PySpark Basics

Business Overview

Apache Spark is a distributed processing engine that is open source and used for large data applications. For quick analytic queries against any quantity of data, it uses in-memory caching and efficient query execution. It offers code reuse across many workloads such as batch processing, interactive queries, real-time analytics, machine learning, and graph processing. It provides development APIs in Java, Scala, Python, and R.

Data Pipeline:

A data pipeline is a technique for transferring data from one system to another. The data may or may not be updated, and it may be handled in real-time (or streaming) rather than in batches. The data pipeline encompasses everything from harvesting or acquiring data using various methods to storing raw data, cleaning, validating, and transforming data into a query-worthy format, displaying KPIs, and managing the above process.

Tech stack:
➔Language: Python
➔Package: Pyspark

PySpark:

PySpark is a Python interface for Apache Spark. It not only lets you develop Spark applications using Python APIs, but it also includes the PySpark shell for interactively examining data in a distributed context. PySpark supports most of Spark's capabilities, including Spark SQL, DataFrame, Streaming, MLlib, and Spark Core. In this project, you will learn about core Spark architecture, Spark Sessions, Transformation, Actions, and Optimization Techniques using PySpark.

Key Takeaways:
- Understanding the project overview
- Introduction to PySpark
- Understanding Spark Architecture and Lifecycle
- Introduction to Spark Operations
- Understanding the components of Spark Apache
- Understanding Resilient Distributed Data (RDD)
- Difference between Transformation and Action
- Understanding Interactive Spark Shell

- Understanding the concept of Directed Acyclic Graph(DAG)
- Features of Spark
- Applications of Spark