**Day 18 Assignment - Vamsi Viswanadham**

**Date: 23/12/2023**

Worked on the below concepts

DataFrames are a crucial component of data manipulation and analysis in Python, especially with libraries like PySpark, Dask, and Pandas. Here's an overview of various operations and techniques you can use with DataFrames in these libraries:

**Creating DataFrames in PySpark and Dask:**

- PySpark: Typically used for handling large datasets, you create DataFrames in PySpark by reading data from various sources like CSV, JSON, or databases.

- Dask: Similar to PySpark, Dask is used for parallel computing. You can create DataFrames in Dask by reading data from different file formats or converting existing pandas DataFrames to Dask DataFrames.

**Looking at Data in DataFrames:**

- This involves viewing the data structure, previewing the first few rows, and understanding the basic statistics of the DataFrame.

**Selecting, Renaming, and Filtering Data in a Pandas DataFrame:**

- Selecting specific columns, renaming columns for clarity, and filtering rows based on certain conditions.

**Manipulating, Dropping, Sorting, Aggregations, Joining, GroupBy in DataFrames:**

- Includes operations like modifying data, removing unnecessary columns, organizing data, performing aggregate functions, merging multiple DataFrames, and grouping data for analysis.

Overview of basic functions and operations in PySpark, particularly focusing on DataFrames. Here's a summary of the key points:

**Reading Data:** Data can be loaded using spark.read.load. It supports various formats like parquet and CSV. An example is shown where a CSV file containing coronavirus cases is loaded.

**Viewing Data:** DataFrames can be examined using .show() or converted to Pandas format for easier visualization with .toPandas().

**Renaming Columns:** Column names in Spark DataFrames can be changed using withColumnRenamed for individual columns or toDF for renaming all columns.

**Selecting Columns:** A subset of columns can be selected using the select keyword.

**Sorting Data:** Data can be sorted using the sort method. For descending order, F.desc from PySpark SQL functions is used.

**Type Casting:** The cast function is useful for converting data types, like converting a column to an integer or string type.

**Filtering Data:** DataFrames can be filtered using conditions combined with logical operators like AND (&), OR (|), and NOT (~).

**Grouping and Aggregation:** The groupBy function, combined with aggregation functions from pyspark.sql.functions, is used for grouping and aggregating data. Column names in the aggregation result can be renamed using the alias keyword.

**Joining DataFrames:** DataFrames can be joined using the join method. An example is given where two CSV files are joined.

**Broadcast/Map Side Joins:** For joining a large table with a small one, the broadcast keyword optimizes the join by broadcasting the smaller table to each node.