Pyspark Peer Review

Shishir's Approach

In dataframe.py:

1. It imports the required libraries such as-
   requests, json, pprint, pyspark, SparkSession, StructType, StructField, StringType, and IntegerType.

2. It creates a SparkSession object using PySpark, which is used to interact with Spark for distributed data processing.

3. It retrieves data from the COVID-19 API by making a GET request to the API with the required headers for authentication. The response is in JSON format.

4. The retrieved JSON data is cleaned by removing unnecessary information, converting data types, and trimming the state names.

5. A PySpark DataFrame is created from the cleaned data by defining the schema of the DataFrame using StructType, StructField, StringType, and IntegerType.

   The DataFrame is created using the **spark.createDataFrame()** method with the defined schema and the cleaned data.

6. Finally, the main part of the code calls the functions in the following order:
   - **getData()** to fetch COVID-19 data from the API and store it in jsonResponse.
   - **cleanData()** to clean the retrieved JSON data and store it in data
   - **createDf()** to create a PySpark DataFrame from the cleaned data and store it in df.

In app.py:

In app.py created a Flask app and inside it created the routes for all the queries used in demo.py and retured the output in json format.

Ashwat's Approach

In dataframe.py:

1. getDataFromUrl(): This function sends a GET request to an API endpoint to fetch COVID-19 data in India. The response is returned as a JSON object.

2. trimCheck(): This function takes a state name as input and removes the asterisk character from it, if present. It is used to clean up some state names that have an asterisk in them.

3. cleanData(): This function takes the JSON object returned by getDataFromUrl() and processes it into a list of lists. Each inner list contains data for a single state: the serial number, state name, number of confirmed cases, number of cured cases, number of deaths, and total number of cases.

4. getDataFrame(): This function takes the list of lists returned by cleanData() and converts it into a PySpark DataFrame. It creates a PySpark RDD from the list of lists, converts each inner list into a Row object, defines a schema for the DataFrame using StructType, and uses spark.createDataFrame() to create the DataFrame.

In app.py:

1. Importing dataframe.py: This file is imported to use the functions defined in it.

2. Writing queries: Various queries are written to fetch the required data from the PySpark DataFrame created in dataframe.py. These queries include sorting the data by different columns, filtering data based on certain conditions, and aggregating data.

3. Creating endpoints: Endpoints are created using Flask to display the results of the queries. These endpoints can be accessed via a web browser or an API client.