etl-through-python-postgres

August 31, 2022

Extract

1. Import csv_1 (pollutan o3) data
2. Import csv_2 (pollutan pm25) data

[ ]:
```
#import required dependencies
import pandas as pd
import datetime
```

[ ]:
```
#1. import csv_1 data
df1 = pd.read_csv("/home/achmadadyatma/Documents/learncode-dataEngineer_/
 ↪my-data-engineer_project/ETL-python-postgreSQL/all_years_o3.csv")
df1.head()
```

[ ]:

	Date	Country	City	Specie	count	min (ppb)	max (ppb)	median (ppb)
0	1/1/2017	US	Los Angeles	o3	24	2	31	10
1	1/1/2017	CN	Shanghai	o3	188	1	39	11
2	1/1/2017	US	Manhattan	o3	24	3	28	16
3	1/1/2017	US	Jacksonville	o3	46	2	18	12
4	1/1/2017	CN	Beijing	o3	235	1	8	2

[ ]:
```
#2. import csv_2 data
df2 = pd.read_csv("/home/achmadadyatma/Documents/learncode-dataEngineer_/
 ↪my-data-engineer_project/ETL-python-postgreSQL/all_years_pm25.csv")
df2.head()
```

[ ]:

	Date	Country	City	Specie	count	min (ug/m3)	max (ug/m3)
0	1/1/2017	IN	New Delhi	pm25	24	54.9	282.7
1	1/1/2017	CN	Shanghai	pm25	215	12.0	63.2
2	1/1/2017	US	Jacksonville	pm25	115	3.1	113.6
3	1/1/2017	US	Los Angeles	pm25	69	3.3	109.7

```
4  1/1/2017      US     Manhattan   pm25     96          4.3          23.0

   median (ug/m3)
0           177.7
1            34.4
2             8.1
3            15.9
4            12.1
```

Transform

df1

1. drop last 3 columns
2. rename the count columns
3. delete the specie column
4. convert date column as object type to datetime

[ ]:
```
# drop last 3 columns
df1.drop(df1.columns[[5,6,7]], axis=1, inplace=True)
df1.head()
```

[ ]:
```
        Date Country          City Specie  count
0  1/1/2017      US   Los Angeles     o3     24
1  1/1/2017      CN      Shanghai     o3    188
2  1/1/2017      US     Manhattan     o3     24
3  1/1/2017      US  Jacksonville     o3     46
4  1/1/2017      CN       Beijing     o3    235
```

[ ]:
```
#rename the count columns
df1.columns = ['Date', 'Country', 'City', 'Specie', 'Count_o3']
df1.head()
```

[ ]:
```
        Date Country          City Specie  Count_o3
0  1/1/2017      US   Los Angeles     o3        24
1  1/1/2017      CN      Shanghai     o3       188
2  1/1/2017      US     Manhattan     o3        24
3  1/1/2017      US  Jacksonville     o3        46
4  1/1/2017      CN       Beijing     o3       235
```

[ ]:
```
#delete specie
df1.drop(df1.columns[[3]], axis=1, inplace=True)
df1.head()
```

[ ]:
```
        Date Country          City  Count_o3
0  1/1/2017      US   Los Angeles        24
1  1/1/2017      CN      Shanghai       188
```

```
2  1/1/2017      US     Manhattan        24
3  1/1/2017      US  Jacksonville        46
4  1/1/2017      CN       Beijing       235
```

[ ]:
```
#convert 'date' column as object type to datetime
df1['Date'] = pd.to_datetime(df1['Date'])
df1.head()
```

[ ]:
```
        Date Country          City  Count_o3
0 2017-01-01      US   Los Angeles        24
1 2017-01-01      CN      Shanghai       188
2 2017-01-01      US     Manhattan        24
3 2017-01-01      US  Jacksonville        46
4 2017-01-01      CN       Beijing       235
```

df2

1. drop last 3 columns
2. rename the count columns
3. delete the specie column
4. convert date column as object type to datetime

[ ]:
```
# drop last 3 columns
df2.drop(df2.columns[[5,6,7]], axis=1, inplace=True)
df2.head()
```

[ ]:
```
        Date Country          City Specie  count
0  1/1/2017      IN     New Delhi   pm25     24
1  1/1/2017      CN      Shanghai   pm25    215
2  1/1/2017      US  Jacksonville   pm25    115
3  1/1/2017      US   Los Angeles   pm25     69
4  1/1/2017      US     Manhattan   pm25     96
```

[ ]:
```
#rename the count columns
df2.columns = ['Date', 'Country', 'City', 'Specie', 'Count_pm25']
df2.head()
```

[ ]:
```
        Date Country          City Specie  Count_pm25
0  1/1/2017      IN     New Delhi   pm25          24
1  1/1/2017      CN      Shanghai   pm25         215
2  1/1/2017      US  Jacksonville   pm25         115
3  1/1/2017      US   Los Angeles   pm25          69
4  1/1/2017      US     Manhattan   pm25          96
```

[ ]:
```
#delete specie
df2.drop(df2.columns[[3]], axis=1, inplace=True)
df2.head()
```

[ ]:

	Date	Country	City	Count_pm25
0	1/1/2017	IN	New Delhi	24
1	1/1/2017	CN	Shanghai	215
2	1/1/2017	US	Jacksonville	115
3	1/1/2017	US	Los Angeles	69
4	1/1/2017	US	Manhattan	96

[ ]:
```
#convert 'date' column as object type to datetime
df2['Date'] = pd.to_datetime(df2['Date'])
df2.head()
```

[ ]:

	Date	Country	City	Count_pm25
0	2017-01-01	IN	New Delhi	24
1	2017-01-01	CN	Shanghai	215
2	2017-01-01	US	Jacksonville	115
3	2017-01-01	US	Los Angeles	69
4	2017-01-01	US	Manhattan	96

merge 2 dfs

[ ]:
```
merge_df = pd.merge(df1, df2, how='left', left_on=['Date', 'Country','City'],␣
↪right_on=['Date', 'Country', 'City'])
merge_df.head()
```

[ ]:

	Date	Country	City	Count_o3	Count_pm25
0	2017-01-01	US	Los Angeles	24	69
1	2017-01-01	CN	Shanghai	188	215
2	2017-01-01	US	Manhattan	24	96
3	2017-01-01	US	Jacksonville	46	115
4	2017-01-01	CN	Beijing	235	386

Load

Connect to postgeSQL database

1. Method 1 - sqlalchemy
2. Method 2 - psycopg2

[ ]:
```
#CONNECT TO POSTGRESQL DATABASE METHOD 1
#SQLalchemy - general ORM (Object Relational Mapping) library

#IMPORT THE SQLALCHEMY LIBRARY's CREATE_ENGINE METHOD
from sqlalchemy import create_engine

#DEFINE THE DATABASE CREDENTIALS
user = 'achmadadyatma'
password = '541997'
host = '127.0.0.1'
```

```
port = '5432'
dbname ='learndb'

# PYTHON FUNCTION TO CONNECT TO THE POSTGRESQL DATABASE
# RETURN THE SQLALCHEMY ENGINE OBJECT
def get_connection_sqlalchemy():
        return create_engine(
                url="postgresql://{0}:{1}@{2}:{3}/{4}".format(
                        user, password, host, port, dbname
                )
        )

if __name__== '__main__':
        try:
                #GET THE CONNECTION OBJECT (ENGINE) FOR THE DATABASE
                engine = get_connection_sqlalchemy()
                print(f"connection to the {host} for user {user} and database␣
 ↪named {dbname} created successfully.")
        except Exception as ex:
                print("Connection could not be made due to the following error:␣
 ↪\n", ex)
```

```
connection to the 127.0.0.1 for user achmadadyatma and database named learndb
created successfully.
```

[ ]:
```
#CONNECT TO POSTGRESQL DATABASE METHOD 2
#psycopg2 - database driver for postgresql

#IMPORT THE THE PSYCOPG2 MODULE TO CREATE CONNECTION OBJECT
import psycopg2

#PYTHON FUNCTION TO CONNECT TO THE POSTGRESQL DATABASES AND DATABASE CREDENTIALS
def get_connection_psycopg2():
        return psycopg2.connect(
         user = 'achmadadyatma',
         password = '541997',
         host = '127.0.0.1',
         port = '5432',
         dbname ='learndb')

if __name__== '__main__':
        try:
                #GET THE CONNECTION OBJECT (DRIVER) FOR THE DATABASE
                driver = get_connection_psycopg2()
                print(f"connection to the {host} for user {user} and database␣
 ↪named {dbname} created successfully.")
        except Exception as ex:
```

```
            print("Connection could not be made due to the following error:␣
 ↪\n", ex)
```

```
connection to the 127.0.0.1 for user achmadadyatma and database named learndb
created successfully.
```

Test - Query to operate CRUD

1. CRUD in sqlalchemy - “persons”
2. CRUD in pyscopg2 - “customers”

[ ]:
```
# 1. Query in sqlalchemy
from sqlalchemy import create_engine

db = engine

#CRUD
# Capital letters are ignored
# C - Create table named "persons" in learndb database
db.execute("CREATE TABLE persons(ID int NOT NULL, LastName varchar(255) NOT␣
 ↪NULL, FirstName varchar(255), Age int, DateOfBirth date, PRIMARY KEY(ID))")
```

[ ]:
```
<sqlalchemy.engine.cursor.LegacyCursorResult at 0x7ff94556a4a0>
```

[ ]:
```
# C - Create 2 records into "persons" table
db.execute("INSERT INTO persons(ID, LastName, FirstName, Age, DateOfBirth)␣
 ↪VALUES ('1', 'Ardi', 'Achmad', '25', '1997-04-05')")

db.execute("INSERT INTO persons(ID, LastName, FirstName, Age, DateOfBirth)␣
 ↪VALUES ('3', 'Wulandari', 'Qonita', '26', '1996-12-13')")
```

[ ]:
```
<sqlalchemy.engine.cursor.LegacyCursorResult at 0x7ff94510ff10>
```

[ ]:
```
#R - Read all records on "persons" table
result_set = db.execute("SELECT * FROM persons")
for r in result_set.fetchall():
    print(r)
```

```
(1, 'Ardi', 'Achmad', 25, datetime.date(1997, 4, 5))
(3, 'Wulandari', 'Qonita', 26, datetime.date(1996, 12, 13))
```

[ ]:
```
#U - Update existing record where ID = 1
db.execute("UPDATE persons SET FirstName='Adyatma' WHERE ID='1'")
```

[ ]:
```
<sqlalchemy.engine.cursor.LegacyCursorResult at 0x7ff954125ff0>
```

[ ]:
```
#D - Delete existing record where ID = 1
db.execute("DELETE FROM persons WHERE ID='1'")
```

[ ]:
```
<sqlalchemy.engine.cursor.LegacyCursorResult at 0x7ff94556ae30>
```

[ ]:
```
# 2. Query in psycopg2
import psycopg2

conn = get_connection_psycopg2()
cursor = conn.cursor()

#CRUD
# C - Create table named "customers" in learndb database
sql = '''
    CREATE TABLE customers(
    ID INT NOT NULL,
    Name VARCHAR (20) NOT NULL,
    Age INT NOT NULL,
    Address CHAR (25),
    Salary DECIMAL (18, 2),
    PRIMARY KEY (ID));
'''
cursor.execute(sql)
print("Table created succesfully")
conn.commit()
```

```
Table created succesfully
```

[ ]:
```
# C - Create 2 records into "customers" table
conn = get_connection_psycopg2()
cursor = conn.cursor()
sql_1 = '''
    INSERT INTO customers(ID, Name, Age, Address, Salary)
    VALUES ('1', 'Laiba', '27', 'Taman ubud kencana 2', '23.4');
'''

sql_2 = '''
    INSERT INTO customers(ID, Name, Age, Address, Salary)
    VALUES ('2', 'Nafisah', '29', 'Saribumi indah 3', '32.1');
'''

cursor.execute(sql_1)
cursor.execute(sql_2)
print("Values created succesfully")
conn.commit()
```

```
Values created succesfully
```

[ ]:
```
#R - Read all records on "customers" table
conn = get_connection_psycopg2()
```

```
cursor = conn.cursor()

sql_3 = '''
    SELECT * FROM customers;
'''

result_set = db.execute(sql_3)
for r in result_set.fetchall():
    print(r)
```

```
(1, 'Laiba', 27, 'Taman ubud kencana 2     ', Decimal('23.40'))
(2, 'Nafisah', 29, 'Saribumi indah 3         ', Decimal('32.10'))
```

[ ]:
```
#U - Update existing record where ID = 1
conn = get_connection_psycopg2()
cursor = conn.cursor()

sql_4 = '''
    UPDATE customers SET Age='25' WHERE ID='1';
'''

cursor.execute(sql_4)
print("Updated value created succesfully")
conn.commit()
```

```
Updated value created succesfully
```

[ ]:
```
#D - delete existing record where ID = 2
conn = get_connection_psycopg2()
cursor = conn.cursor()

sql_4 = '''
    DELETE FROM customers WHERE ID='2';
'''

cursor.execute(sql_4)
print("Updated value created succesfully")
conn.commit()
```

```
Updated value created succesfully
```

Load Merged Data

1. Create Schema Table “pollution”

CREATE TABLE pollution (“Date” DATE, “Country” VARCHAR, “City” VARCHAR, “Count_o3” INT, “Count_pm25” INT);

2. Check available tables in database

3. Show dataframe's data that you want to load
4. Load the dataframe 'merge_df' to database
5. Confirm data has been added to the table

[ ]:
```
#1. Create Schema Table 'pollution'
import psycopg2

conn = get_connection_psycopg2()
cursor = conn.cursor()

sql = '''
    CREATE TABLE pollution(
    "Date" DATE,
    "Country" VARCHAR,
    "City" VARCHAR,
    "Count_o3" INT,
    "Count_pm25" INT);
'''

cursor.execute(sql)
print("Table created succesfully")
conn.commit()
```

```
Table created succesfully
```

[ ]:
```
#2. Check available table in database (make sure the schema table of␣
 ↪'pollution' have been created)
from sqlalchemy import create_engine
from sqlalchemy import inspect

db = engine
insp = inspect(db)
print(insp.get_table_names())
```

```
['persons', 'customers', 'pollution']
```

[ ]:
```
#3. Show dataframe's data that you want to load
print(merge_df.head())
print(merge_df.dtypes)
```

```
        Date Country          City  Count_o3  Count_pm25
0 2017-01-01      US   Los Angeles        24          69
1 2017-01-01      CN      Shanghai       188         215
2 2017-01-01      US     Manhattan        24          96
3 2017-01-01      US  Jacksonville        46         115
4 2017-01-01      CN       Beijing       235         386
Date          datetime64[ns]
Country               object
```

```
City                    object
Count_o3                 int64
Count_pm25               int64
dtype: object
```

[ ]:
```
#4. Load the data to database
merge_df.to_sql(name="pollution", con=db, if_exists='append', index=False)
```

[ ]:
```
100
```

[ ]:
```
#5. Confirm data has been added to the table
print(pd.read_sql_query("SELECT * FROM pollution", con=db).head())
print(pd.read_sql_query("SELECT * FROM pollution", con=db).tail())
```

```
         Date Country          City  Count_o3  Count_pm25
0  2017-01-01      US   Los Angeles        24          69
1  2017-01-01      CN      Shanghai       188         215
2  2017-01-01      US     Manhattan        24          96
3  2017-01-01      US  Jacksonville        46         115
4  2017-01-01      CN       Beijing       235         386
            Date Country          City  Count_o3  Count_pm25
5095  2020-06-30      US   Los Angeles        24          72
5096  2020-06-30      CN         Wuhan       382         384
5097  2020-06-30      US     Manhattan        24          49
5098  2020-06-30      US  Jacksonville       120         141
5099  2020-06-30      CN      Shanghai       828         853
```