**Google Cloud Composer**

**Cloud Composer** is a **fully managed workflow orchestration service** built on **Apache Airflow**. It allows you to author, schedule, and monitor data pipelines across hybrid and multi-cloud environments using Python.

**Core Concepts of Cloud Composer**

1. **Apache Airflow & DAGs**
   1. Workflows are defined as **Directed Acyclic Graphs (DAGs)**.
2. **Composer Environment Architecture**
   1. A Composer environment is a self-contained Airflow deployment hosted on **Google Kubernetes Engine (GKE)**. It includes:
      1. **Airflow web server**: UI for managing DAGs and tasks.
      2. **Airflow database**: Stores metadata about DAG runs and task states.
      3. **Cloud Storage bucket**: Stores DAG files, logs, plugins, and data.
      4. **Composer Agent & Monitoring**: Handles health checks, metrics, and logging.
3. **Interfaces for Management**
   1. You can manage Composer environments and DAGs using:
      1. **Google Cloud Console**
      2. **gcloud CLI**
      3. **Composer REST API**
      4. **Terraform**
   2. DAGs can be managed via:
      1. **Airflow UI**
      2. **Google Cloud Console**
      3. **CLI commands**
4. **Security & Access Control**
   1. IAM roles control access to Composer environments and DAGs.
   2. Composer supports **Private IP**, **Shared VPC**, **VPC Service Controls**, and **CMEK encryption**.
   3. Airflow UI also supports role-based access control.
5. **Monitoring & Logging**
   1. Logs are stored in **Cloud Logging** and the Composer bucket.
   2. Metrics are available in **Cloud Monitoring**.
   3. Airflow UI provides task-level logs and DAG run history.
6. **Customization & Extensibility**
   1. Install custom **plugins**, **operators**, **hooks**, and **sensors**.
   2. Add Python dependencies from PyPI or private repositories.
   3. Customize Airflow configurations (some are restricted by Composer).

**Integration with Google Cloud Services**

1. Composer integrates seamlessly with:
   1. **BigQuery, Cloud Storage, Dataflow, Dataproc, Pub/Sub, Vertex AI**
2. What Is a DAG in Apache Airflow?
   1. A **DAG (Directed Acyclic Graph)** is the backbone of Airflow. It’s a Python-based blueprint that defines:
      1. **Tasks**: Units of work (e.g., run a query, move a file).
      2. **Dependencies**: The order in which tasks should run.
      3. **Schedule**: When and how often the workflow should execute.
   2. The term “acyclic” means tasks can’t loop back—no circular dependencies allowed.
   3. Each DAG is a Python script that defines tasks and their dependencies.
   4. Tasks can do anything: ingest data, transform datasets, trigger APIs, or send alerts.
   5. DAGs can be triggered manually, on a schedule, or by events (e.g., file upload to GCS).
   6. Airflow tasks are powered by **operators**, which are wrappers around actions. Examples include:

| **Operator Type** | **What It Does** |
| --- | --- |
| PythonOperator | Runs Python functions |
| BashOperator | Executes shell commands |
| GCSToBigQueryOperator | Loads data from GCS to BigQuery |
| EmailOperator | Sends email notifications |
| BigQueryInsertJobOperator | Runs SQL jobs in BigQuery |

* 1. DAG Triggers: Manual, Scheduled, or Event-Based
     1. **Manual**
        1. Run from Airflow UI or CLI (airflow dags trigger <dagid> ).
        2. Useful for testing or ad hoc runs.
     2. **Scheduled**
        1. Defined using schedule\_interval (e.g., @daily, @hourly, cron).
        2. Airflow’s scheduler automatically runs DAGs at specified intervals.
     3. **Event-Based Trigger**
        1. Triggered by external events like:
           1. File upload to GCS
           2. Pub/Sub message
           3. API call
        2. Requires sensors or external orchestration (e.g., Cloud Functions + Airflow REST API).
  2. Using gcloud CLI, we can pause the DAG.
     1. **gcloud composer environments run ENV\_NAME --location LOCATION dags pause -- DAG\_ID**

1. **Composer Environment Architecture**
   1. Cloud Composer environments are split across **two projects**:

| **Layer** | **Description** |
| --- | --- |
| **Customer Project** | Your GCP project where you create and manage Composer environments. |
| **Tenant Project** | A Google-managed project that hosts sensitive infrastructure like the Airflow database and runtime components. |

* 1. **Key Environment Components**
     1. **Environment’s Bucket (Customer Project)**
        1. A **Cloud Storage bucket** stores:
           1. DAG files (/dags)
           2. Plugins (/plugins)
           3. Data dependencies (/data)
           4. Logs (/logs)
        2. When you upload a DAG to this bucket, Composer **automatically syncs** it to the Airflow runtime
     2. **Airflow Web Server (Tenant Project)**
        1. Hosts the **Airflow UI** for managing DAGs.
        2. Access is controlled via **IAM roles** and **user identity bindings**.
        3. Supports **role-based access control** (RBAC) for granular permissions.
     3. **Airflow Database (Cloud SQL)**
        1. Stores **Airflow metadata**: DAG runs, task states, connections, variables.
        2. Hosted in **Cloud SQL** within the tenant project.
        3. Access is restricted to the **environment’s service account** for security.
     4. **Airflow Schedulers**
        1. Parse DAG files and **schedule DAG runs**.
        2. Queue tasks for execution by workers.
        3. You can configure multiple schedulers for **load balancing** and **resilience**
     5. **Airflow Workers**
        1. Execute tasks scheduled by the scheduler.
        2. Auto-scaled based on the number of queued tasks.
        3. You can set **min/max worker limits** to control resource usage.
     6. **Airflow Triggerers**
        1. Monitor **deferrable tasks** asynchronously.
        2. Required if you use operators like BigQueryInsertJobOperator with deferral.
        3. Enabled by default in Composer 3.
     7. **Airflow DAG Processors**
        1. Parse DAG files into executable DAG objects.
        2. Run as **separate components** in Composer 3 for better performance isolation

1. **Interfaces for Management**
   1. **Google Cloud Console**
      1. The **Google Cloud Console** provides a **graphical interface** for managing Composer environments and DAGs.
      2. **Environment management**
         1. Create, update, and delete Composer environments.
         2. Configure scaling parameters (e.g. number of schedulers, workers).
         3. Set networking options (Private/Public IP, VPC, firewall rules).
         4. Monitor environment health via built-in dashboards.
      3. **DAG Management**
         1. Upload DAGs to the /dags folder in the Composer bucket.
         2. View DAGs, task instances, logs, and run history in the **Airflow UI**.
         3. Pause/unpause DAGs, trigger manual runs, inspect DAG diagrams.
   2. **Gcloud CLI**
      1. Ideal for scripting and terminal based control.
      2. **Environment Command**