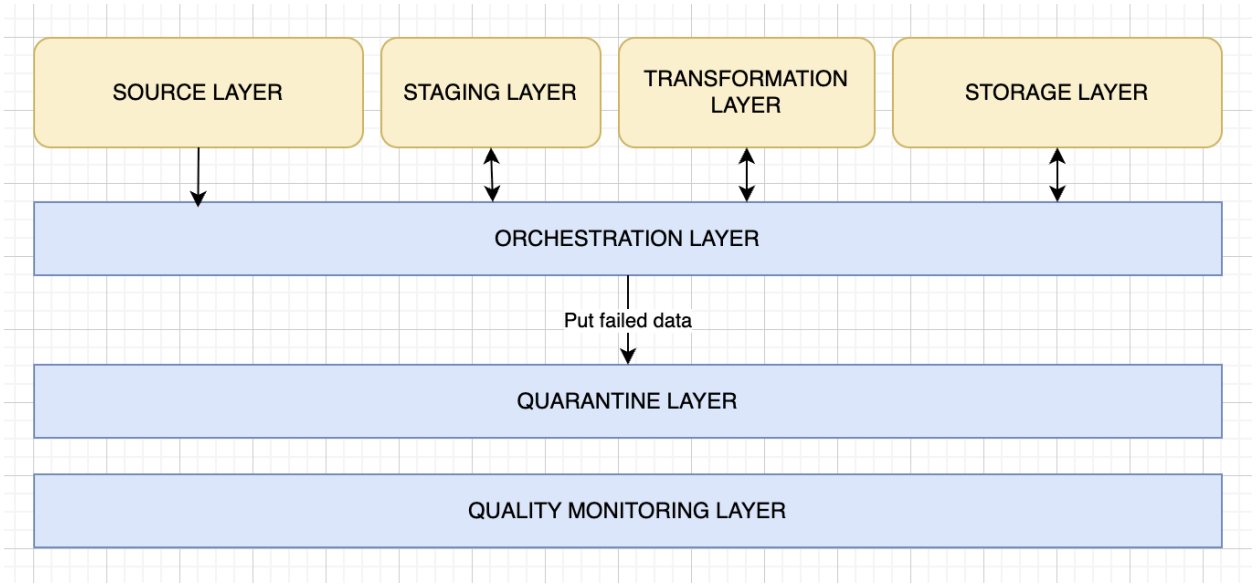
GENERAL ARCHITECTURE



Generally, the data pipeline consists of

1. Source layers: it can be tabular data or unstructured data from a data lake

2. Staging layers: the data source that has already passed the basic validations ends up in this layer. Particularly, the unstructured data, including texts, images, videos, or voices, may require additional steps to transform them into a tabular data style using a machine learning model or rule-based logic.

3. Transformation layer: The data that has passed the next validations, including business rule validations, and also has already undergone data transformations like averaging, outlier or null handling, or aggregation, will end up in this layer.

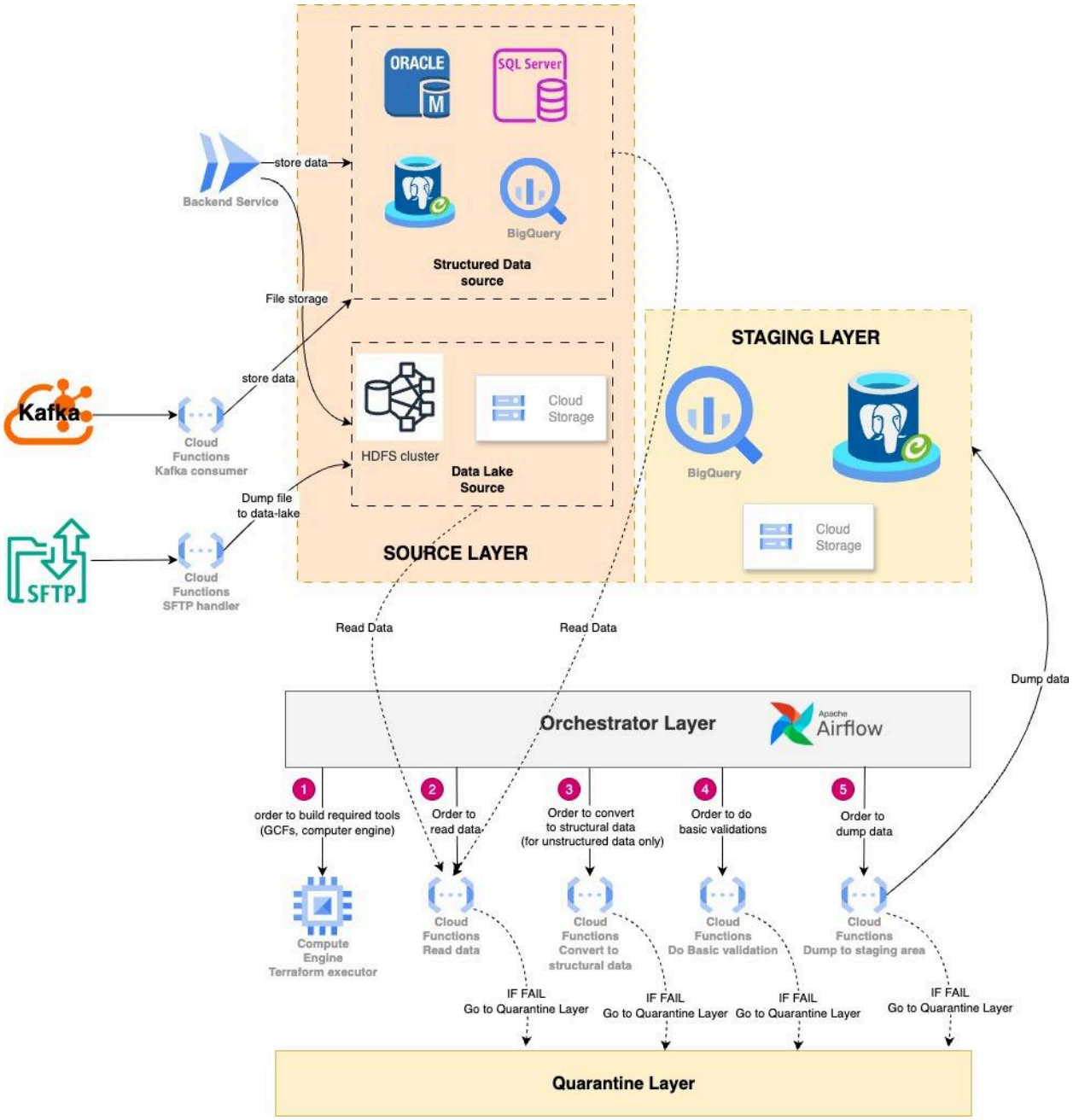
4. Storage layer: Final storage for all transformed data, combined to create a unified view. We can add a data mart as well here.

5. Orchestrator layer: A layer that manages the data movement from the source layer, to the staging layer, and transform layer, then finally resides in the storage layer.

6. Quarantine layer: A layer that contains various services, which are dedicated to handling the failure data points

7. Quality monitoring layer: A layer for monitoring data quality within the data pipeline, including data completeness, uniqueness, correctness, and consistency.

SOURCE ORCHESTRATION



Data comes from various sources, including backend data, messages (through Kafka, Pub/Sub, or AWS SQS), file transfers, etc.

Backend Data

- The data will end up in a structured database like Postgres, Oracle, or BigQuery

- Some unstructured data, like a large text file or image, will end up in a data lake

Messages Data

- The data from messaging services, including Kafka, BigQuery, or AWS SQS, will likely end up as structured data in Postgres or Oracle

SFTP Data

- We need to create a small service in GCP Cloud Function or AWS Lambda, to listen to the SFTP server activities and put the data into Google Cloud Storage

The next step is the data orchestration. We set up Apache Airflow to do data orchestration from the source layer to the staging layer. The orchestration jobs include various steps, which are:

1. Orchestrate to prepare the required infrastructure: Airflow instructs a dedicated GCP Compute engine instance for Terraform, to create the required infrastructures for the job (various GCP Cloud Functions)

2. Once all the required infrastructure was created, Airflow managed to order the GCP Cloud Function to read the data from numerous sources, either from structured sources or from the data lake. The GCF service consists of some business logic to read the data according to its types, formats, and sources.

   The data will end up in a Redis memory store or as Parquet in GCS, depending on the engineering requirements and business SLA.

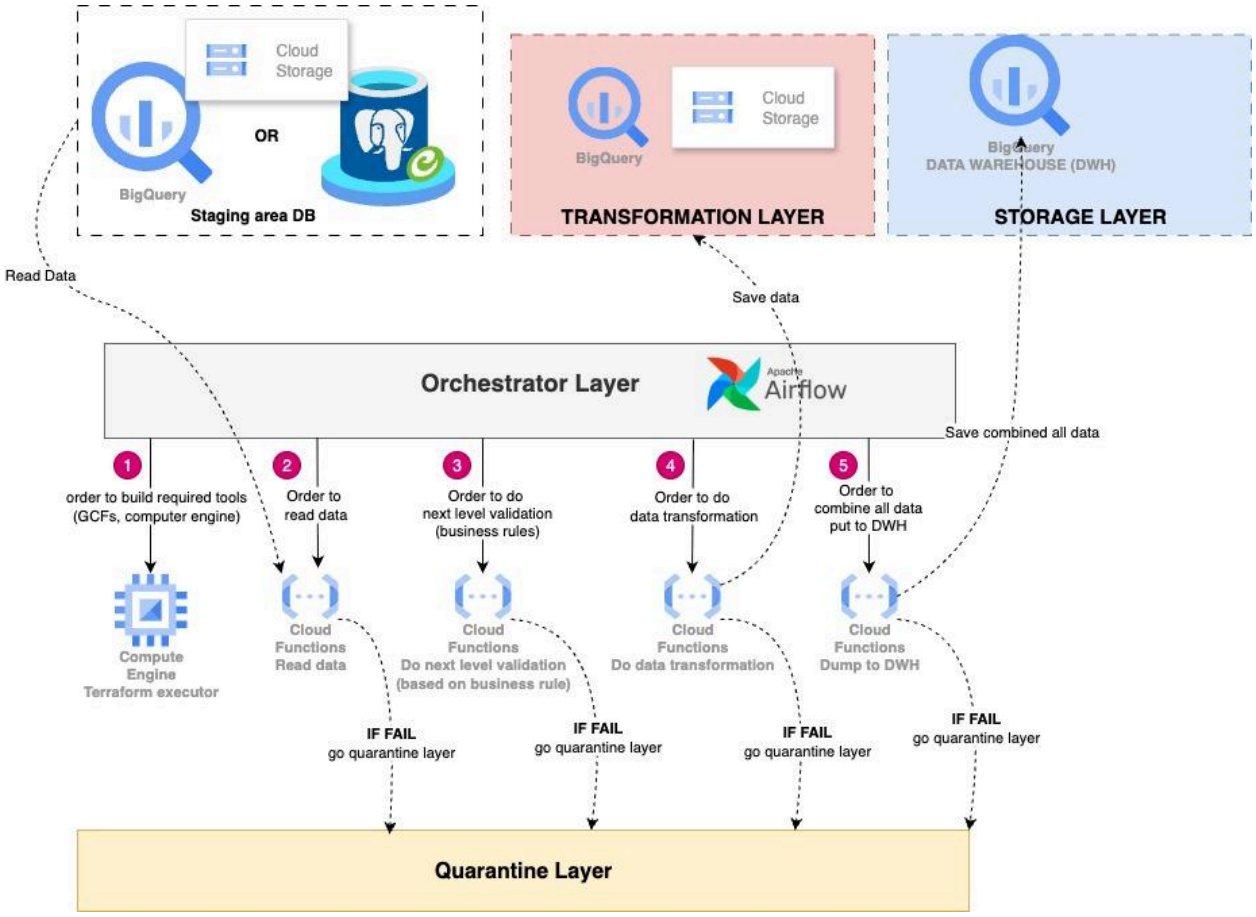
3. Then, Airflow orders the other GCF service to convert the unstructured data to be structured data. At this point, it may require a machine learning model to convert images, videos, texts, or voices into tabular data. But it goes back to the business requirements and budgets.

4. After that, Airflow instructs a GCF service to do some basic validations, including
   - Required columns
   - Column data types
   - Column data ranges
   - Referential integrity

5. Finally, Airflow manages to orchestrate another GCF service to put the data in the staging layer.

Any failed data points for each step will be sent to the Quarantine Layer. At this layer, we manage to do various data failure handlers, including lowering data validation requirements, manual checking, and the last option is to set data failure expiry.

TRANSFORMATION ORCHESTRATION



At this point. We managed to set up Apache Airflow to do data orchestration from the staging layer to the transformation and storage layer. The orchestration jobs include various steps, which are:

1. Orchestrate to prepare the required infrastructure: Airflow instructs a dedicated GCP Compute engine instance for Terraform, to create the required infrastructures for the job (various GCP Cloud Functions)

2. Read data from the staging layer

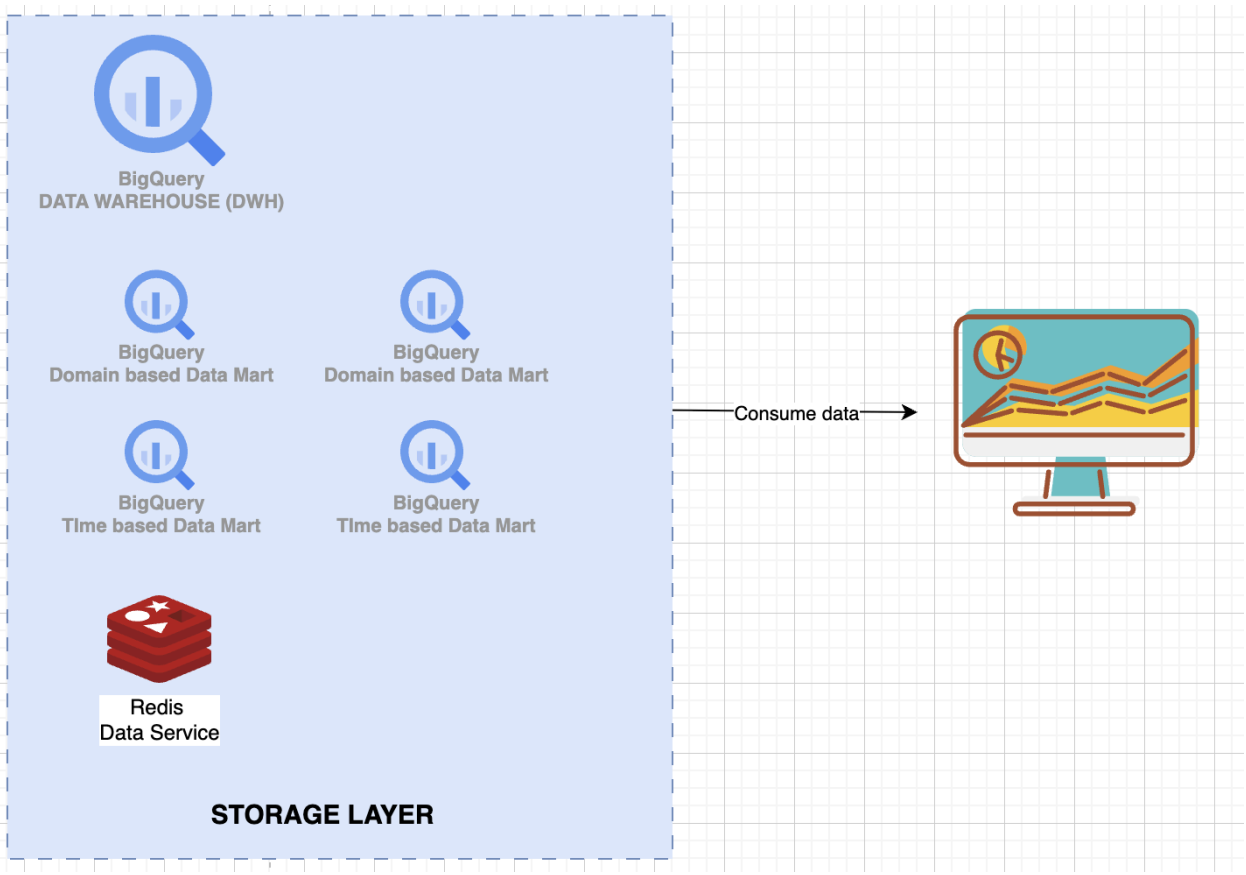
3. Order the GCF service to do the next-level validations, including the business rule validations

4. Instruct another GCF service to do the data transformations, like data averaging and aggregations. The outlier and NaN handling is also performed here.

5. The CGF will put the transformation results into the transformation layer.

6. Lastly, Airflow orchestrates another GCF to combine the transformation results and put them into the storage layer.

STORAGE LAYER and DATA VISUALIZATION



We can create various data marts according to the business requirements.
The data mart can be developed based on business types, like the sales data mart and campaign data mart. It can also be developed based on time domains, like a monthly or yearly data mart.

We can put a dedicated Redis instance as a cache for highly retrieved data, for example: daily sales, daily revenue, monthly revenue, etc.