Part2

How to implement file on-premise RDBMS to cloud with scalability

- The Cloud Service that I Choose is Google Cloud Service (GCS)
- I would like to implement the ETL dataflow using python file as a source data by using Apache Airflow to write data to a Data Lake table that will be set up on Google Bigquery and use schedule queries to take these data to the data warehouse to a partitioned table that will be partitioned by dates. The purpose of partitioning is to reduce the cost when querying data each time interval based on the scheduled time intervals.

To design the database with scalability in order to support big amount of users, the design should follow these guidelines

- Set up load-balance at each tier. load balancing lets distribute traffic among group of resource.It will help to ensure resource don't become overloaded while other idle and choose to split database service instead of expand size of database service because in case expand size when server was down it will not have a backup sever to process data

- Determine scaling profile to decide what minimum acceptable level of performance by autoscaling features for ensure when on unexpected amount of user situation

The solution for monitoring/logging/auto notification

- I suggest using Apacha Airflow tools for monitoring/logging/auto notification on step run ETL python pipeline from on-premise database to Bigquery. When transfering from datalake to datawarehouse I will use scheduled query functions on BigQuery service which will automatically save logs and notify through emails directly when issues occur.

Explain Tools

- Apache Airflow : I chose this tool because it is an open-source tool that can show the full workflow in detailed and support Centralized Management to set up jobs called "DAGs". It has WebUI that is easy to understand for configuration, so in case a job fails to run it can be monitored via WebUI. Airflow has a function called retry mechanisms that can retry tasks both automatically or manually when there is a problem. Finally, to address issues concerning scalability, Airflow has a separate architecture such as sequential mode and parallel mode that can be applied to any workflow.
  For more detail about Apache Airflow : apache/airflow: Apache Airflow - A platform to programmatically author, schedule, and monitor workflows (github.com)