Incremental Data Ingestion from Files

Learning Objectives

- What is incremental data Ingestion from file
- COPY INTO
- Auto Loader

Incremental Data Ingestion

- Loading new data files encountered since the last ingestion
- Reduces redundant processing
- 2 mechanisms:
  - COPY INTO
  - Auto loader

COPY INTO

- SQL command
- Idempotently and incrementally load new data files
  - Files that have already been loaded are skipped.

COPY INTO

- **COPY INTO** my_table
  FROM '/path/to/files'
  FILEFORMAT = <format>
  FORMAT_OPTIONS (<format options>)
  COPY_OPTIONS (<copy options>);

Example

- **COPY INTO** my_table

  FROM '/path/to/files’

  FILEFORMAT = CSV

  FORMAT_OPTIONS ('delimiter' = ' | ’,

  'header' = 'true')

  COPY_OPTIONS ('mergeSchema' = 'true’)

Auto loader

- Structured Streaming
- Can process billions of files
- Support near real-time ingestion of millions of files per hour.

Auto loader Checkpointing

- Store metadata of the discovered files
- Exactly-once guarantees
- Fault tolerance

Auto Loader in PySpark API

```
spark.readStream
        .format("cloudFiles")
        .option("cloudFiles.format", <source_format>)
        .load('/path/to/files')
    .writeStream
        .option("checkpointLocation", <checkpoint_directory>)
        .table(<table_name>)
```

Auto Loader + Schema

```
spark.readStream
        .format("cloudFiles")
        .option("cloudFiles.format", <source_format>)
        .option("cloudFiles.schemaLocation", <schema_directory>)
        .load('/path/to/files')
    .writeStream
        .option("checkpointLocation", <checkpoint_directory>)
        .option("mergeSchema", "true")
        .table(<table_name>)
```

COPY INTO vs. Auto Loader

COPY INTO

- Thousands of files
- Less efficient at scale

Auto Loader

- Millions of files
- Efficient at scale