APACHE AIRFLOW

1. What is Apache Airflow?

Apache Airflow is an open-source platform to programmatically author, schedule, and monitor workflows. It allows you to automate complex data pipelines, ensuring tasks run in the correct order with proper dependencies.

- Workflow Definition: Workflows are defined as Directed Acyclic Graphs (DAGs) using Python code.
- Task Management: Each node in the DAG is a task, which can be anything like running a script, executing SQL, or transferring files.
- Scheduler: Airflow has a scheduler that triggers tasks based on time or external events.
- Monitoring: Provides a web UI to monitor, track, and troubleshoot workflows.

2. Why is Airflow needed?

Modern data processing often involves multiple tasks that need to happen in a specific order. Airflow is needed because:

- **Automation**
  Eliminates manual intervention. Example: Loading data from multiple sources into a data warehouse every day automatically.

- **Scheduling**
  Tasks can be scheduled at fixed intervals (daily, hourly, or custom cron schedules).

- **Dependency Management**
  Handles task dependencies. Example: Task B will run only after Task A succeeds.

- **Scalability**
  Can run multiple tasks in parallel across multiple workers

- **Monitoring and Logging**
  Tracks task progress, failures, and logs. Provides a retry mechanism for failed tasks.

- **Extensibility**
  Supports multiple operators like BashOperator, PythonOperator, SqlOperator, and cloud service operators.

3. Where is Airflow used?

Airflow is widely used in data engineering, data analytics, and DevOps. Common use cases:

- **ETL Pipelines**
  Extract, Transform, Load workflows from multiple data sources into a warehouse.

- **Machine Learning Pipelines**
  Automate ML workflows including data preprocessing, model training, evaluation, and deployment.

- **Data Orchestration**
  Coordinate tasks across different systems. Example: Run a Spark job, upload results to S3, send notification email.

- **Reporting and Analytics**
  Automate generation of dashboards or reports. Example: Run SQL queries daily to generate sales reports and email them to stakeholders.

- **Cloud and DevOps Automation**
  Orchestrate cloud operations, backups, data transfers, and batch jobs

4. Advantages of Airflow

- Open-source and free to use.
- Workflow as code: fully programmable using Python.
- Supports complex dependencies with DAGs.
- Highly scalable and can distribute workloads across multiple workers.
- Provides built-in logging, monitoring, and alerting.
- Extensible with custom operators and plugins.
- Flexible scheduling with time-based or event-based triggers.

5. Key Components of Airflow

- **DAGs**
  Directed Acyclic Graphs define the workflow structure, tasks, and dependencies.

- **Tasks**
  The smallest unit of work in Airflow, which can run Python code, bash scripts, SQL queries, or API calls.

- **Operators**
  Predefined templates for different types of tasks, e.g., BashOperator, PythonOperator, EmailOperator.

- **Scheduler**
  Executes tasks according to the DAG schedule and monitors dependencies.

- **Executor**
  Determines how and where tasks run.

- **WebUI**
  Graphical interface for monitoring DAGs, task logs, and performance metrics.

- **Metadata Database**
  Stores DAG definitions, task states, and logs for tracking workflow execution.