Data Processing Framework Documentation

Submitted by : Ravi Kumbar (senior data engineer , ravi.kumbar.de@gmail.com)

This is just basic attempt to create this framework lot of things can be improved.

Overview

This document provides comprehensive documentation for a data processing framework implemented using Apache Spark. The framework is designed to handle data ingestion, transformation, correlation, and storage in an optimized and fault-tolerant manner. It includes functionalities for reading data from various sources, processing it, writing to storage, and auditing the operations.

Components

The framework consists of the following main components:

1. **Data Ingestion**: Functions for reading data from different sources such as JSON, CSV, and Avro files, and ingesting it into the Spark environment for further processing.
2. **Data Transformation**: Functions for converting data types, standardizing values, and performing necessary transformations on the ingested data.
3. **Data Correlation**: Functions for correlating different datasets, such as correlating ad impressions with clicks and conversions to provide meaningful insights.
4. **Data Storage**: Functions for writing processed data to storage systems, including Amazon S3, in optimized formats such as Parquet.
5. **Audit Logging**: Mechanism for logging audit information, including operations performed, record counts, and timestamps, to enable monitoring and tracking of data processing activities.
6. **Error Handling and Notification**: Handling errors gracefully and sending email notifications in case of job failures to alert stakeholders.
7. **Integration with Redshift**: Integration with Amazon Redshift for creating external schema tables and reading data with proper access control for analytical queries.
8. **Monitoring and Reporting**: Creation of a Power BI dashboard for monitoring the audit layer, refreshing it periodically, and setting up alerts for early morning checks.

Functions

1. Data Ingestion

- `read_json_data`: Reads JSON data from the specified input path using the given schema.
- `read_csv_data`: Reads CSV data from the specified input path.
- `read_avro_data`: Reads Avro data from the specified input path.

2. Data Transformation

- `convert_data_types`: Converts data types to appropriate types.
- `standardize_data`: Standardizes the data by modifying column values.
- `transform_data`: Applies transformations to the data.

3. Data Correlation

- `correlate_data`: Correlates ad impressions with clicks and conversions to provide meaningful insights.

4. Data Storage

- `write_data_to_s3`: Writes DataFrame to Amazon S3 in Parquet format.

5. Audit Logging

- `write_audit_data`: Writes audit information to the audit layer in S3.

6. Error Handling and Notification

- `send_email_notification`: Sends email notification in case of failures.

7. Integration with Redshift

- *Functionality*: Creates external schema tables on Redshift and reads data with proper access control for analytical queries.

8. Monitoring and Reporting

- *Functionality*: Creates a Power BI dashboard for monitoring the audit layer, refreshes it periodically, and sets up alerts for early morning checks.

Workflow

The typical workflow of the data processing framework involves the following steps:

1. **Data Ingestion**: Read data from various sources such as JSON, CSV, or Avro files.
2. **Data Transformation**: Convert data types, standardize values, and apply necessary transformations.
3. **Data Correlation**: Correlate different datasets to derive meaningful insights.
4. **Data Storage**: Write processed data to optimized storage systems such as Amazon S3 in Parquet format.
5. **Audit Logging**: Log audit information for monitoring and tracking data processing activities.

6. **Error Handling and Notification**: Handle errors gracefully and send email notifications in case of failures.
7. **Integration with Redshift**: Integrate with Amazon Redshift for analytical queries and access control.
8. **Monitoring and Reporting**: Create a Power BI dashboard for monitoring the audit layer, refresh it periodically, and set up alerts for early morning checks.

Usage

Users can utilize the functions provided by the framework to implement end-to-end data processing pipelines in Spark, ensuring efficient ingestion, transformation, correlation, and storage of data. Additionally, they can integrate with Amazon Redshift for analytical queries and Power BI for monitoring and reporting.

Conclusion

The data processing framework offers a robust solution for handling large-scale data processing tasks in Spark environments. With functionalities for ingestion, transformation, correlation, storage, audit logging, error handling, integration with Redshift, and monitoring/reporting, it provides a comprehensive toolkit for building scalable and fault-tolerant data pipelines.

This documentation provides a detailed overview of the data processing framework, covering its components, functions, workflow, usage instructions, and conclusion. It serves as a comprehensive guide for developers and data engineers to understand, implement, and utilize the framework effectively for their data processing needs.