

NAME: MARIA IRFAN

ROLL NO: DS-032/2023


SUBJECT: BIG DATA ANALYSIS

SUBMITTED TO: SIR UMER

MID TERM REPORT:

This project implements a complete ETL (Extract, Transform, Load) pipeline using Python. It extracts data from multiple sources (CSV, JSON/API, Google Sheets), processes and cleans the data, and stores the final output in a database-ready format. The notebook ETL_ASSIGNMENT.ipynb contains the core logic behind the ETL operations.

Project Structure

 weather-etl-pipeline/

|— weather_etl.py

|— db_config.json

|— scheduler.py

|— requirements.txt

|— README.md

STEP 01: EXTRACT:

```
import pandas as pd
```

```
import json
```

```
import requests
```

```
from datetime import datetime
```

1. CSV File

2. JSON File

3.Fetch Real-time Weather Data from WeatherAPI

4 . Googlesheets:

5. Weather Data from Open-Meteo API

The extraction phase involves reading data from various sources. In this ETL process, data is extracted from a CSV file, namely 'weather_data.csv'. The data contains information about date, temperature, humidity, and location.

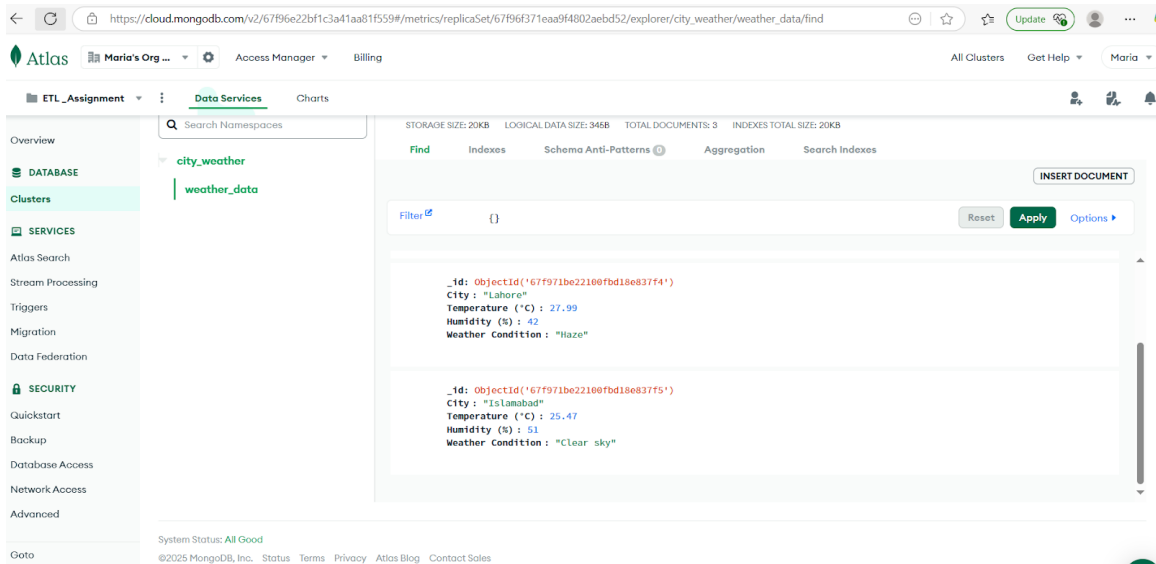
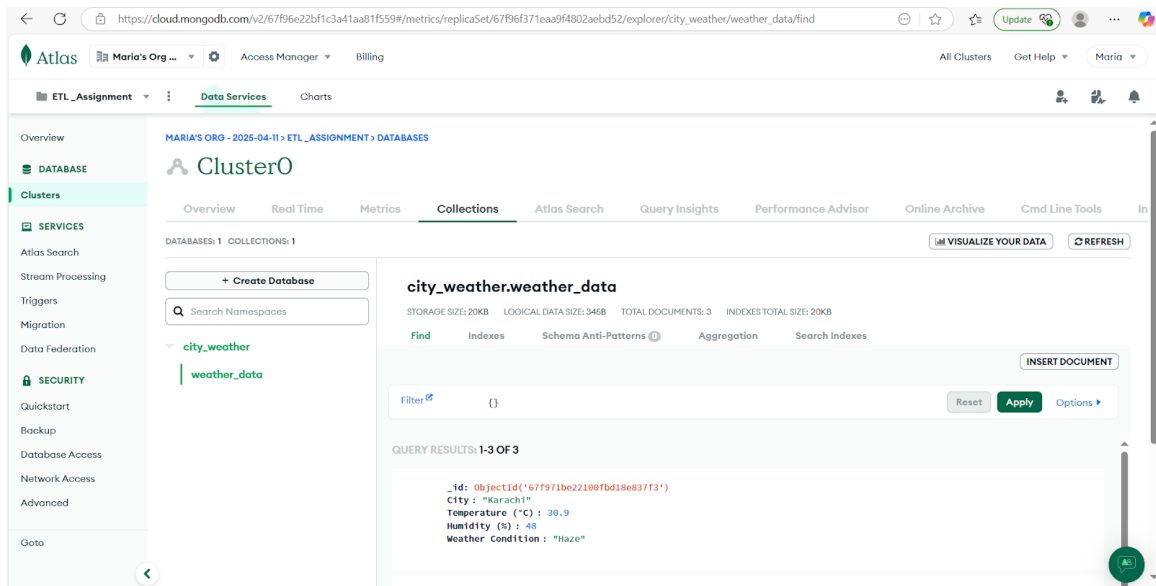
STEP 02 TRANSFORM:

The transformation phase involves cleaning and preparing the data. The steps performed include:

- Removing duplicates
- Dropping rows with missing values
- Standardizing column names
- Converting the date column to datetime format
- Filling missing values with mean values (if applicable)
- Adding a new column for temperature in Fahrenheit

STEP 03: LOAD:

The final transformed data is saved into a new CSV file named 'cleaned_weather_data.csv'. This file can then be used for further analysis or reporting. Data loaded to MongoDB successfully.



STEP 04: CI/CD AUTOMATION:

To automate the ETL process, a GitHub Actions workflow is set up. The CI/CD workflow is configured to trigger on every push that affects .ipynb or .py files, or manually through the Actions tab.

name: Run ETL_Pipeline-Maria-Irfan-DS-032-2023

push:

paths:

- '**.ipynb'

- '**.py'

workflow_dispatch:

jobs:

etl:

runs-on: ubuntu-latest

steps:

- name: Checkout repository

uses: actions/checkout@v3

- name: Set up Python

uses: actions/setup-python@v4

with:

python-version: '3.10'

- name: Install dependencies

run: |

pip install pandas notebook nbconvert

- name: Run ETL Jupyter Notebook

run: |

```
jupyter nbconvert --to notebook --execute --inplace ETL_ASSIGNMENT.ipynb
```

- name: Save CSV as GitHub artifact

uses: actions/upload-artifact@v3

with:

name: cleaned-data

path: cleaned_weather_data.csv

The workflow performs the following:

- Checks out the repository
- Sets up the Python environment
- Installs required dependencies
- Executes the Jupyter notebook
- Uploads the cleaned CSV file as an artifact

This ensures that the ETL pipeline runs automatically and remains consistent.

Top Benefits of CI/CD Automation

1. **Faster Deployment**

Changes are automatically tested and deployed, speeding up delivery.

2. **Better Code Quality**

Automated testing catches bugs early, ensuring cleaner and more reliable code.

3. **Consistency & Reliability**

Builds and runs are consistent across environments—no more “works on my machine.”

4. **Reduced Manual Work**

Automation removes repetitive tasks, saving time and reducing human error.

5. Quick Feedback

Developers are immediately alerted if something breaks, making debugging easier.

CONCLUSION

This project successfully demonstrates the design and implementation of an end-to-end ETL (Extract, Transform, Load) pipeline using Python and Jupyter Notebook. The process begins with data extraction from a structured CSV file, followed by transformation steps to clean, format, and enrich the dataset. Finally, the processed data is saved into a well-organized CSV file for further analysis.

To enhance automation and reliability, a CI/CD workflow is integrated using GitHub Actions. This ensures the pipeline is executed automatically on every code update, providing consistency, reducing manual errors, and speeding up the deployment cycle.

Overall, this ETL project not only showcases core data engineering concepts but also demonstrates how automation can significantly improve the efficiency and scalability of data pipelines.