Experiment - 9

Aim: To perform Exploratory data analysis using Apache Spark and Pandas

Theory:

1) **What is Apache Spark and it works?**

Ans:

Apache Spark is an open-source, distributed computing system designed for big data processing and analytics. It's known for its speed, scalability, and ease of use, especially for large-scale data processing tasks like machine learning, stream processing, and SQL queries.

Language Support: Scala (native), Java, Python (PySpark), R, and SQL.

At a high level, Spark works in four main stages:

1. Driver Program Starts
The user writes an application using Spark APIs (in Python, Scala, etc.).
The Driver Program is the heart of Spark, where the main control flow runs.
It converts user code into DAGs (Directed Acyclic Graphs) of stages.

2. Cluster Manager Allocates Resources
Spark uses a Cluster Manager (e.g., YARN, Mesos, or its own standalone manager) to allocate resources.
Executors (worker nodes) are launched across the cluster.

3. Tasks Are Sent to Executors
The DAG Scheduler breaks the job into tasks.
These tasks are sent to the Executors, which actually do the work like reading data, running transformations, and writing output.

4. Executors Run Tasks and Return Results
Executors process data in memory for fast performance.
Intermediate results are cached when possible.
Final results are returned to the Driver or written to storage.

Use Cases

Processing logs or real-time event data.
Running ML models on large datasets.
ETL (Extract, Transform, Load) pipelines.
Analyzing huge volumes of structured or unstructured data.

2) How data exploration done in Apache spark? Explain steps.

Ans:

Data exploration in **Apache Spark**—especially using **PySpark** (Python API for Spark)—is a crucial step in any big data analytics or machine learning pipeline. It helps you understand the structure, quality, and patterns in your dataset.

Steps for Data Exploration in Apache Spark:

1. Loading the Data

The first step in data exploration is importing the data into Spark from sources such as CSV files, JSON, Parquet, databases, or cloud storage systems. Apache Spark provides various APIs to connect to and load data from different formats and sources efficiently.

2. Understanding the Schema

Once the data is loaded, the schema (structure) of the dataset is examined. This includes:

- Column names
- Data types (e.g., Integer, String, Date)
- Nullable fields

Understanding the schema helps identify incorrect or inconsistent data types and ensures the data is correctly interpreted.

3. Viewing Sample Records

Exploratory analysis begins with viewing a few records from the dataset. This helps in gaining a quick overview of:

- The kind of data stored
- Formatting or entry errors
- Presence of special characters or missing values

4. Generating Summary Statistics

Statistical summaries of numerical columns are generated to understand the distribution and spread of the data. These statistics include:

- Count
- Mean
- Minimum and Maximum values

- Standard deviation

This step is essential for detecting outliers and understanding the scale of data.

5. Identifying Missing or Null Values

It is important to detect and assess missing or null values in the dataset. This helps in deciding whether to remove, replace, or impute missing data before further analysis.

6. Analyzing Value Distributions

The distribution of values within categorical or numerical columns is analyzed. This includes grouping data based on categories and counting their occurrences. It helps to:
Identify class imbalances
Understand the frequency of certain values

7. Examining Relationships and Correlations

In this step, the relationships between numerical variables are explored. Correlation analysis is performed to measure how strongly two variables are related. This insight is useful for feature selection in machine learning.

8. Filtering and Conditional Queries

Subsets of data are explored by applying filters and conditions. This helps focus on specific segments or conditions, such as high-income individuals or records with specific attributes.

9. Sampling for Detailed Analysis

If the dataset is very large, a smaller representative sample is taken for detailed analysis or visualization. This reduces computational cost and allows easier inspection.

10. Preparing for Visualization or Modeling

Although Spark itself is not primarily used for visualizations, after exploration, data is often summarized or exported for plotting using external tools. The insights gained during exploration guide the next steps in data cleaning, transformation, or modeling.

Conclusions:

Apache Spark is a fast and general-purpose distributed computing system designed for large-scale data processing. It utilizes a driver-executor architecture and performs in-memory computations to achieve high performance. Spark supports various APIs, including RDDs, DataFrames, and Datasets, enabling efficient data manipulation and processing across multiple programming languages.