**Optimizing Healthcare Management: A Comprehensive Data Engineering Solution for the COVID-19 Pandemic**

**A Data Engineering Case Study**

**Stream : Spark Azure track**

**Curated by : Anandh Kumar M**

**Technology :**

**PySpark,Spark Streaming ,Azure SQL ,Git**

**Problem Statement:**

In response to the urgent need for improved healthcare management brought about by the recent Covid-19 pandemic, a critical aspect that often goes overlooked is the prediction and observation of patient length of stay (LOS) in hospitals. Efficient healthcare management, an area with multifaceted applications for data science, can be significantly enhanced by focusing on LOS as a key parameter.

This parameter becomes instrumental in identifying patients with a high LOS risk, enabling hospitals to optimize treatment plans upon admission, thus minimizing LOS and reducing the risk of infections among staff and visitors. Moreover, the predictive understanding of LOS facilitates strategic logistics planning, including the allocation of rooms and beds within the hospital. In the context of this imperative, envision yourself as the Data Scientist at HealthMan, a non-profit organization committed to proficiently managing hospital operations.

Tasked as a Big Data consultant, your role involves leveraging Spark to explore and load datasets into Spark DataFrames, merging these data frames based on common keys, and creating a unified DataFrame. The subsequent step entails meticulous examination for missing data. The ultimate objective is to develop precise predictions for the length of stay on a case-by-case basis, empowering hospitals with information crucial for optimal resource allocation and enhanced operational efficiency. The length of stay is categorized into 11 classes, ranging from 0-10 days to more than 100 days, utilizing data from the COVID-19 Data Repository by the Center for Systems Science and Engineering (CSSE) at Johns Hopkins University.

**Project Overview:**

For the data engineering project based on the provided problem statement, the expectations include developing a robust data ingestion process using Spark to fetch COVID-19 datasets from Johns Hopkins University. The project involves integrating and transforming the data, implementing data quality checks, and handling missing data. Efficient storage strategies, scalability considerations, and workflow orchestration are crucial aspects, with documentation ensuring clarity in data structures and processes. Collaboration with the data science team is essential to meet their requirements for model development. Monitoring, logging, security measures, and compliance with data governance are integral for a secure and reliable data engineering solution. The final deliverable should be a well-documented, scalable infrastructure that supports accurate length of stay predictions and facilitates seamless collaboration between data engineering and data science teams.

**Datasets:**

**Batch Processing**

[**https://github.com/akgeoinsys/ltimindtree2023/tree/main/casestudies/datasets/healthcare/worldometer\_data.csv**](https://github.com/akgeoinsys/ltimindtree2023/blob/main/casestudies/datasets/healthcare/worldometer_data.csv)

**Spark Streaming Datasets: Latest**

[**https://github.com/akgeoinsys/ltimindtree2023/tree/main/casestudies/datasets/healthcare/country\_wise\_latest.csv**](https://github.com/akgeoinsys/ltimindtree2023/blob/main/casestudies/datasets/healthcare/country_wise_latest.csv)

**Key Objectives:**

**Detailed Data Engineering Tasks for COVID-19 Datasets Using PySpark, Spark Streaming, ADLS, and Azure Data Factory:**

**Data Ingestion:**

Develop a PySpark-based data ingestion process to retrieve the worldwide COVID-19 datasets from the specified source (CSSE at Johns Hopkins University) with considerations for efficient and distributed data loading.

Implement Spark Streaming to continuously fetch and process the latest COVID-19 data in real-time.

**Data Integration and Transformation:**

Perform data integration by merging the historical worldwide COVID-19 datasets and the real-time Spark Streaming data based on a common key, ensuring comprehensive coverage.

Execute necessary data transformations to standardize schemas, handle outliers, and derive relevant features for predictive modeling.

**Data Quality and Missing Data Handling:**

Implement robust data quality checks within the PySpark framework to identify and rectify any anomalies or inconsistencies in the historical and real-time datasets.

Develop advanced strategies for handling missing data, ensuring that imputation techniques maintain the integrity of the analysis and predictions.

**Data Storage and Management:**

Design an efficient data storage strategy by leveraging Azure Data Lake Storage (ADLS) to store both historical and real-time COVID-19 datasets. Optimize storage considerations such as partitioning, file formats (e.g., Parquet), and compression for enhanced performance and cost-effectiveness.

Organize data in ADLS with a structured hierarchy that facilitates easy accessibility and retrieval.

**Scalability and Performance:**

Ensure scalability of the PySpark data engineering processes to seamlessly handle the increasing volume of historical and real-time COVID-19 data.

Fine-tune Spark configurations and employ effective partitioning strategies to optimize processing performance.

**Workflow Orchestration**:

11. Develop a comprehensive workflow orchestration system using Azure Data Factory to coordinate and automate the execution of data engineering tasks. This includes scheduling both batch processing for historical data and Spark Streaming for real-time updates.

**Case Study Execution Plan:**

**Team Structure**: A group of 4 or 5 members will execute the case study.

**Task Assignment:** Each member will have specific tasks aligned with project objectives.

**Concurrent Work**: Team members will work concurrently, ensuring parallel progress.

**Integration:** Individual contributions will integrate during the final project stage.

**Final Presentatio**n: Completed case study will be presented to SMEs and Mentors.

**Task Breakdown:**

**Data Loading and Exploration:**

Load the COVID-19 dataset into a PySpark DataFrame.

Explore the dataset to understand its structure, data types, and any missing values.

Identify key statistical measures for relevant columns (e.g., TotalCases, TotalDeaths).

**Data Transformation:**

Use PySpark to perform necessary data transformations.

Create additional columns for metrics such as Mortality Rate (TotalDeaths/TotalCases) and Recovery Rate (TotalRecovered/TotalCases).

Explore any relevant patterns or trends in the COVID-19 data.

**ADLS Integration:**

Save the transformed DataFrame into a new CSV file.

Store the CSV file in Azure Data Lake Storage (ADLS).

Ensure proper organization within ADLS, considering folders and file naming conventions.

**Advanced Analysis (Optional):**

(Optional) Perform more advanced analysis, such as time-series analysis or regional comparisons.

Explore factors contributing to variations in COVID-19 metrics.

**Documentation:**

Document the steps taken for data loading, transformation, and ADLS integration.

Include observations from exploratory analysis and any insights gained.

Clearly outline any challenges faced during the process.

**Azure Data Factory Workflow:**

Create an Azure Data Factory pipeline to orchestrate the entire data analysis process.

Set up Linked Services for connecting to ADLS.

Define datasets for input (raw COVID-19 data) and output (analyzed data).

Utilize a Data Flow activity to execute the PySpark script.

**Automation and Scheduling:**

Schedule the Azure Data Factory pipeline to run at specified intervals.

Consider optimal scheduling based on the frequency of data updates and analysis requirements.

**Presentation of Findings:**

Prepare a summary presentation or report showcasing key findings from the COVID-19 data analysis.

Highlight any actionable insights for public health decision-making.

**Enhanced Data Engineering Approach with Additional Tasks(Optional):**

Data Ingestion and Cleansing:

**PowerBI Report Creation:**

Participants can create simple PowerBI reports showcasing key metrics during data ingestion.

Visualize the data quality metrics, such as the count of missing values or duplicate records.

**Machine Learning for Data Quality:**

Implement a basic machine learning model using PySpark to identify and handle missing values more intelligently.

Leverage PySpark's MLlib for this task and integrate it into the data ingestion pipeline.