```
import pandas as pd
import json
import glob
import mysql.connector
from sqlalchemy import create_engine

#AGGREGATED TRANSACTION


#specify the folder path using '*' pattern
path_1_a = 'C:\\pulse\\data\\aggregated\\transaction\\country\\india\\state\\*\\*\\*.json'
files_1_a = glob.glob(path_1_a)
# declaring a empty list
dfs_1_a = []
#itering over the each file opened from the directory
for file in files_1_a:
    parts = file.split("\\")
    state_name = parts[8]
    year = int(parts[9][:4])
    quarter_number = int(parts[10][0])
    quarter = "Q" + str(quarter_number)
  # opening the json file one by one into python dictionary
    with open(file) as f:
        data = json.load(f)
  # declaring a empty list to capture only the required data for analysing
    transactions_1_a = []
    for transaction in data['data']['transactionData']:
        name = transaction['name']
        payment_instrument = transaction['paymentInstruments'][0]
        count = int(payment_instrument['count'])
        amount = int(payment_instrument['amount'])
        transactions_1_a.append({'country': 'India','state': state_name,'year': year, 'quarter': quarter, 'transaction_type': name,
'transaction_count': count, 'total_amount': amount})
     # converting the list of dictionaries to dataframe
    df_1_a = pd.DataFrame(transactions_1_a)
    # adding converted data frame into new list
    dfs_1_a.append(df_1_a)
#concatenate
df_agg_tran = pd.concat(dfs_1_a)
#index reset
df_agg_tran = df_agg_tran.reset_index(drop=True)
```

```
df_agg_tran.to_csv('C:\\Users\\Shruthy\\Downloads\\agg_trans.csv',index=False)
#print(df_agg_tran)


#AGGREGATED USER


#specify the folder path using '*' pattern
path_1_u = 'C:\\pulse\\data\\aggregated\\user\\country\\india\\state\\*\\*\\*.json'
files_1_u = glob.glob(path_1_u)
# declaring a empty list
dfs_1_u= []
#itering over the each file opened from the directory
for file in files_1_u:
    parts = file.split("\\")
    state_name = parts[8]
    year = int(parts[9][:4])
    quarter_number = int(parts[10][0])
    quarter = "Q" + str(quarter_number)
    # opening the json file one by one into python dictionary
    with open(file) as f:
        data = json.load(f)
    # declaring a empty list to capture only the required data for analysing
    transactions_1_u = []
    # itering through the first file and extracting values of name, type, count, and amount of the payment instrument, and stores them in a
new dictionary.
    for user in data['data']['aggregated']:
        registered = int(data['data']['aggregated']['registeredUsers'])
        app = int(data['data']['aggregated']['appOpens'])
        transactions_1_u.append({'country': 'india','state': state_name,'year': year, 'quarter': quarter,'registered_users':
registered,'apps_opened': app})
    # converting the list of dictionaries to dataframe
    df_1_u = pd.DataFrame(transactions_1_u)
    # adding converted data frame into new list
    dfs_1_u.append(df_1_u)
# concatinate all the list having dataframe of each file in the directory
df_agg_user = pd.concat(dfs_1_u)
# reseting the index
df_agg_user = df_agg_user.reset_index(drop=True)
df_agg_user.to_csv('C:\\Users\\Shruthy\\Downloads\\agg_users.csv',index=False)
#print(df_agg_user)
```

```
#MAP TRANSACTIONS

#specify the folder path using '*' pattern
path_2_a = 'C:\\pulse\\data\\map\\transaction\\hover\\country\\india\\state\\*\\*\\*.json'
files_2_a = glob.glob(path_2_a)
dfs_2_a = []

for file in files_2_a:
    parts = file.split("\\")
    state_name = parts[9]
    year = int(parts[10][:4])
    quarter_number = int(parts[11][0])
    quarter = "Q" + str(quarter_number)

    with open(file) as f:
        data = json.load(f)

    hoverdatas_2_a = []
    for hoverdata in data['data']['hoverDataList']:
            name = hoverdata['name']
            metrics = hoverdata['metric'][0]
            count = int(metrics['count'])
            amount = int(metrics['amount'])
            #A list of dictionaries is created, where each dictionary corresponds to one transaction.
            hoverdatas_2_a.append({'country': 'india','state': state_name,'year': year,'quarter':quarter,'district_name':
name,'transaction_count': count, 'total_amount': amount})
        # converting the list of dictionaries to dataframe
    df_2_a = pd.DataFrame(hoverdatas_2_a)
    dfs_2_a.append(df_2_a)

df_map_tran = pd.concat(dfs_2_a)
df_map_tran = df_map_tran.reset_index(drop=True)
df_map_tran.to_csv('C:\\Users\\Shruthy\\Downloads\\_map_trans.csv',index=False)
#print(df_map_tran)


#MAP USER
```

```
#specify the folder path using '*' pattern
path_2_u = 'C:\\pulse\\data\\map\\user\\hover\\country\\india\\state\\*\\*\\*.json'
files_2_u = glob.glob(path_2_u)
# declaring a empty list
dfs_2_u= []
#itering over the each file opened from the directory
for file in files_2_u:
    parts = file.split("\\")
    state_name = parts[9]
    year = int(parts[10][:4])
    quarter_number = int(parts[11][0])
    quarter = "Q" + str(quarter_number)
    # opening the json file one by one into python dictionary
    with open(file) as f:
        data = json.load(f)
    # declaring a empty list to capture only the required data for analysing
    transactions_2_u = []
    # itering through the first file and extracting values of name, type, count, and amount of the payment instrument, and stores them in a
new dictionary.
    for states, users in data['data']['hoverData'].items():
        state = states
        registered = int(users['registeredUsers'])
        app = int(users['appOpens'])
        transactions_2_u.append({'country': 'india','state': state_name,'year': year,'quarter':quarter,'states':state,'registered_users':
registered,'apps_opened': app})
    # converting the list of dictionaries to dataframe
    df_2_u = pd.DataFrame(transactions_2_u)
    # adding converted data frame into new list
    dfs_2_u.append(df_2_u)
# concatinate all the list having dataframe of each file in the directory
df_map_user = pd.concat(dfs_2_u)
# reseting the index
df_map_user = df_map_user.reset_index(drop=True)
df_map_user.to_csv('C:\\Users\\Shruthy\\Downloads\\map_users.csv',index=False)
#print(df_map_user)

#TOP TRANSACTIONS

#specify the folder path using '*' pattern
path_3_a = 'C:\\pulse\\data\\top\\transaction\\country\\india\\state\\*\\*\\*.json'
files_3_a = glob.glob(path_3_a)
```

```
dfs_3_a=[]
for file in files_3_a:
    parts = file.split("\\")
    state_name = parts[8]
    year = int(parts[9][:4])
    quarter_number = int(parts[10][0])
    quarter = "Q" + str(quarter_number)

    with open(file) as f:
        data = json.load(f)

    entity_list_3_a= []

    for dist in data['data']['districts']:
                name = dist['entityName']
                metrics = dist['metric']
                count = int(metrics['count'])
                amount = int(metrics['amount'])
                #A list of dictionaries is created, where each dictionary corresponds to one transaction.
                entity_list_3_a .append({'country': 'india','state': state_name,'year':
year,'entity_type':'district','quarter':quarter,'district&pincode': name, 'transaction_count': count, 'total_amount': amount})
    for pin in data['data']['pincodes']:
                name = pin['entityName']
                metrics = pin['metric']
                count = int(metrics['count'])
                amount = int(metrics['amount'])
                #A list of dictionaries is created, where each dictionary corresponds to one transaction.
                entity_list_3_a .append({'country': 'india','state': state_name,'year':
year,'entity_type':'pincode','quarter':quarter,'district&pincode': name, 'transaction_count': count, 'total_amount': amount,})
    # converting the list of dictionaries to dataframe
    #
    df_3_a = pd.DataFrame(entity_list_3_a)
    dfs_3_a.append(df_3_a)

df_top_agg = pd.concat(dfs_3_a)
df_top_agg  = df_top_agg .reset_index(drop=True)
df_top_agg.to_csv('C:\\Users\\Shruthy\\Downloads\\top_trans.csv',index=False)
#print(df_top_agg)

#TOP USER
```

```
#specify the folder path using '*' pattern
path_3_u = 'C:\\pulse\\data\\top\\user\\country\\india\\state\\*\\*\\*.json'
files_3_u = glob.glob(path_3_u)
# declaring a empty list
dfs_3_u = []
#itering over the each file opened from the directory
for file in files_3_u:
    parts = file.split("\\")
    state_name = parts[8]
    year = int(parts[9][:4])
    quarter_number = int(parts[10][0])
    quarter = "Q" + str(quarter_number)
    # opening the json file one by one into python dictionary
    with open(file) as f:
        data = json.load(f)
    # declaring a empty list to capture only the required data for analysing
    transactions_3_u = []
    # itering through the first file and extracting values of name, type, count, and amount of the payment instrument, and stores them in a
new dictionary.
    # for states in data['data']['states']:
    #     state = states['name']
    #     registered = int(states['registeredUsers'])
    #     transactions.append({'state_district_pin':state,'registered_users': registered,'entity_type':'state'})
    for dist in data['data']['districts']:
        district = dist['name']
        registered = int(dist['registeredUsers'])
        transactions_3_u.append({'country': 'india','state': state_name,'year':
year,'quarter':quarter,'entity_type':'district','district&pin':district,'registered_users': registered})
    for pin in data['data']['pincodes']:
        pincode = pin['name']
        registered = int(pin['registeredUsers'])
        transactions_3_u.append({'country': 'india','state': state_name,'year':
year,'quarter':quarter,'entity_type':'pincode','district&pin':pincode,'registered_users': registered,})
    # converting the list of dictionaries to dataframe
    df_3_u = pd.DataFrame(transactions_3_u)
    # adding converted data frame into new list
    dfs_3_u.append(df_3_u)
# concatinate all the list having dataframe of each file in the directory
df_top_user = pd.concat(dfs_3_u)
# reseting the index
df_top_user = df_top_user.reset_index(drop=True)
df_top_user.to_csv('C:\\Users\\Shruthy\\Downloads\\top_users.csv',index=False)
```

```
#print(df_top_user)



# using create_engine module opening the MySql with correct credentials
engine = create_engine('mysql+mysqlconnector://root:Shruthy#123@127.0.0.1:3306/pulse')

config = {
  'user': 'root',
  'password': 'Shruthy#123',
  'host': '127.0.0.1',
  'database': 'pulse',
  'raise_on_warnings': True
}

# Connect to the database
cnx = mysql.connector.connect(**config)

# Check if the connection is successful
if cnx.is_connected():
  print("Connection to MySQL database established.")
else:
  print("Connection to MySQL database failed.")


  # create a table name and store the dataframe-1
df_agg_tran.to_sql(name='aggregate_transaction', con=engine, if_exists='replace', index=False)
# create a table name and store the dataframe-2
df_agg_user.to_sql(name='aggregate_users', con=engine, if_exists='replace', index=False)
# create a table name and store the dataframe-3
df_map_tran.to_sql(name='map_transcation', con=engine, if_exists='replace',index=False)
# create a table name and store the dataframe-4
df_map_user.to_sql(name='map_users', con=engine, if_exists='replace', index=False)
# create a table name and store the dataframe-5
df_top_agg.to_sql(name='top_transaction', con=engine, if_exists='replace', index=False)
 #create a table name and store the dataframe-6
df_top_user.to_sql(name='top_users', con=engine, if_exists='replace', index=False)



# # Create a cursor to execute SQL queries
```

```
cursor = cnx.cursor()

# Define the SQL query to retrieve data from the "table
query = "SELECT * FROM aggregate_transaction"
# Execute the SQL query and store the result in a Pandas dataframe
aggregate_transaction = pd.read_sql(query, cnx)
# Print the first 5 rows of the dataframe
print(aggregate_transaction.head())


# SQL query to retrieve data from the "table
query = "SELECT * FROM aggregate_users"
# executing the  SQL query and store the result in a Pandas dataframe
aggregate_users = pd.read_sql(query, cnx)
# Print the first 5 rows of the dataframe
print(aggregate_users.head())



# SQL query to retrieve data from the "table
query = "SELECT * FROM map_transaction"
# executing the  SQL query and store the result in a Pandas dataframe
map_transaction = pd.read_sql(query, cnx)
# Print the first 5 rows of the dataframe
print(map_transaction.head())


# SQL query to retrieve data from the "table
query = "SELECT * FROM map_users"
# executing the  SQL query and store the result in a Pandas dataframe
map_users = pd.read_sql(query, cnx)
# Print the first 5 rows of the dataframe
print(map_users.head())



# SQL query to retrieve data from the "table
query = "SELECT * FROM top_transaction"
# executing the  SQL query and store the result in a Pandas dataframe
top_transaction = pd.read_sql(query, cnx)
# Print the first 5 rows of the dataframe
```

```
print(top_transaction.head())


# SQL query to retrieve data from the "table
query = "SELECT * FROM top_users"
# executing the  SQL query and store the result in a Pandas dataframe
top_users = pd.read_sql(query, cnx)
# Print the first 5 rows of the dataframe
print(top_users.head())
```