Big Data processing with Pyspark and Apache Airflow

Module 1: Introduction to ETL with PySpark

- Overview of ETL (Extract, Transform, Load)
- Advantages of using PySpark for ETL processes
- Key components of a PySpark ETL workflow

Module 2: File Transformations in PySpark

- Reading and writing various file formats:
    - CSV
    - JSON
    - Parquet
    - Avro
- Handling nested and semi-structured data
- File partitioning strategies for large datasets
- Best practices for schema management during file transformations

Module 3: DataFrame Handling in PySpark

- Introduction to DataFrames in PySpark
- Creating and initializing DataFrames from:
    - Structured data files
    - RDDs
    - Databases
- Common DataFrame operations:
    - Selecting, filtering, and sorting data
    - Aggregations and group-by operations
    - Joins: Inner, outer, left, right, and cross joins
- Managing schema evolution and schema enforcement

Module 4: Advanced DataFrame Operations

- Window functions and their use cases
- Handling nulls and missing data in DataFrames
- Writing efficient UDFs (User Defined Functions)
- Broadcasting variables and its implications in distributed systems
- Optimizing PySpark jobs using Catalyst Optimizer

Module 5: Working with Tables in PySpark

- Reading and writing tables in Hive or Delta Lake
- Performing CRUD operations on tables
- Partitioning and bucketing tables
- Managing table metadata and catalogs
- Using PySpark SQL for complex table queries

Module 6: Creating DAGS in Airflow to Orchestrate PySpark Jobs

- Overview of Apache Airflow and its architecture
- Key concepts: DAGs, Operators, and Tasks
- Setting up Airflow to run PySpark jobs:
    - Creating custom operators for PySpark
    - Configuring connections and variables in Airflow
- Designing robust workflows with Airflow:
    - Scheduling and triggering DAGs
    - Setting task dependencies and execution sequences
    - Monitoring and debugging failed DAG runs
- Best practices for scaling Airflow workflows

Module 7: Orchestration of Multiple PySpark Jobs

- Strategies for dividing ETL processes into modular PySpark jobs

- Chaining PySpark jobs using Airflow:
    - Passing data and metadata between jobs
    - Managing interdependencies and avoiding race conditions
- Handling retries and fault tolerance in job orchestration
- Logging and tracking progress across multiple jobs

Module 8: Building an end-to-end pipeline

- Implementing an end-to-end ETL pipeline using PySpark and Airflow:
    - Data extraction from multiple file formats
    - Transformations involving schema standardization and data cleansing
    - Loading processed data into a database or data lake
    - Orchestrating the pipeline with Airflow DAGs