DATA ENGINEERING | INTERMEDIATE | 3 DAYS

Data Pipelines with Apache Airflow

Build a strong foundation in orchestrating production-grade data pipelines using Apache Airflow, from DAG fundamentals to operational best practices. Learn how to schedule reliable workflows, handle retries and backfills, debug failures, implement validations, and run pipelines with monitoring and alerting patterns.

Why Apache Airflow?

Reliable orchestration

Schedule and manage complex pipelines with clear dependencies.

Scalable automation

Orchestrate batch workflows across warehouses, lakes, and multi-cloud systems.

Improved visibility

Track pipeline status, failures, retries, and SLA performance in one place.

Better governance

Standardize workflow execution with versioned DAGs and audit-friendly operations.

Faster recovery

Built-in retries and alerting reduce downtime when jobs fail.

What you'll be able to do

- Understand why orchestration is essential in modern Data Engineering.
- Build, schedule, and monitor data pipelines using Apache Airflow.
- Understand Airflow concepts including DAGs, tasks, operators, scheduling, retries, and SLAs.
- Implement production-ready workflow best practices such as idempotency, retries, backfill, task dependencies, and timeouts.
- Integrate Airflow with Python ETL scripts, SQL transformations, and dbt workflows (starter).
- Manage Airflow connections, variables, and secrets safely.
- Implement data quality checks and failure alerting strategies.
- Handle operational workflows including reprocessing, partial reruns, and failure recovery.
- Build an end-to-end orchestrated data pipeline as a capstone project.

Who this is for

- Data Engineers
- Analytics Engineers
- Data Platform Engineers
- Data Ops teams
- Backend engineers working with data workflows
- BI engineers supporting scheduled reporting pipelines

Curriculum

01 Why Orchestration Matters in Modern Data Engineering

- What orchestration solves (reliability, sequencing, automation)
- Pipelines vs workflows (tasks, dependencies, retry behavior)
- Batch scheduling challenges (late data, failures, reprocessing)
- Airflow's role in modern data stacks (ETL/ELT + observability)
- **Hands-on:** Activity: Break down a real data pipeline into tasks and dependencies

02 Apache Airflow Fundamentals (Core Concepts)

- Airflow architecture overview (scheduler, webserver, workers, metadata DB)
- Core concepts (DAGs, tasks, operators, task instances)
- Scheduling basics (start_date, schedule_interval, catchup)
- Retries, timeouts, SLAs, and failure behavior
- **Hands-on:** Lab: Create your first DAG and validate it runs successfully

03 Building Pipelines with Operators and Dependencies

- Operator types (PythonOperator, BashOperator, SQL operators overview)
- Task dependencies (linear, fan-out/fan-in, branching concepts)
- Trigger rules and common patterns (all_success vs all_done)
- Task grouping basics (TaskGroup intro)
- **Hands-on:** Lab: Build a DAG with multiple tasks, dependencies, and parallel branches

04 Scheduling, Monitoring, and Operational Visibility

- DAG scheduling patterns (hourly/daily/cron)
- Monitoring DAG runs and task runs in the UI
- Logs, retries, and debugging failures
- SLA monitoring and detecting pipeline freshness issues
- **Hands-on:** Lab: Schedule a DAG, simulate failure, and validate retries + SLA behavior

05 Production Workflow Best Practices (Idempotency, Backfill, Timeouts)

- Idempotency patterns (safe re-runs, overwrite vs append decisions)
- Backfill strategy and catchup behavior
- Timeouts, retries, and exponential backoff concepts
- Designing dependency chains that avoid cascading failures
- **Hands-on:** Lab: Implement an idempotent DAG with a backfill scenario and validate reprocessing safety

06 Integrating Airflow with Python ETL, SQL, and dbt (Starter)

- Calling Python ETL scripts from Airflow
- Running SQL transformations and incremental patterns (starter)
- dbt integration concepts (dbt run, dbt test via operators)
- Passing parameters (run_date, env, paths) into tasks
- **Hands-on:** Lab: Build a DAG that runs Python ETL → SQL transforms → dbt models + tests

07 Connections, Variables, and Secrets (Safe Operations)

- Airflow Connections (databases, APIs, cloud services)
- Variables for configuration and dynamic pipelines
- Secrets handling best practices (no hardcoding, secret backend concepts)
- Environment separation patterns (dev/stage/prod)
- **Hands-on:** Lab: Configure a DB connection + variables and run a DAG using secure values

08 Data Quality Checks and Failure Alerting

- Data quality check patterns (null checks, duplicates, row counts, ranges)
- Fail-fast vs warn-only approach
- Alerting concepts (email/Slack/webhooks overview)
- Adding runbook-ready context to failure alerts
- **Hands-on:** Lab: Add data quality tasks and trigger alerts on validation failures

09 Operational Workflows (Reprocessing, Partial Reruns, Recovery)

- Clearing tasks safely and rerunning only failed steps
- Reprocessing patterns (rebuild specific partition/date)
- Handling partial failures and dependencies
- Failure recovery checklist for production pipelines
- **Hands-on:** Lab: Perform partial rerun recovery and validate pipeline correctness after fix

10 Capstone Project (End-to-End Orchestrated Data Pipeline)

- Capstone goal: Build a production-style orchestrated pipeline
- Ingest data (Python ETL) with parameters (run_date)
- Transform using SQL and dbt models
- Add data quality checks and alerts
- Support retries, backfill, and safe reruns
- **Hands-on:** Capstone Lab: Deliver the working Airflow DAG with evidence, logs, and a short runbook