Open Table Format - Summary Document

Introduction

This document summarizes the steps taken to explore Open Table Formats using Databricks, Delta Lake, and PySpark. The process includes data ingestion, transformation, storage optimization, and querying techniques in Databricks.

Topics Covered

1. Open Table Formats Overview

- Explanation of open table formats like Delta Lake, Apache Iceberg, and Hudi.
- Importance of these formats in data engineering and analytics.

2. Databricks and Spark Cluster Setup

- Overview of Databricks.
- Configuring a Spark Cluster in Databricks.
- Setting up the development environment for PySpark.

3. Data Ingestion into Databricks using PySpark

- Loading CSV files into a Spark DataFrame:

```
df = spark.read.format("csv") \
   .option("header", True) \
   .option("inferSchema", True) \
   .load("/FileStore/tables/sales_data_first.csv")
```

- Displaying the data using:

```
df.show()
```

4. Delta Lake Explained

- Understanding how Delta Lake improves data reliability and performance.
- Delta Lake as an open-source storage layer that enables ACID transactions.

5. Working with Delta Tables

- Writing DataFrame as Delta Table:

```
df.write.format("delta") \
```

.mode("overwrite") \

.option("path", "/FileStore/tables/sinkdata/sales_data_first_delta") \

.save()

- Querying the Delta Table using Spark SQL:

SELECT * FROM delta.`/FileStore/tables/sinkdata/sales_data_first_delta`

6. Delta Log and Transactions

- Explanation of how Delta Log records changes.
- Understanding how Delta maintains consistency and rollback capabilities.

7. Time Travel & Data Versioning in Delta Lake

- Querying previous versions of data:

SELECT * FROM delta.`/FileStore/tables/sinkdata/sales_data_first_delta` VERSION AS OF 2;

- Restoring data to a previous state using:

RESTORE TABLE delta.`/FileStore/tables/sinkdata/sales_data_first_delta` TO VERSION AS OF 2;

8. VACUUM in Delta Lake

- Cleaning up old snapshots to free up storage:

VACUUM delta.`/FileStore/tables/sinkdata/sales_data_first_delta` RETAIN 0 HOURS;

9. Schema Changes in Delta Lake

- Modifying schemas in Delta tables without breaking existing data.

10. Spark Optimization Techniques in Delta Lake

- Caching, indexing, and partitioning strategies for faster queries.

11. Streaming with Delta Lake

- Using Delta tables as a source and sink for Spark Structured Streaming.

Conclusion

This document provides an overview of implementing Open Table Formats using Databricks and Delta Lake. The steps covered include data ingestion, storage optimization, schema evolution, and transaction management in Delta Lake.

For more details, refer to the attached Python scripts and Jupyter notebooks.