AIRFLOW – 2

1. What is the role of DAGs in monitoring and auditing pipelines?

In Apache Airflow, Directed Acyclic Graphs (DAGs) represent workflows as a set of interdependent tasks executed in a defined sequence. DAGs play a critical role in monitoring and auditing data pipelines because they explicitly define the flow of tasks, their dependencies, and execution order. Each task in a DAG can generate detailed logs, track execution time, and record success or failure status.. For instance, in a data audit workflow, DAGs ensure that data pull, validation, logging, and final status tasks run sequentially and that results are stored in a persistent format for review.

A **DAG (Directed Acyclic Graph)** in Airflow represents the workflow structure of a data pipeline. Each node is a task, and edges define the execution order.

Monitoring: DAGs allow you to see which tasks have run successfully, which failed, and their execution time. This is done through the Airflow web UI, logs, and task-level monitoring.

Auditing: Because every task can log its inputs, outputs, and results, DAGs create a traceable history of the workflow. For example, in a data audit DAG, you can track which data was pulled, whether it passed validation rules, and what the final audit status was.

2. How can Airflow be adapted for event-driven workflows (e.g., reacting to external changes)?

While Airflow is primarily a time-scheduled orchestrator, it can be adapted for event-driven workflows that respond to external triggers or changes. Event-driven DAGs can react when new data arrives in a cloud storage bucket, when a database table is updated, or when an external system sends a webhook. Airflow supports this through sensors, which pause a DAG until certain conditions are met, and through the REST API, which allows external applications to trigger DAG runs dynamically. By combining sensors, PythonOperators, and API triggers, workflows can respond to events in near real-time rather than relying solely on fixed schedules. This makes Airflow suitable for applications where data processing must begin immediately after changes occur, enabling reactive auditing, alerting, and automated data management.

3. **Compare Airflow with cron-based scripting, with at least 2 advantages.**

Airflow provides several advantages over traditional cron-based scripting for managing pipelines. First, dependency management ensures that tasks execute in the proper order, and downstream tasks can be paused or retried if an upstream task fails. Cron, in contrast, executes scheduled scripts blindly at predefined times without considering task dependencies. Second, monitoring and logging are more advanced in Airflow: the web UI provides a complete execution history, task-level logs, and visual workflow graphs, whereas cron logs are usually limited to standard output and error files. Additional benefits include automatic retries, alerting, distributed execution, and better scalability. These features make Airflow more suitable for complex or critical workflows, where reliability and observability are essential.

Advantages:

Dependency Management: Airflow ensures correct task order and prevents downstream failures, unlike cron.

Monitoring and Logging: Airflow provides detailed logs, UI, and retry management, whereas cron requires manual monitoring.

4. **How can Airflow be integrated with external logging/alerting systems?**

Airflow can be integrated with external logging and alerting systems to improve observability and operational control. Task logs can be forwarded to platforms such as the ELK Stack or cloud-native systems like CloudWatch, allowing centralized log analysis and long-term storage. For alerting, Airflow supports email notifications (email_on_failure, email_on_retry) and can be extended with Python or Bash operators to send alerts to Slack, Microsoft Teams or other messaging services. This integration ensures that failures or anomalies are immediately detected and communicated, enabling proactive responses and maintaining compliance in production environments. By combining Airflow’s native logging with external systems, enterprises gain a robust, auditable, and fully monitored data pipeline framework.