Task 1. ETL pipeline code

Your job in this assignment:

- Task-1:

How would you read or analyse the results of above query? Also share the results produced in readable format(csv)

Bonus: How would you improvise the documentation of the code provided, and document it better for readability?

*** I'm not sure if I understand the question correctly. The read/analyze is to analyze the given data or to analyze it further and another confusion is I must read/analyze it further and load the results in CSV format or convert Parquet to CSV and write an explanation about the read/analyze. So, I decided to do both. Since I work on the weekend, HR didn't reply to my email. That's not their fault; I just explained the situation. ***

Answer:

def transform()

The query in the transformation functions that transforms the extracted data to find total transactions and total customers and transforms the extracted data to find customers' favorite

From the df_favorites query is to finds each customer's favorite products by looking at all the products they have purchased and counting the number of items the customer has purchased (SUM(vt.units Sold)). Then, it ranks the products the customer has purchased the most by (RANK() OVER), and finally selects RANK 1, which is the number 1 product purchased by each person. Then, it counts the number of customers who have the same favorite product (COUNT(custId)), and orders the product with the most favorites on top (ORDER BY cnt DESC).

Let's say we are store managers from this data we can find the trends of best-selling products in each customer and use data to adjust the stock item more on products that have the highest sold.

def analyze()

The function in the analyze function is a function that I added because I was not sure if I understood the problem correctly. I took the data from the customer's favorite products and found out which products were liked the most by the customer and how many products the customer liked the most from the data in the transform function and load to csv format in both query.

Ranking_product_sold: Find the most product that the customer buy the most from the customer table using Count and GROUP BY to sort by product group sold (productSold), then descending by sales quantity (totalproductsale).

Ranking_transaction_customer: Find the number of customers' favorite products also use count and GROUP BY by customer ID (custId), then descending by quantity (total transaction).

- Task-2:

Make changes to save the results additionally in postgres in a database called "warehouse" and a table called "customers"

The following command should be executable:

docker-compose run etl python main.py --source /opt/data/transaction.csv --database warehouse --table customers

Hint:

- Make the required changes in:

- etl_jobs/EtlJobForSertis.py (You may use the 'pass' Python keyword left in the code as a helping marker)

- docker-compose (to add postgres service)

- Dockerfile (optional to add missing dependencies)

- You may rename .env.sample to .env and use the env variables in it

```
PS D:\download\take-home-test-take-home-test-for-interns\take-home-test-take-home-test-for-interns> docker-compose exec postgres psql -U myuser -d warehouse
time="2025-03-16T22:03:17+07:00" level=warning msg="D:\\download\\take-home-test-take-home-test-for-interns\\take-home-test-take-home-test-for-interns\\docker-compose.yml: `version` is obsolete"
service "postgres" is not running
PS D:\download\take-home-test-take-home-test-for-interns\take-home-test-take-home-test-for-interns> docker-compose exec postgres psql -U myuser -d warehouse
time="2025-03-16T22:03:27+07:00" level=warning msg="D:\\download\\take-home-test-take-home-test-for-interns\\take-home-test-take-home-test-for-interns\\docker-compose.yml: `version` is obsolete"
psql (13.20 (Debian 13.20-1.pgdg120+1))
Type "help" for help.

warehouse=# select * from customers limit 10;
 custId  | productSold | cnt
---------+-------------+-----
 0023262 | DETA800     |   1
 0023263 | PURA100     |   1
 0023264 | PURA500     |   1
 0023266 | SUPA104     |   1
 0023267 | PURA100     |   1
 0023268 | PURA250     |   1
 0023269 | SUPA101     |   1
 0023270 | PURA500     |   1
 0023271 | PURA250     |   1
 0023273 | SUPA101     |   1
(10 rows)

warehouse=#
```

Task 2. System Architecture

Preparation

This task does not have to be implemented. Imagine you write a proposal to us for data warehouse solution on Cloud.

Choose one of the major cloud platforms, **Amazon Web Services**, **Google Cloud Platform** or **Microsoft Azure** as a basis of your solution and use services provided by the chosen platform.

Requirements

Describe what technolgoies would you choose and why, in order to build your proposed architecture

Bonus:

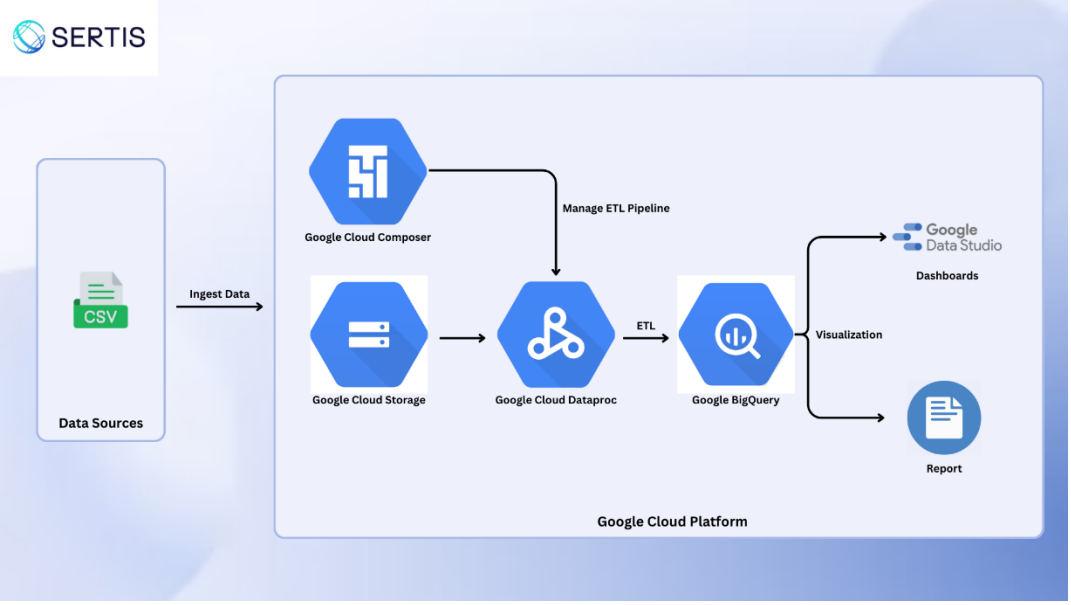
* Propose solutions on an additional cloud platform, covering the same requirements

Submission

Explanation of the proposed solution, chosen tools/frameworks, justification on why you chose them, and any other supporting documentation, in PDF or markdown format.

Google Cloud Platform Data Warehouse Solution Proposal for Sertis

Proposed Solution: Data Warehouse Architecture Overview



Data warehouse solutions are structured in different layers, not just a single data warehouse solution. I do this to give a more overview. The solution uses Google Cloud Platform technologies to meet the scalability, performance, and usability requirements and is based on the work in Sertis. The architecture consists of:

Google Cloud Storage: Google Cloud Storage is a highly scalable, durable object storage service designed to store structured and unstructured data. Google Cloud Storage is for all raw data, whether batch or streaming for real-time scenarios.

Google Cloud Dataproc: Google Cloud Dataproc is a fully managed cloud service that simplifies the process of running Apache Spark and Hadoop. It is particularly well-suited for organizations like Sertis, which specializes in big data solutions. With Dataproc, we can leverage a highly scalable to execute large-scale batch and stream processing workflows. The service supports frameworks such as PySpark and Hive, which are essential for processing and analyzing large datasets.In this architecture Dataproc processes raw data from GCS by transformations using PySpark, and prepares it for storage in BigQuery.

Google Cloud Composer: Google Cloud Composer is a managed service that helps automate and manage workflows, based on Apache Airflow. It's perfect for orchestrating data pipelines, such as ETL (Extract, Transform, Load) processes in this architecture

Google BigQuery: Google BigQuery is a fully managed, AI-ready data analytics platform that helps you unlock the full potential of your data. It supports SQL-based queries and is optimized for large-scale analytics, delivering high performance for real-time data analysis. With Google BigQuery, teams can analyze massive datasets in seconds, enabling faster insights and decision-making. Its powerful capabilities make it an ideal tool for Sertis the organizations that looking to get the most value out of their data quickly and efficiently for cutting-edge AI and Big Data solutions.

Google Data Studio: Data Studio is a free, user-friendly tool for creating interactive dashboards and connects to Google BigQuery directly allowing to visualize.

Benefits of the Proposed Solution

Scalability: BigQuery and Dataproc automatically scale to handle growing data volumes, ensuring performance as your needs expand.

Cost Efficiency: Pay-per-use pricing (e.g., BigQuery's query-based pricing, Dataproc's on-demand clusters) ensures you only pay for what you use.

Fully Managed: GCP services like Dataproc, BigQuery, and Cloud Composer are fully managed, reducing operational overhead for your team.

Real-Time Insights: BigQuery's real-time querying and Data Studio's live dashboards enable immediate access to insights.

Security & Compliance: IAM, KMS, and Monitoring/Logging ensure data security and compliance with industry standards.

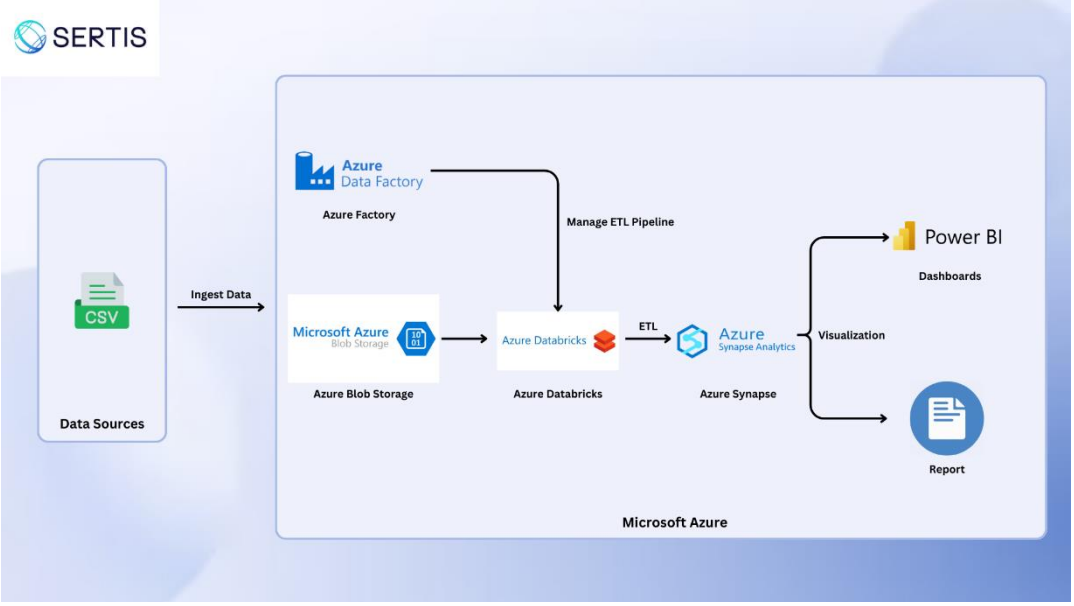
Ease of Use: Tools like Data Studio and BigQuery's SQL interface make the solution accessible to both technical and non-technical users.

Conclusion

This GCP-based data warehouse solution is good for Sertis needs for a modern, cloud-native data platform for Unlocking the full potential of customers data with cutting-edge AI and Big Data solutions. It combines scalability, cost-efficiency, and ease of use with robust security and advanced analytics capabilities. By leveraging fully managed services, your team can focus on deriving value from data rather than managing infrastructure. Whether you need real-time dashboards, predictive models, or secure data sharing, this solution provides flexibility and performance to meet Sertis goals.

Microsoft Azure Data Warehouse Solution Proposal for Sertis

Proposed Solution: Data Warehouse Architecture Overview



I designed a data warehouse solution on Microsoft Azure that mirrors the structure covering the same requirements as GCP proposal, ensuring scalability, security, and ease of use for Sertis data needs. This solution uses Azure's managed services to handle data ingestion, processing, storage, and analytics, with tools for visualization.

Azure Blob Storage: Azure Blob Storage is scalable object storage, perfect for storing large datasets like CSV or JSON files and integrates well with other Azure services Azure Blob Storage is chosen for its ability to handle larger amount of data with high durability. It also supports batch uploads and real-time streaming ensuring flexibility for both historical and live data ingestion and tiered storage options (Hot, Cool, Archive) optimize costs.

Azure Databricks: Azure Databricks is a managed Spark service that efficiently manages big data processing using Spark for ETL jobs and scales on demand. Azure Databricks, a managed Apache Spark platform, integrates with Blob Storage and Synapse to provide a scalable environment for big data processing, supporting scalability and cost efficiency.

Azure Data Factory: Azure Data Factory providing managed workflow orchestration based on Apache Airflow principles. It automates ETL pipelines, scheduling tasks like running Databricks jobs and loading data into Synapse SQL. It handles dependencies, retry logic, and error handling, ensuring reliability, and scales to manage complex workflows without manual intervention.

Azure Synapse SQL: Azure Synapse SQL is a serverless data integration service designed for seamless SQL querying and analytics. It automatically scales to handle large datasets and concurrent queries, offering high performance with a distributed query processing architecture. With its pay-per-use model, it ensures cost efficiency while delivering powerful analytics capabilities. Integration with other Azure services like Power BI and Data Factory enables the creation of an efficient data pipeline the entire process from data ingestion to visualization that needs unlocking the full potential of customer data with cutting-edge AI and Big Data solutions.

Microsoft Power BI: Microsoft Power BI connects to Synapse SQL to create interactive dashboards and reports, providing real-time insights for stakeholders.

Benefits of the Proposed Solution

Scalability: Services like Synapse SQL and Databricks scale automatically, handling growing data volumes.

Cost Efficiency: Pay-per-use pricing ensures you only pay for resources consumed, with options like Blob Storage tires reducing costs.

Fully Managed: Managed services reduce operational overhead, allowing your team to focus on analytics.

Real-Time Insights: Synapse SQL's querying and Power BI's dashboards provide immediate insights.

Security & Compliance: Azure's security tools ensure data protection and compliance with industry standards.

Ease of Use: Power BI and Synapse SQL's SQL interface are accessible to both technical and non-technical users.

Conclusion

This Azure-based data warehouse solution is an ideal fit for Sertis, providing a modern, cloud-native platform that offers scalability, cost-efficiency, user-friendliness, and strong security, alongside advanced analytics capabilities. By utilizing Azure's fully managed services, your team can focus on generating actionable insights from data, whether through real-time dashboards, predictive models, or secure data sharing. This solution guarantees that Sertis can achieve its data management objectives with the flexibility, performance, and confidence it needs. Unlike GCP, Azure Synapse integrates built-in Spark capabilities, which can streamline processing. Azure Synapse's robust features ensure a unified approach to data management, optimizing analytics while maintaining flexibility for advanced processing requirements.