Technical Project Documentation: Batch Processing Data Pipeline

Overview

This project implements a modular batch data processing system that ingests, transforms, and visualizes real-world taxi trip data. It uses a microservices architecture managed with Docker Compose and relies entirely on open-source technologies. The primary goal is to provide a reproducible, scalable, and maintainable pipeline for analytical workloads.

GitHub Repository: https://github.com/ivanmitkov/BatchProcessingDataProject

System Architecture

The project consists of four core microservices:

- Ingestion Service: Downloads and processes the raw dataset from Kaggle.
- Storage Layer: Uses Supabase (PostgreSQL-compatible) as the centralized database.
- Transformation Service: Applies PySpark logic to aggregate and clean the data.
- Visualization Service: A Streamlit dashboard that visualizes aggregated results.

Each component runs in its own container and is orchestrated via Docker Compose. The services communicate indirectly through Supabase, which acts as the single source of truth.

Prerequisites

Before running the project, ensure you have the following installed:

- Docker Desktop
- Docker Compose
- Python 3.10+ (optional, for local testing)
- Supabase account and project with API credentials
- Kaggle account with accepted terms for the dataset

You must also configure a `.env` file and `kaggle.json` credentials.

Environment Variables

Create a `.env` file at the root of the project with the following content:

- SUPABASE_URL=https://your-project.supabase.co
- SUPABASE_KEY=your-supabase-service-role-key

Ensure this file is copied into each service folder or mounted appropriately.

Directory Structure

BatchProcessingDataProject/

├── ingestion_service/

│ ├── ingestion.py

│ ├── Dockerfile

│ ├── requirements.txt

├── transformation_service/

│ ├── transformation.py

│ ├── Dockerfile

│ ├── requirements.txt

├── visualization_service/

│ ├── visualization.py

│ ├── Dockerfile

│ ├── requirements.txt

├── .env

├── docker-compose.yml

└── README.md

Running the Project

To build and launch all services:

```
docker-compose up --build
```

This will:

- Download and process the data from Kaggle (2.3 GB).
- Upload raw data into Supabase (`raw_data` table).
- Wait for data ingestion to complete.
- Transform and aggregate the data with Spark.
- Display results on a Streamlit dashboard at: http://localhost:8501

Service Details

Ingestion Service:

- Downloads dataset using kagglehub
- Extracts SQLite files
- Uploads ~30,000 rows to Supabase

Transformation Service:

- Waits up to 30 minutes for data
- Aggregates daily totals with PySpark
- Writes to `agg_fares_by_day` with uniqueness constraint

Visualization Service:

- Streamlit app at http://localhost:8501
- Displays daily revenue and tips

Data Schema

raw_data Table:

Columns: *vendorid, tpep_pickup_datetime, tpep_dropoff_datetime, passenger_count, trip_distance, ratecodeid, store_and_fwd_flag, pulocationid, dolocationid, payment_type, fare_amount, extra, mta_tax, tip_amount, tolls_amount, improvement_surcharge, total_amount*

agg_fares_by_day Table:

Columns: *pickup_date (date, unique), total_revenue (float), total_tips (float)*

Common Errors and Solutions

1. Issue: Docker error (pipe not found)
   Solution: Ensure Docker Desktop is running
2. Issue: Supabase credentials missing
   Solution: Verify .env file setup
3. Issue: Ingestion not starting
   Solution: Check Kaggle API and network
4. Issue: Transformation retry timeout
   Solution: Increase retries or wait longer

How to Extend

- Replace dataset with another Kaggle SQLite or CSV
- Add more aggregation metrics
- Improve Streamlit interactivity
- Deploy with GitHub Actions or Supabase Edge Functions

Final Notes

This project is ideal for demonstrating batch ETL in a real-world cloud-native setting. It uses production-level tooling in an educational context and offers clear entry points for debugging, experimentation, and automation.

GitHub Repository: https://github.com/ivanmitkov/BatchProcessingDataProject