  / S&P BSE SENSEX      1.00   Consumer Durables       3442.05
3455.00  3470.00  3423.60          3470.00         2268.90          3438.65
3.33                    0.10                    0.28        304071.20
142913.46
4  DR.REDDY'S LABORATORIES LTD.      5719.85   48.60    0.86 2023-11-29
DRREDDY    500124     A  / S&P BSE 100      5.00          Healthcare
5671.25      5700.00  5742.45  5654.55          5986.20         4176.85
5704.21                3.83                    0.07                    0.16
95400.83              69642.61
```

[ ]:
```
jdbcHostname = "mainsqldbserver.database.windows.net"
jdbcDatabase = "nifty50db"
jdbcPort = 1433
jdbcUrl = "jdbc:sqlserver://{0}:{1};database={2}".format(jdbcHostname,␣
 ↪jdbcPort, jdbcDatabase)
connectionProperties = {
  "user" : "jangid6",
  "password" : "EnterYourPassWord",
  "driver" : "com.microsoft.sqlserver.jdbc.SQLServerDriver"
}
table_name = "nifty50Table"
nifty50_dailydata_exists = False
try:
  spark.read.jdbc(url=jdbcUrl, table=table_name,␣
 ↪properties=connectionProperties)
  nifty50_dailydata_exists = True
except:
    columns = [
      "`companyName` VARCHAR(50)",
      "`currentValue` FLOAT",
      "`change` FLOAT",
      "`pChange` FLOAT",
      "`updatedOn` DATE",
      "`securityID` VARCHAR(50)",
      "`scripCode` VARCHAR(50)",
      "`sharegroup` VARCHAR(50)",
      "`faceValue` FLOAT",
      "`industry` VARCHAR(50)",
      "`previousClose` FLOAT",
      "`previousOpen` FLOAT",
      "`dayHigh` FLOAT",
      "`dayLow` FLOAT",
      "`fiftytwoweekHigh` FLOAT",
      "`fiftytwoweekLow` FLOAT",
```

```
        "`weightedAvgPrice` FLOAT",
        "`totalTradedQuantityLakh` FLOAT",
        "`totalTradedValueCr` FLOAT",
        "`twoWeekAvgQuantityLakh` FLOAT",
        "`marketCapFullCr` FLOAT",
        "`marketCapFreeFloatCr` FLOAT"
    ]
    drop_query = f"DROP TABLE {table_name}"
    spark.sql(drop_query)
    create_query = f"CREATE TABLE {table_name} ({','.join(columns)})"
    spark.sql(create_query)
    nifty50DailyDailyData_spark_df.
 ↪createOrReplaceTempView("nifty50dailydata_temp_table")
    nifty50DailyDailyData_spark_df.write.jdbc(url=jdbcUrl, table=table_name,␣
 ↪mode="overwrite",
                                              properties=connectionProperties)
    nifty50_dailydata_exists = True
```

[ ]:
```
queryMaxDate = f"SELECT MAX(updatedOn) as max_date FROM {table_name}"
queryresult = spark.read.jdbc(url=jdbcUrl, table="({0}) temp".
 ↪format(queryMaxDate), properties=connectionProperties)
sql_max_date = queryresult.first()[0]
df_max_updatedOn = nifty50DailyTable['updatedOn'].max()
if (sql_max_date == None) or (sql_max_date < df_max_updatedOn):
    nifty50DailyDailyData_spark_df.write.jdbc(url=jdbcUrl, table=table_name,␣
 ↪mode="append",
                                              properties=connectionProperties)
    print("[update Completed] Table is updated with latest data")
else:
    print("[No Update Required] Table is already updated with latest data")
```

```
[update Completed] Table is updated with latest data
```