**📈 Currency Exchange Rate ETL Project with Azure Databricks & Fixer API**

This repository contains a complete ETL (Extract, Transform, Load) pipeline built on **Azure Databricks**. The pipeline extracts historical currency exchange rates from the **Fixer API**, transforms the data through bronze, silver, and gold layers using structured streaming, and stores results as Delta tables ready for analytics.

**📝 Overview of Project Structure**

The ETL pipeline consists of four clear data layers:

| **Layer** | **Description** | **Output Location (Delta Table)** |
| --- | --- | --- |
| Raw | Fetch raw JSON data from Fixer API | /mnt/raw\_data (JSON files) |
| Bronze | Ingest raw JSON into Delta Lake | fixer\_bronze |
| Silver | Flatten & clean data into structured form | fixer\_silver |
| Gold | Pivot currency data into analytic format | fixer\_gold |

**🚀 Technologies Used**

* **Azure Databricks** for scalable Spark environment
* **PySpark Structured Streaming** for incremental ingestion
* **Delta Lake** for efficient data storage and management
* **Fixer API** for currency exchange rate data
* **Python** for ETL pipeline scripting

**⚙️ ETL Pipeline Steps (Batch + Streaming)**

**✅ Step 1: Raw Data Extraction (Batch API Fetch)**

* Fetch currency exchange rates weekly (7 days of historical data each time).
* Store the data as JSON files in the raw data layer (/mnt/raw\_data).

**✅ Step 2: Bronze Layer Ingestion (Structured Streaming)**

* Ingest raw JSON data incrementally using Spark Structured Streaming into a bronze Delta table (fixer\_bronze).
* Schema enforced for reliability.

**✅ Step 3: Silver Layer Transformation (Structured Streaming)**

* Flatten JSON structure, extracting and exploding currency rates.
* Save results in the structured Delta table (fixer\_silver).

**✅ Step 4: Gold Layer Analytics (Structured Streaming with Delta MERGE)**

* Pivot data into clear analytical columns per currency.
* Uses watermarking and foreachBatch for safe and incremental upserts.
* Final structured analytical table (fixer\_gold).

**📂 Project Directory Structure**

pgsql

CopyEdit

/mnt/raw\_data/ # Raw JSON data storage and checkpoints

├── checkpoint/

│ ├── bronze\_checkpoint/

│ ├── silver\_checkpoint/

│ └── gold\_checkpoint/

└── delta/

├── fixer\_silver/ # Silver table Delta location

└── fixer\_gold/ # Gold table Delta location

**🛠 Setting Up the Project**

1. **Prerequisites:**
   * Azure Databricks Workspace.
   * Mounted Databricks storage (/mnt/raw\_data).
   * Fixer API subscription key.
2. **Create Databricks widgets:**
   * Add a widget named "api\_key" with your Fixer API key.

python

CopyEdit

dbutils.widgets.text("api\_key", "<your-fixer-api-key>")

1. **Initialize Delta Tables (Run once initially):**

python

CopyEdit

# Initialize Bronze, Silver, Gold tables once

spark.sql("CREATE DATABASE IF NOT EXISTS default")

**🖥️ Running the ETL Pipeline (Incremental Steps)**

Every time you ingest new data (e.g., weekly), perform these steps clearly:

1. **Ingest new data from API (Raw JSON)**:

python

CopyEdit

ingest\_instance = Ingest(api\_key=dbutils.widgets.get("api\_key"))

ingest\_instance.ingest\_batch("2025-01-01", "2025-01-07")

1. **Run Bronze Streaming (Incremental):**

python

CopyEdit

bronze\_layer = BronzeLayer()

bronze\_query = bronze\_layer.process\_stream()

bronze\_query.awaitTermination()

1. **Run Silver Streaming (Incremental):**

python

CopyEdit

silver\_layer = SilverLayer()

silver\_query = silver\_layer.process\_stream()

silver\_query.awaitTermination()

1. **Run Gold Streaming (Incremental with upsert):**

python

CopyEdit

gold\_layer = GoldLayer()

gold\_query = gold\_layer.process\_stream()

gold\_query.awaitTermination()

**📊 Querying Final Data (Analytics)**

Your final analytical results are stored in the **Gold Delta Table**:

sql

CopyEdit

SELECT \* FROM fixer\_gold ORDER BY date DESC;

The data clearly provides pivoted currency rates (USD, JPY, CNY) by date and event timestamps.

**🧹 Error Handling & Troubleshooting**

* **Schema mismatch errors:** Recreate tables once clearly with overwriteSchema=True.
* **Watermark issues:** Clearly defined watermark intervals (e.g., "7 days").
* **Streaming aggregation errors:** Use foreachBatch clearly to handle complex aggregations with upsert.

**💡 Future Improvement Suggestions**

* Implement automated scheduled workflows with Databricks Jobs.
* Add monitoring and alerting on pipeline failures.
* Integrate Tableau or Power BI dashboards directly on Delta Lake tables for visualization.

**📖 References**

* Fixer API Documentation
* [Azure Databricks Documentation](https://learn.microsoft.com/en-us/azure/databricks/)
* [Delta Lake Documentation](https://delta.io/)