Data Pipelines and Workflow Orchestration

Running a Data Pipeline with NYC Taxi Data

Objective

By the end of this exercise, you will:

- Load, transform, and analyze NYC Taxi dataset using Python.
- Understand ETL (Extract, Transform, Load) processes in a data pipeline.
- Identify common data pipeline errors and troubleshoot them.

Setup Instructions

1. Create a new notebook on Jupyter

Ensure you have the following Python libraries installed:

Copy

```
pip install pandas pyarrow apache-airflow
```

2. Dataset Download

Download the sample **NYC Taxi Data** CSV file:

Copy

```
!wget "https://data.cityofnewyork.us/resource/m6nq-qud6.csv" -O nyc_taxi_data.csv
```

Step 1: Extract Data

Load the dataset and inspect its structure.

Copy

```
import pandas as pd

# Load dataset
file_path = "nyc_taxi_data.csv"
df = pd.read_csv(file_path)

# Display first few rows
df.head()
```

✅ **Check:** Do you see columns like `tpep_pickup_datetime`, `tpep_dropoff_datetime`, `trip_distance`, `fare_amount`?

Step 2: Transform Data

Perform cleaning and transformation tasks:

Copy

```
# Convert timestamps to datetime format
df['tpep_pickup_datetime'] = pd.to_datetime(df['tpep_pickup_datetime'])
```

```
df['tpep_dropoff_datetime'] = pd.to_datetime(df['tpep_dropoff_datetime'])

# Filter out trips with zero or negative fares
df = df[df['fare_amount'] > 0]

# Calculate trip duration
df['trip_duration'] = (df['tpep_dropoff_datetime'] - df['tpep_pickup_datetime'])
```

✅ **Check:** Run `df.info()` to ensure data types are correct.

Step 3: Load Data

Store the cleaned dataset in a new file:

Copy

```
# Save cleaned dataset
df.to_csv("nyc_taxi_cleaned.csv", index=False)
```

✅ **Check:** Confirm the file `nyc_taxi_cleaned.csv` is generated and contains cleaned data.

Step 4: Pipeline Debugging & Troubleshooting

Common Issues & Fixes

Issue	Cause	Solution
`ParserError` when loading CSV	Incorrect file path or format	Verify file name & use `pd.read_csv('file.csv', error_bad_lines=False)`
`NaT` values in datetime columns	Invalid data format	Use `pd.to_datetime(df['column'], errors='coerce')`
Negative trip durations	Incorrect data entries	Filter out invalid durations with `df[df['trip_duration'] > 0]`

Step 5: Automate with Apache Airflow

Apache Airflow helps automate ETL workflows. It's a platform that programmatically authors, schedules, and monitors data pipelines, making them more maintainable, reliable, and scalable.

We are representing the workflows with DAG (Directed Acyclic Graph):

- Directed: Tasks flow in one direction from upstream to downstream
- Acyclic: No cycles allowed - tasks cannot create circular dependencies
- Graph: A collection of nodes (tasks) connected by edges (dependencies)

Instead of running Python scripts manually or using basic schedulers like cron jobs, Airflow provides:

- Dependency Management
- Robust Scheduling
- Error Handling
- Monitoring
- Scalability
- History Tracking

Here's a basic DAG for orchestrating the NYC Taxi data pipeline:

Copy

```
from airflow import DAG
from airflow.operators.python import PythonOperator
from datetime import datetime
import pandas as pd

def extract():
    df = pd.read_csv("nyc_taxi_data.csv")
    df.to_csv("extracted.csv", index=False)

def transform():
    df = pd.read_csv("extracted.csv")
    df['tpep_pickup_datetime'] = pd.to_datetime(df['tpep_pickup_datetime'])
    df = df[df['fare_amount'] > 0]
    df.to_csv("transformed.csv", index=False)

def load():
    df = pd.read_csv("transformed.csv")
    df.to_csv("nyc_taxi_final.csv", index=False)
```

```
define_dag = DAG(
    'nyc_taxi_pipeline',
    schedule_interval='@daily',
    start_date=datetime(2024, 3, 1),
    catchup=False
)

extract_task = PythonOperator(task_id='extract', python_callable=extract, dag=def
transform_task = PythonOperator(task_id='transform', python_callable=transform, d
load_task = PythonOperator(task_id='load', python_callable=load, dag=define_dag)

extract_task >> transform_task >> load_task
```

✅ **Check:** This creates an Airflow DAG that automates the ETL process.

Wrap-Up

You have successfully:

- Extracted data from a real-world dataset.
- Cleaned and transformed data using Pandas.
- Stored and managed the dataset.
- Automated the process using Apache Airflow.

< Previous

Next >