How to retrieve data using sqlalchemy from sqlite db in pandas?

Asked today Modified today Viewed 14 times



i am not super skilled with python and i have the following problem. i am trying to make a python script where i download some data from yhaoo finance, if there is no db already available create it, if there is, upload the rows that are not already available. (incremental process). For the moment, i am trying to retrieve data from db with a query. The code is the following:





```
import pandas as pd
import yfinance as yf
import logging
from datetime import datetime, timedelta
from sqlalchemy import create_engine, Table, MetaData
class DataDownloader():
    def __init__(self, ASSET='TSLA'):
        self.ASSET = ASSET
    def download_and_save_data(self,interval='1d'):
        period = '10y'
        if interval[-1] == 'h':
            period = '719d'
        logging.info("Data download in progress")
        file_dir = os.path.dirname(os.path.abspath(__file__))
        data folder = 'staging1 download'
        file_path = os.path.join(file_dir, data_folder,
f'data_ASSET_{self.ASSET}_interval_{interval}.txt')
        data = yf.download(tickers=self.ASSET, period=period, interval=interval)
        data.sort_index(ascending=True)
        if interval[-1] == 'h':
            data = data.reset_index()
            data['Date'] = data['Datetime']
            data['Date'] = data['Date'].astype(str)
            data['Date'] = data['Date'].str[:-6]
            data = data.set index('Date')
            cols_to_delete = [
                'Adj Close',
                'Datetime',
        else:
            cols_to_delete = [
            'Adj Close',
        data = data.drop(columns=cols_to_delete)
```

```
#managing last available date. We want candles when markets has generated them.
        last date available = data.index[-1]
        current date = datetime.now()
        if interval[-1] == 'd':
            current_date = current_date.replace(hour=0, minute=0, second=0,
microsecond=0)
            if isinstance(last date available, str):
                last date available = datetime.strptime(last date available, '%Y-%m-
%d')
        elif interval[-1] == 'h':
            current_date = current_date.replace(minute=0, second=0, microsecond=0)
            if isinstance(last_date_available, str):
                last date available = datetime.strptime(last date available, '%Y-%m-%d
%H:%M:%S')
        #logging.info(f"data downloder last date available {last date available} and
current {current_date}")
        #logging.info(f"Cond verifiaction {str(current_date) == last_date_available}")
        if current date == last date available:
            data = data[:-1]
        elif current_date < last_date_available:</pre>
            raise Exception("Something is wrong with dates during download. Please
verify with data provider.")
        data.to csv(file path, header=True, index=True, sep=";")
        #when we switch to db
        file_dir = os.path.dirname(os.path.abspath(__file__))
        data_folder = 'staging1_download\\'
        database_path = os.path.join(file_dir, data_folder )
        databasePathName = database_path + 'staging1_download_DailyData.sqlite'
        print(databasePathName)
        #data["interval"] = interval
        data["Asset"] = self.ASSET
        data["INTERVAL"] = interval
        if not os.path.isfile(databasePathName):
            if interval[-1] == 'd':
                engine = create_engine(f'sqlite:///{databasePathName}', echo=True)
                data.to_sql("staging1_download_DailyData", con=engine, index=True)
        else:
            engine = create_engine(f'sqlite:///{databasePathName}', echo=True)
            query = f"""
            SELECT *
            FROM staging1_download_DailyData
            WHERE ASSET={self.ASSET} AND INTERVAL={interval}
            df = pd.read_sql(query, engine)
```

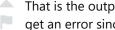
```
print(df.shape)
logging.info("Data download executed")
```

The script should be runned two times. AT first will create the db correctly. At the second should print the df. But no df is printed. Instead it printed the following:

```
2024-02-16 23:48:27,522 INFO sqlalchemy.engine.Engine PRAGMA main.table_info("
            SELECT *
            FROM staging1 download DailyData
            WHERE ASSET=TSLA AND INTERVAL=1d
2024-02-16 23:48:27,524 INFO sqlalchemy.engine.Engine [raw sql] ()
2024-02-16 23:48:27,526 INFO sqlalchemy.engine.Engine PRAGMA temp.table_info("
           SELECT *
            FROM staging1_download_DailyData
           WHERE ASSET=TSLA AND INTERVAL=1d
            ")
2024-02-16 23:48:27,528 INFO sqlalchemy.engine.Engine [raw sql] ()
2024-02-16 23:48:27,531 INFO sqlalchemy.engine.Engine
            SELECT *
            FROM staging1_download_DailyData
           WHERE ASSET=TSLA AND INTERVAL=1d
2024-02-16 23:48:27,532 INFO sqlalchemy.engine.Engine [raw sql] ()
2024-02-16 23:48:27,533 INFO sqlalchemy.engine.Engine ROLLBACK
2024-02-16 23:26:44,201 INFO sqlalchemy.engine.Engine [raw sql] ()
2024-02-16 23:26:44,208 INFO sqlalchemy.engine.Engine ROLLBACK
```

what that means? no error is printed in console

```
python
          pandas
                  sqlalchemy
                                etl
Share Edit Follow Flag
                                                                         asked 1 hour ago
                                                                                Lorenzo Galluzzi
                                                                            New contributor
```



That is the output produced by passing echo=True to create_engine(). I'm surprised that you don't get an error since your code builds a SELECT statement with WHERE ASSET=TSLA instead of WHERE ASSET='TSLA'. Does your table really have a column named "TSLA"? − Gord Thompson 1 hour ago ✓

1 Answer

Sorted by: Highest score (default)











• To properly incorporate variables self.ASSET and interval into the query, I enclosed them

• I utilized SQLAlchemy's text object by importing it with from sqlalchemy import text to

• Instead of passing the engine directly to pd.read_sql(), I passed a connection object (con)

in single quotes since they need to be treated as strings.

Here's the modified code that resolved the issue:

created using engine.connect().

create SQL select statement.

```
from sqlalchemy import create_engine, text
import pandas as pd
engine = create_engine(f'sqlite:///{databasePathName}', echo=True)
con = engine.connect()
query = text(f"""
SELECT *
FROM staging1_download_DailyData
WHERE ASSET='{self.ASSET}' AND INTERVAL='{interval}'
df = pd.read_sql(query, con)
print(df.shape)
```

Share Edit Delete Flag

answered 6 mins ago

