Data Pipelines and Workflow Orchestration

Recap & Preparing for Apache Airflow

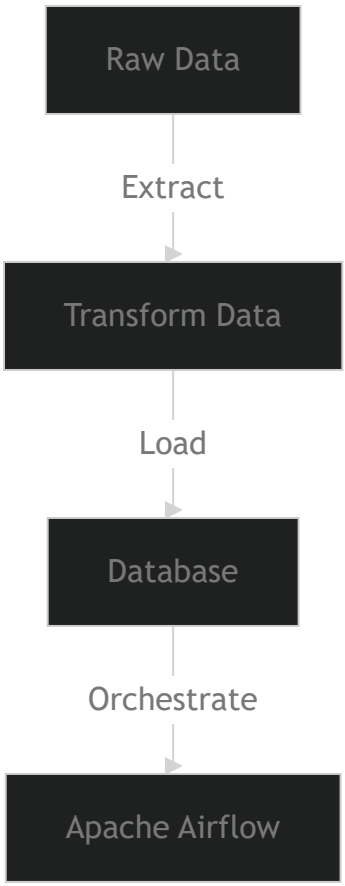
Lesson Recap

Over the past lessons, you have learned how to:

- **Design and implement** a data pipeline using real-world data.
- **Optimize and troubleshoot** pipeline performance and failures.
- **Automate ETL workflows** with best practices for data orchestration.

Key Takeaways

- ✅ **Data Pipelines** ensure efficient data movement and transformation.
- ✅ **Optimization Techniques** improve performance and scalability.
- ✅ **Troubleshooting Skills** help identify and fix failures effectively.
- ✅ **Workflow Orchestration** streamlines execution and scheduling.



Engaging Introduction to Apache Airflow

Before jumping into the lab, let's get hands-on with **some examples** that introduce core Apache Airflow concepts.

Instructor-Guided Mini-Challenges

1. DAGs & Task Dependencies

❓ ***Challenge***: *"How would you define a workflow where Step A must finish before Step B starts?"*

Copy

```
from airflow import DAG
from airflow.operators.dummy_operator import DummyOperator
from datetime import datetime

default_args = { 'start_date': datetime(2024, 3, 1) }

with DAG('simple_dag', default_args=default_args, schedule_interval='@daily') as
```

```
    task_a = DummyOperator(task_id='task_A')
    task_b = DummyOperator(task_id='task_B')

    task_a >> task_b  # Defines dependency
```

Try running this DAG and modifying dependencies!

2. Task Scheduling

***Challenge**: "If a data pipeline should run at 3 AM daily, how would you configure it?"*

Example: Set up a DAG to run daily at 3 AM

Copy

```
with DAG('scheduled_dag', default_args=default_args, schedule_interval='0 3 * * *
    start = DummyOperator(task_id='start')
```

3. Task Failures & Retries

***Challenge**: "What happens if a task fails? How can you retry it automatically?"*

Example: Configuring retries

Copy

```
from airflow.operators.python_operator import PythonOperator
import random

def unstable_task():
    if random.random() < 0.7:
        raise Exception("Random failure!")
    print("Success!")

with DAG('retry_dag', default_args={'retries': 3}, schedule_interval='@daily') as
    task = PythonOperator(task_id='unstable_task', python_callable=unstable_task)
```

Try modifying retry settings to observe behavior!

Preparing for the Apache Airflow Lab

In the next session, you will **build your own data pipeline from scratch** using Apache Airflow.

Next Steps

Review Airflow Documentation: Apache Airflow Docs

< Previous