**Freight Flow Optimizer**

* Shipping Route and Cost Analysis

**CASE STUDY**

Table Of Contents

[1. Data Sources and Descriptions 3](#_Toc199508131)

[2. SQL queries for analysis 4](#_Toc199508132)

[3. Functional,Volumetric Details & (PI) concerns : 17](#_Toc199508133)

[4. Methodology 20](#_Toc199508134)

[5. Query-by-Query Findings & Dataset Mappings 23](#_Toc199508135)

[6. Conclusion 31](#_Toc199508136)

## Data Sources and Descriptions

This dataset consists of **seven interrelated tables**, designed to represent real-world logistics and supply chain optimization problems. The goal is to assign **shipping routes to customer orders** while satisfying multiple **operational constraints** and minimizing **freight and warehousing costs**. The data is modeled around a single day of operations and is deterministic in nature.

1. OrderList

* Purpose: Central table containing historical data about customer orders and their assigned fulfillment routes.
* Key Fields: Order ID, Product ID, Customer ID, Order Quantity, Unit Weight, Warehouse (Plant), Port.

Notes: This is the primary table to be optimized. It includes past routing choices which can be compared against optimized outputs under current constraints.

2. FreightRates

* Purpose: Contains all available shipping options with associated constraints and costs.
* Key Fields: Warehouse, Port, Min Weight, Max Weight, Freight Cost, Transport Day Count (tpt\_day\_cnt).

Notes: Used to determine valid weight-based shipping lanes and cost per shipment.

3. PlantPorts

* Purpose: Lists valid warehouse-to-port connections.
* Key Fields: Warehouse, Port.

Notes: Ensures that routing decisions only use physically possible paths.

4. ProductsPerPlant

* Purpose: Specifies which warehouses are capable of fulfilling specific product types.
* Key Fields: Warehouse, Product ID.

Notes: Prevents routing orders to warehouses that do not support the required product.

5. VmiCustomers

* Purpose: Lists special customers (VMI - Vendor Managed Inventory) who are tied to specific warehouses.
* Key Fields: Customer ID, Warehouse.

Notes: These customers can only be served by listed warehouses. All others can be served by any warehouse not listed here.

6. WhCapacities

* Purpose: Indicates the processing capacity of each warehouse.
* Key Fields: Warehouse, Max Orders.

Notes: Capacity is measured in the number of distinct orders, not units. For example, if a plant has a capacity of 3, it can handle three orders regardless of the number of units in each plant.

7. WhCosts

* Purpose: Represents the storage cost per unit for each warehouse.
* Key Fields: Warehouse, Cost per Unit.

Notes: Used in calculating the warehousing cost as part of the optimization objective.

## 2. SQL queries for analysis

### 1. Total Units and Order Count by Plant.

SELECT

Plant\_Code,

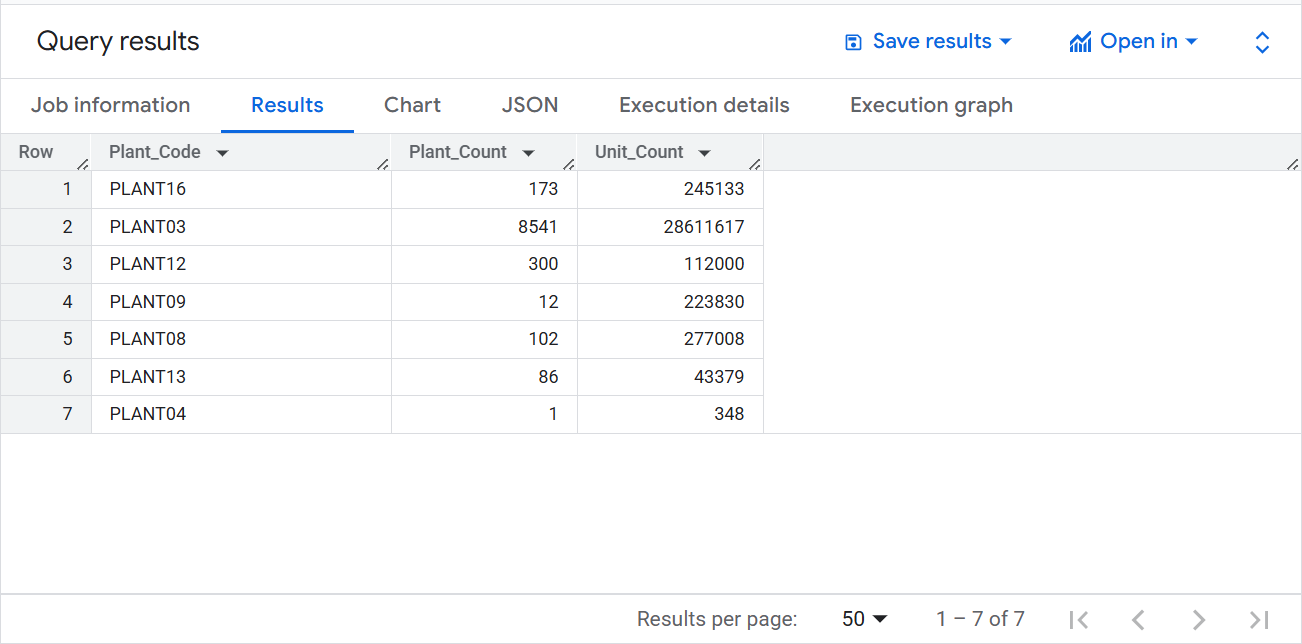
COUNT(Plant\_Code) AS Plant\_Count,

SUM(Unit\_Quantity) AS Unit\_Count

FROM

shipping-logistics-gcp.Shipping\_supply\_chain.OrderList\_table\_new

GROUP BY

Plant\_Code;

#### Explanation of query logic

This query calculates the total number of orders and the total units shipped from each plant in the supply chain. It groups the data by Plant\_Code, then uses COUNT(Plant\_Code) to determine how many orders are associated with each plant and SUM(Unit\_Quantity) to compute the total number of units shipped. The result provides a summary of shipping activity per plant, showing how active each plant is in terms of order frequency and shipment volume.

### 2. Total Weight Shipped by Plant.

SELECT

Plant\_Code,

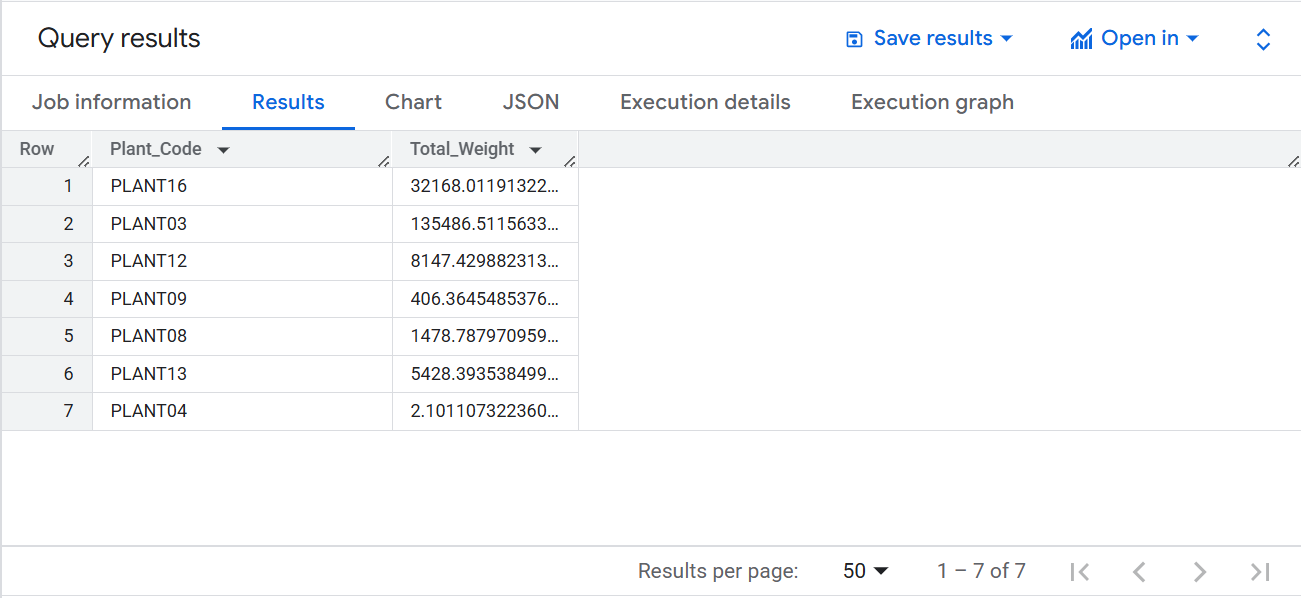
SUM(Weight) as Total\_Weight

FROM

shipping-logistics-gcp.Shipping\_supply\_chain.OrderList\_table\_new

GROUP BY

Plant\_Code;



#### Explanation of query logic

This query calculates the total weight of all shipments made from each plant by grouping the data based on Plant\_Code and summing the Weight column for each group. The result shows the cumulative weight of products shipped from each plant, helping to assess the shipping load and distribution volume handled by different plant locations.

### 3. Product Shipment Volume and Distribution Across Warehouses.

SELECT

Product\_Id,

SUM(Unit\_Quantity)AS Volume,

COUNT(Plant\_Code) AS No\_Of\_Warehouses

FROM

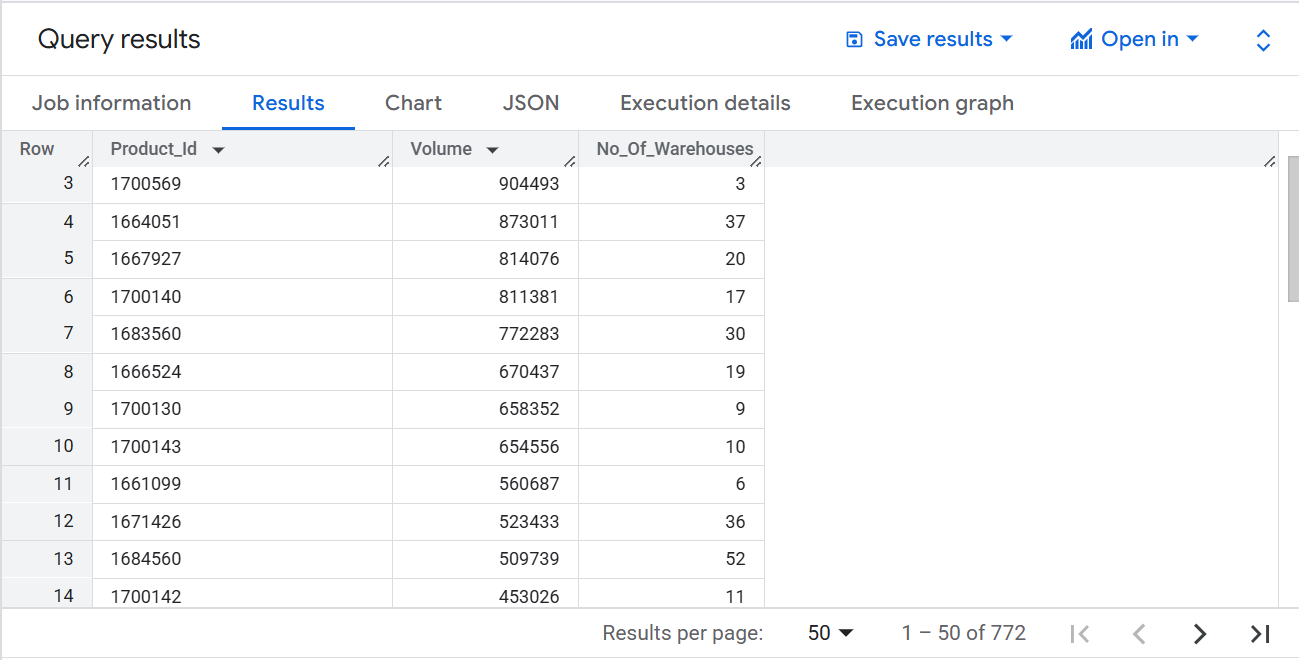
`shipping-logistics-gcp.Shipping\_supply\_chain.OrderList\_table\_new`

GROUP BY

Product\_Id

ORDER BY

Volume DESC;



#### Explanation of query logic

This query analyzes the shipment volume and distribution reach of each product by grouping the data by Product\_Id. It calculates the total number of units shipped using SUM(Unit\_Quantity) as Volume and counts how many different Plant\_Code entries are associated with each product using COUNT(Plant\_Code) as No\_Of\_Warehouses, indicating how widely each product is distributed across plants. The results are ordered in descending order of shipment volume to highlight the most frequently shipped products.

### 4. Average Freight Rate and Total Quantity by Carrier.

SELECT

DISTINCT f.Carrier,

ROUND(AVG(f.Rate), 3) AS RATE,

SUM(o.Unit\_Quantity) AS Total\_Quantity

FROM

shipping-logistics-gcp.Shipping\_supply\_chain.FreighRates\_table f

JOIN

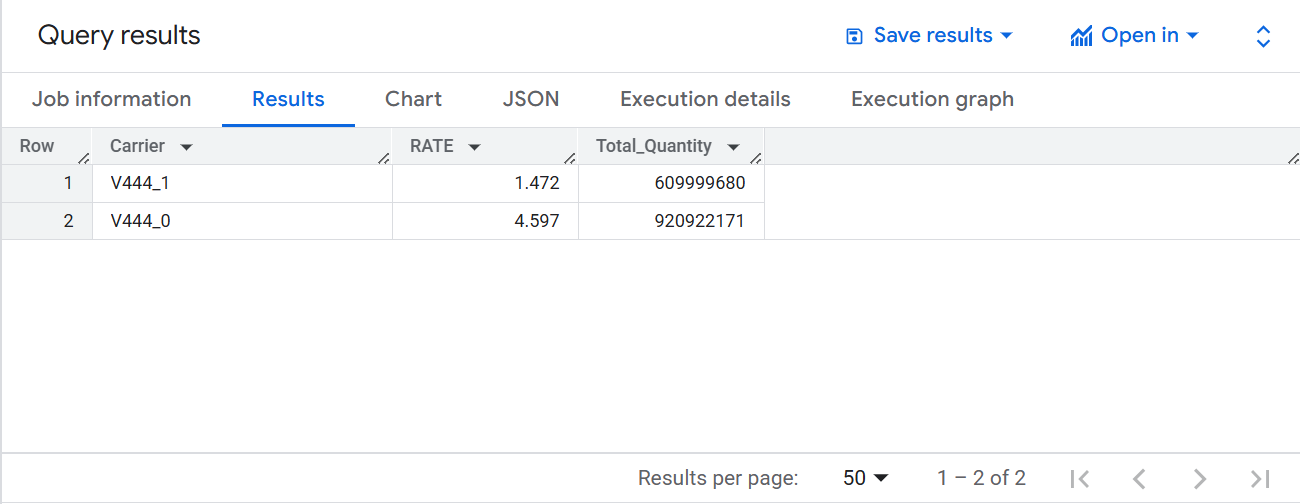
shipping-logistics-gcp.Shipping\_supply\_chain.OrderList\_table\_new o

ON

f.Carrier = o.Carrier

GROUP BY

f.Carrier;



#### Explanation of query logic

This query provides insights into carrier performance by joining the freight rates and order details based on the Carrier field. It selects each unique carrier, calculates the average freight rate using AVG(f.Rate) rounded to three decimal places, and computes the total quantity of units shipped through each carrier using SUM(o.Unit\_Quantity). By grouping the results by f.Carrier, the query summarizes both cost efficiency and shipping volume for each carrier.

### 5. Top 10 Customers by Number of Orders.

SELECT

DISTINCT Customer,

COUNT(Order\_Id) AS No\_of\_Orders

from

shipping-logistics-gcp.Shipping\_supply\_chain.OrderList\_table\_new

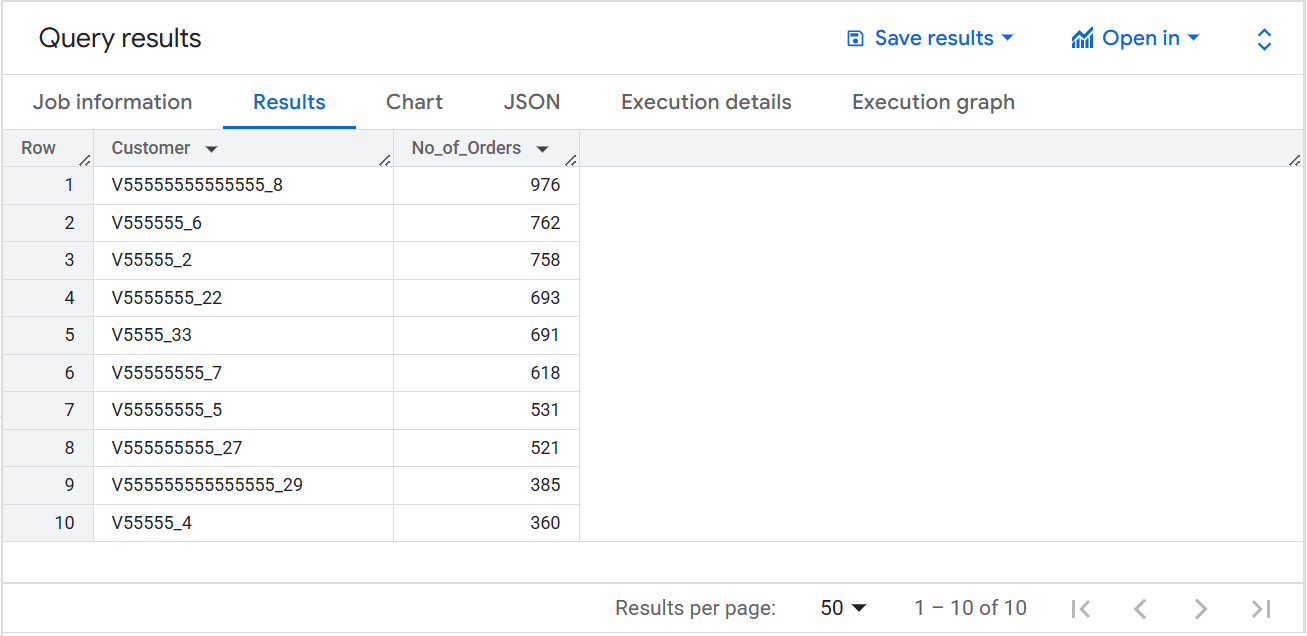
Group by

Customer

ORDER BY

No\_of\_Orders DESC

LIMIT 10;



#### Explanation of query logic

This query identifies the top 10 customers based on the number of orders they have placed. It groups the data by Customer, counts the total number of orders for each using COUNT(Order\_Id) as No\_of\_Orders, and orders the results in descending order to highlight the most frequent customers. The LIMIT 10 clause ensures only the top 10 customers are displayed.

### 6. Average Freight Weight Range by Carrier.

select

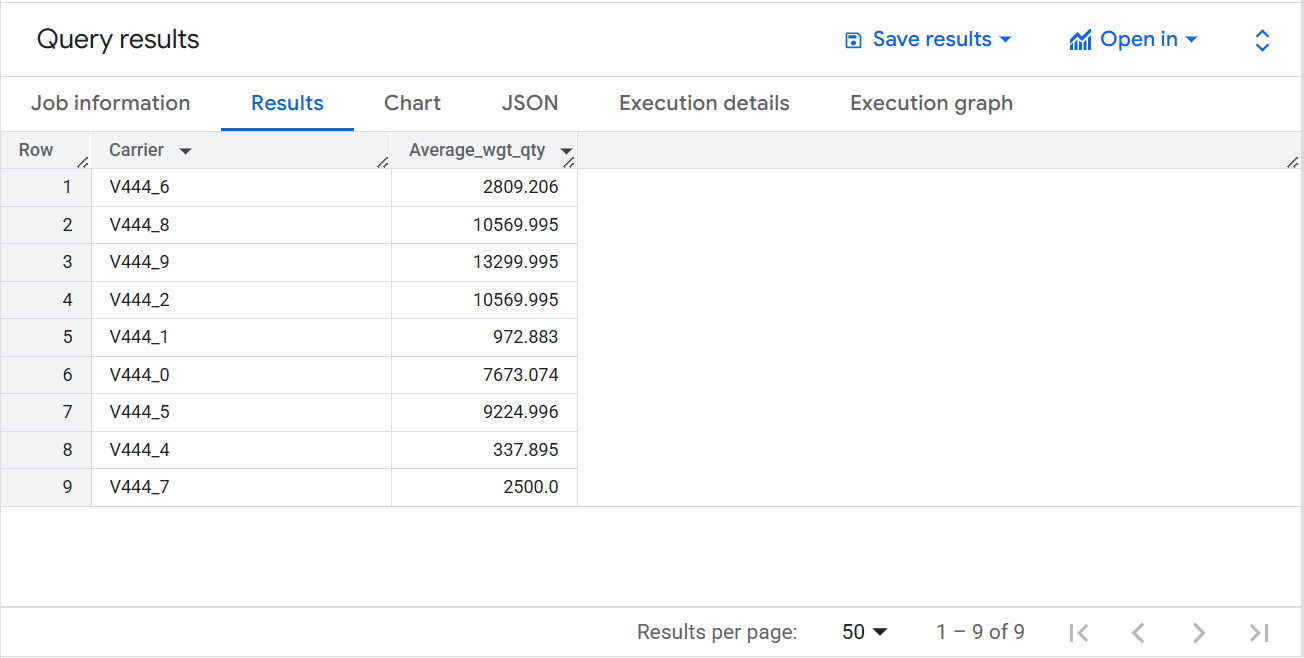
Carrier,

ROUND(AVG((Minm\_wgh\_qty + Max\_wgh\_qty)/2), 3) as Average\_wgt\_qty

from

shipping-logistics-gcp.Shipping\_supply\_chain.FreighRates\_table

group by Carrier;



#### Explanation of query logic

This query calculates the average freight weight range for each carrier by taking the mean of the Minm\_wgh\_qty and Max\_wgh\_qty fields for each record, then averaging that result across all records per Carrier. The result, rounded to three decimal places, provides an estimate of the typical shipment weight each carrier handles, offering insights into carrier capacity and load tendencies.

### 7. Average Freight Weight Range by Plant.

SELECT

o.Plant\_Code,

ROUND(AVG((f.Minm\_wgh\_qty + f.Max\_wgh\_qty)/2), 3) as Average\_wgt\_qty,

FROM

shipping-logistics-gcp.Shipping\_supply\_chain.FreighRates\_table f

JOIN

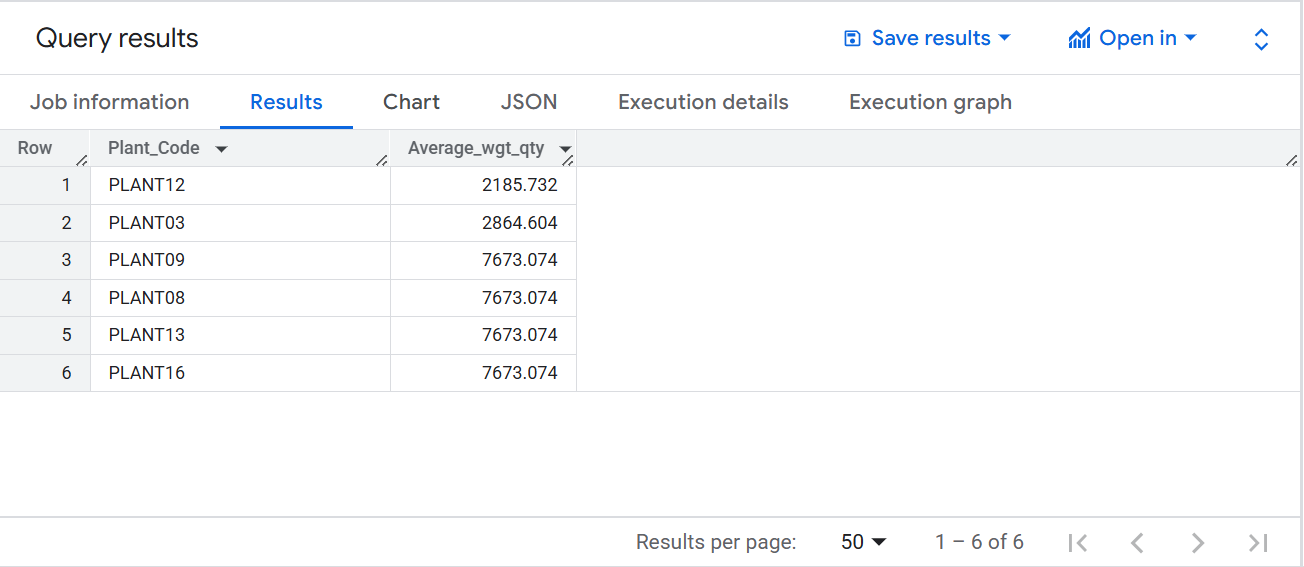
shipping-logistics-gcp.Shipping\_supply\_chain.OrderList\_table\_new o

ON

f.Carrier = o.Carrier

GROUP BY

o.Plant\_Code;



#### Explanation of query logic

This query calculates the average freight weight range handled by each plant by joining the freight rates and order list tables on the Carrier field. For each Plant\_Code, it computes the average of the midpoint between Minm\_wgh\_qty and Max\_wgh\_qty from the freight rates data, giving an estimate of the typical shipment weight associated with each plant. The result is rounded to three decimal places and grouped by plant to reflect shipping capacity or trends at the plant level.

### 8. Total Warehouse Volume and Cost by Warehouse ID.

SELECT

w.Wh,

SUM(p.Daily\_capacity) AS total\_volume,

SUM(p.Daily\_capacity \* w.Cost\_per\_unit) AS total\_value

FROM

shipping-logistics-gcp.Shipping\_supply\_chain.WhCosts\_table w

JOIN

shipping-logistics-gcp.Shipping\_supply\_chain.WhCapacities\_table p

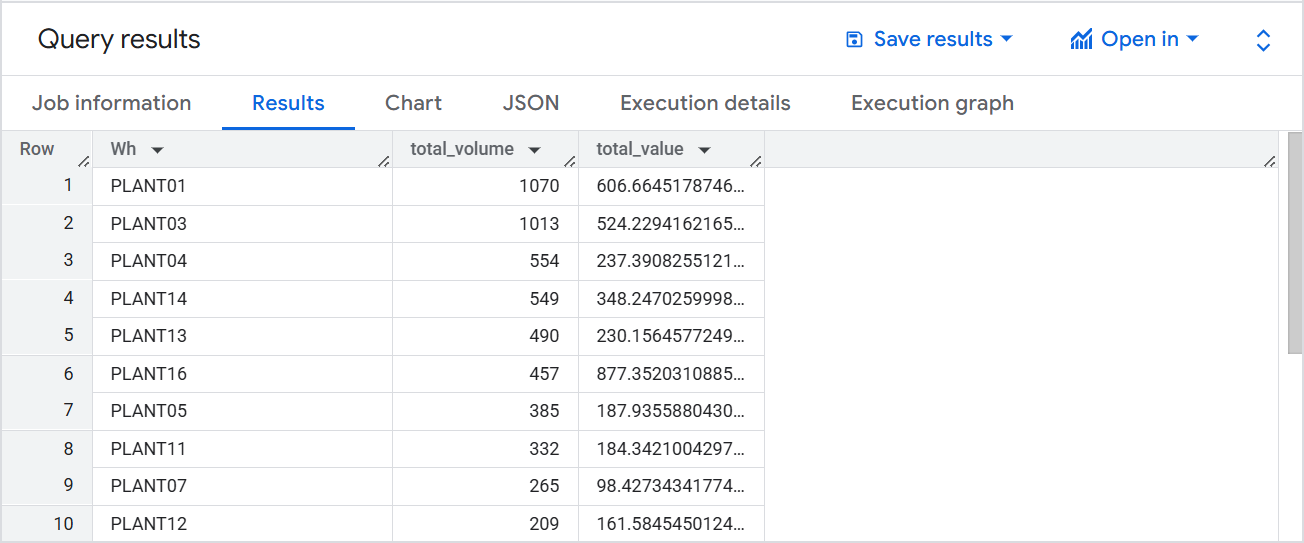
ON w.Wh = p.Plant\_Id

GROUP BY

w.Wh

ORDER BY

total\_volume DESC;



#### Explanation of query logic

This query evaluates warehouse performance by joining the warehouse cost and capacity tables using the warehouse identifier (Wh = Plant\_Id). For each warehouse, it calculates the total volume handled using SUM(p.Daily\_capacity) and the corresponding total value by multiplying daily capacity with Cost\_per\_unit. The result is grouped by warehouse and ordered by total volume in descending order, helping identify the most active and high-cost warehouses in the supply chain.

### 9. Product Count by Service Level.

SELECT

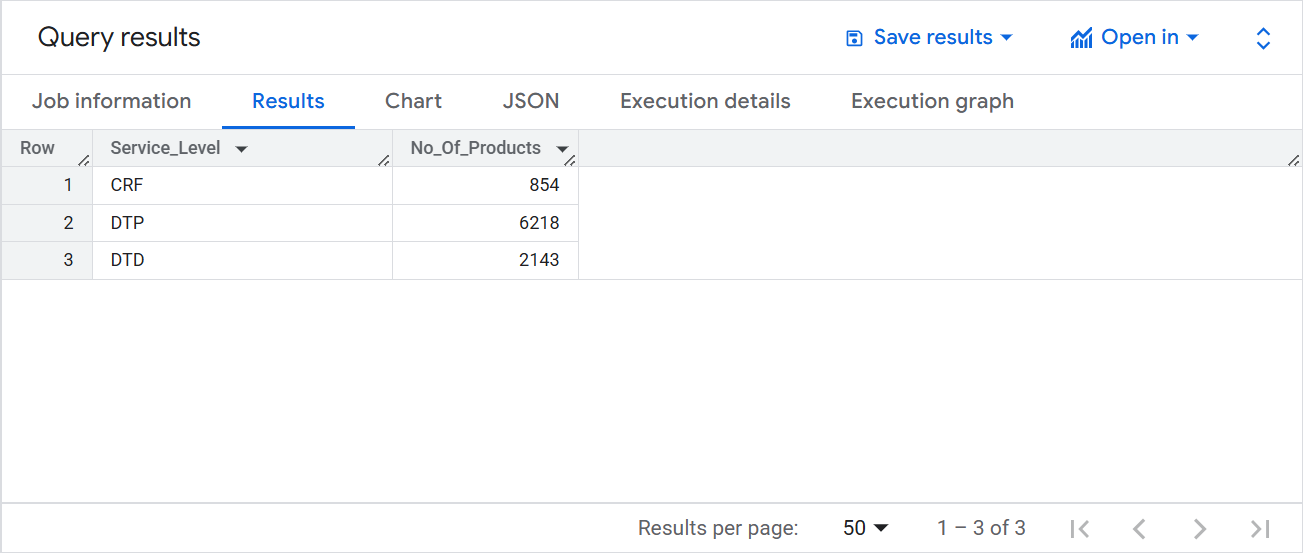
DISTINCT Service\_Level,

COUNT(Product\_Id) AS No\_Of\_Products

FROM `shipping-logistics-gcp.Shipping\_supply\_chain.OrderList\_table\_new`

GROUP BY

Service\_Level;



#### Explanation of query logic

This SQL query provides a summary of product orders grouped by service level. Groups the data by Service\_Level, so that the COUNT(Product\_Id) is calculated per service level.

### 10. Shipping Lead and Delay Days by Origin Port.

SELECT

DISTINCT Origin\_Port,

SUM(Ship\_Ahead\_DayCount) AS Avg\_Ship\_Ahead,

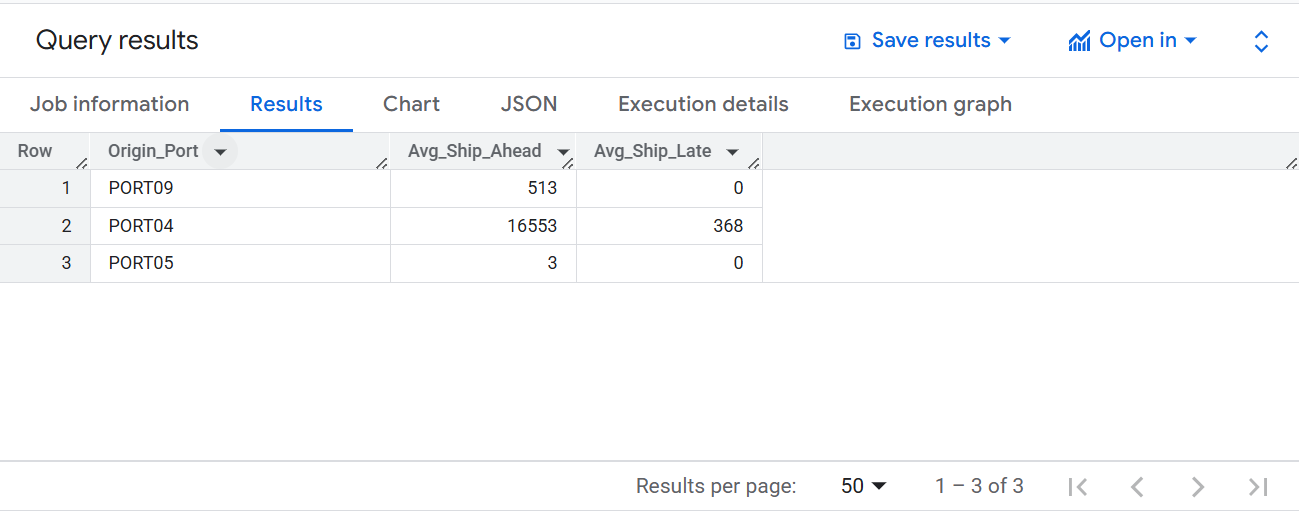
SUM(Ship\_Late\_DayCount) AS Avg\_Ship\_Late

FROM

shipping-logistics-gcp.Shipping\_supply\_chain.OrderList\_table\_new

GROUP BY

Origin\_Port;



#### Explanation of query logic

This SQL query analyzes shipping timeliness per origin port. It focuses on how early or late shipments are grouped by the port they originated from. This query gives a summary of shipping performance by origin port, specifically:

Total early shipment days (Ship\_Ahead\_DayCount)

Total late shipment days (Ship\_Late\_DayCount)

### 11. Freight Rate, Product Count, and Weight by Shipping Mode.

SELECT

f.Mode\_dsc,

Round(Avg(f.Rate),2) AS Rate,

Count(Distinct(o.Product\_Id)) AS No\_of\_Products,

Round(SUM(o.Weight),0) AS Total\_Weight

FROM

`shipping-logistics-gcp.Shipping\_supply\_chain.FreighRates\_table` f

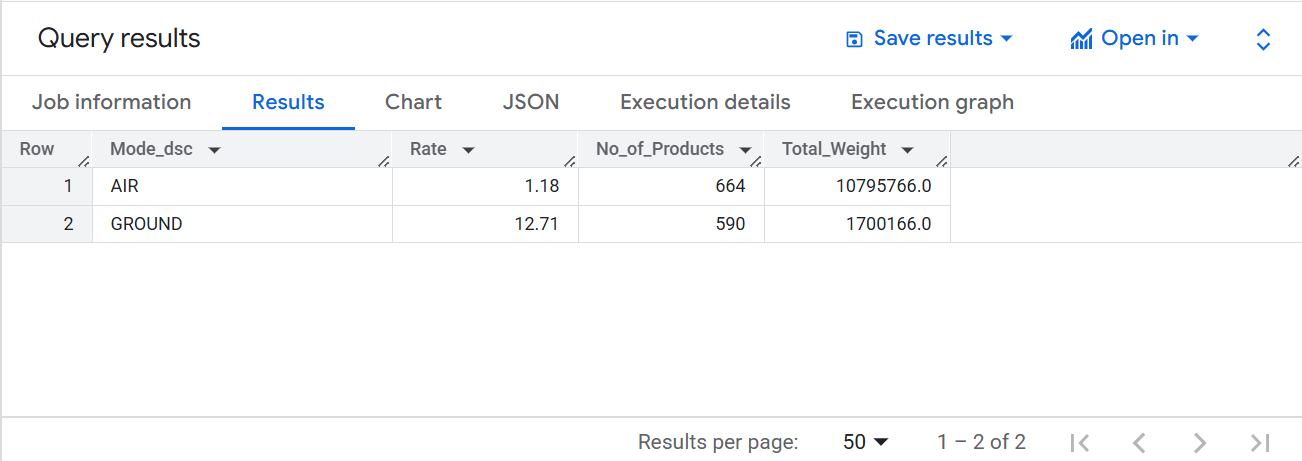
JOIN

`shipping-logistics-gcp.Shipping\_supply\_chain.OrderList\_table\_new` o

ON f.Carrier = o.Carrier

GROUP BY

f.Mode\_dsc;



#### Explanation of query logic

This SQL query analyzes freight shipping modes by combining data from two tables and provides a shipping mode performance summary, showing:

Average shipping rate per mode

Number of unique products shipped per mode

Total weight of shipments per mode

### 12. Order Count by Port of Shipment.

SELECT

p.Port,

COUNT(o.Order\_Id) AS Total\_Orders

FROM

`shipping-logistics-gcp.Shipping\_supply\_chain.PlantPort\_table` p

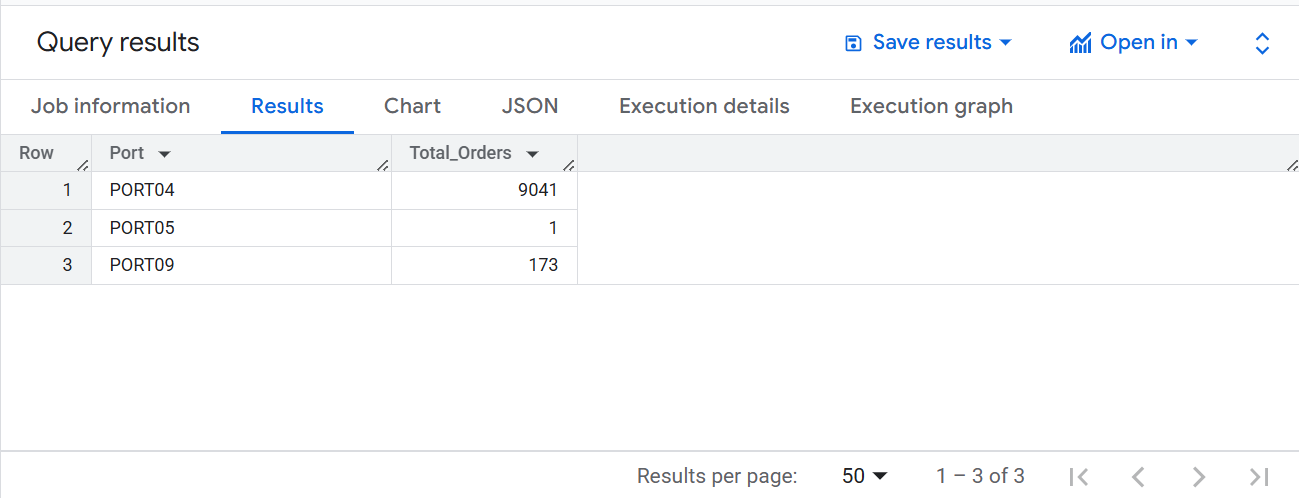
JOIN

`shipping-logistics-gcp.Shipping\_supply\_chain.OrderList\_table\_new` o

ON p.Plant\_Code = o.Plant\_Code

GROUP BY

p.Port;



#### Explanation of query logic

This SQL query is designed to summarize the total number of orders handled per port and returns a summary of how many orders are associated with each port, based on the plant-to-port mapping. It:

Joins order data with plant-port data via Plant\_Code, Groups results by port and finally Counts total orders per port.

### 13. Order and Weight Analysis for VMI Customers.

SELECT

v.Customers,

COUNT(o.Product\_Id) AS No\_Of\_Order,

ROUND(AVG(o.Weight), 2) AS Avg\_weight

FROM

`shipping-logistics-gcp.Shipping\_supply\_chain.VmiCustomers\_table` v

LEFT JOIN

`shipping-logistics-gcp.Shipping\_supply\_chain.OrderList\_table\_new` o

ON

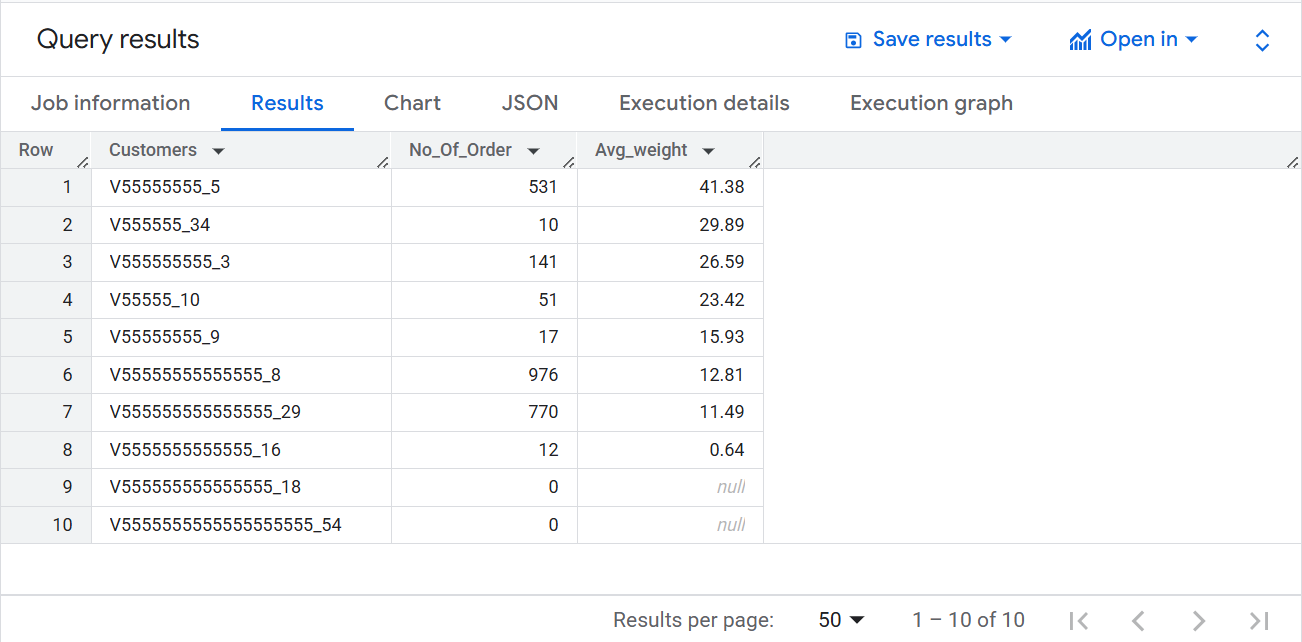
v.Customers = o.Customer

GROUP BY

v.Customers

ORDER BY

Avg\_weight DESC;



#### Explanation of query logic

This SQL query is designed to analyze customer orders by combining data from two tables and shows a customer-level summary of orders:

Total number of orders (No\_Of\_Order), Average weight of ordered products (Avg\_weight), Includes all customers, even those with zero orders, Results are sorted by average weight in descending order.

## 3. Functional,Volumetric Details & (PI) concerns :

This dataset contains seven interconnected tables which create a complete information system which supports analysis and decision-making for order shipment and warehouse logistics and transportation management. OrderList as the main dataset holds information on over 9,000 customer orders that include shipment origin ports and carrier details with product and plant identifiers and established service levels and weights. Shipping rate structure data stored in the FreightRates table serves customers alongside PlantPorts which defines allowed shipping routes between plants and ports. The ProductsPerPlant dataset helps analysts analyze warehouse-product matching potential for fulfillment purposes. Operational limits in the model exist through WhCustomers and WhCosts because they specify warehouse storage prices and daily shipment capabilities. VmiCustomers retains the unique restrictions between particular warehouses and their specific clientele. The dataset includes a mix of moderate to high volume tables each containing between 14 and over 9,000 rows with column maximums of 14. The dataset primarily delivers operational and configuration-based time coverage rather than historical time-series data. The main PII candidate fields are customer identifiers located among the data.

#### 1. FreightRates

* **a. Functional Meaning**: Describes available couriers, weight thresholds for shipping lanes, and associated cost/rate parameters.
* **b. Volume**: 1,540 rows × 11 columns.
* **c. Time Coverage**: Not explicitly stated but contains Tpt\_day\_cnt (transport days count), possibly indicating average transit durations rather than time span.
* **d. PII Check**: No direct PII. Data includes logistics metadata (Carrier, Origin Port), which is not person specific.

#### 2. OrderList

* **a. Functional Meaning**: Captures customer order details including origin, shipping info, customer/product/plant identifiers, and quantities.
* **b. Volume**: 9,215 rows × 14 columns.
* **c. Time Coverage**: Contains a column named order date which shows the date of order (as given in the data, it is of 1 day only) and time-related fields like Ship ahead, Ship late day count.
* **d. PII Check**:
  + **Customer ID**: Potential PII (if traceable to individual).
  + Other columns like Order ID, Product ID, Plant Code are not PII.

#### 3. PlantPorts

* **a. Functional Meaning**: Defines real-world links between plants and ports (for shipment routing).
* **b. Volume**: 22 rows × 2 columns.
* **c. Time Coverage**: Static mapping; likely doesn’t vary over time.
* **d. PII Check**: No PII. Only geographic or logistic mapping info.

#### 4. ProductsPerPlant

* **a. Functional Meaning**: Lists all valid warehouse-product combinations.
* **b. Volume**: 2,036 rows × 2 columns.
* **c. Time Coverage**: Presumably static or slowly changing master data.
* **d. PII Check**: No PII.

#### 5. VmiCustomers

* **a. Functional Meaning**: Specifies special warehouse-customer relationships where only listed customers can be served by certain warehouses.
* **b. Volume**: 14 rows × 2 columns.
* **c. Time Coverage**: Likely static or rare changes.
* **d. PII Check**:
  + **Customer ID** could be PII depending on context.
  + Otherwise, no direct PII.

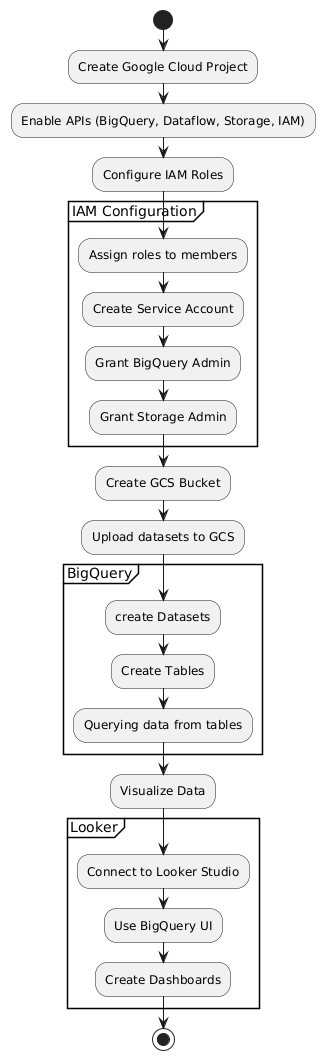
#### 6. WhCustomers

* **a. Functional Meaning**: Warehouse capacity constraints (e.g., number of orders per day).
* **b. Volume**: 19 rows × 2 columns.
* **c. Time Coverage**: Static or reflects daily operational limits.
* **d. PII Check**: No PII.

#### 7. WhCosts

* **a. Functional Meaning**: Specifies cost to store products in warehouses (cost per unit).
* **b. Volume**: 19 rows × 2 columns.
* **c. Time Coverage**: Likely static cost data.
* **d. PII Check**: No PII.

## 4. Methodology



### GCS:

In this project, Google Cloud Storage (GCS) was used as the primary data lake to store raw datasets from multiple sources. Data was ingested in various formats such as CSV, JSON, and Parquet and securely stored in GCS buckets. The scalable nature of GCS allowed efficient handling of large volumes of structured and unstructured data with minimal latency. GCS also played a critical role in the data pipeline by acting as the staging area for data preprocessing before loading into BigQuery for analytics. Using Cloud Functions and scheduled workflows, files uploaded to GCS triggered automated processes like validation, transformation, and metadata extraction. This ensured a seamless and cost-effective integration with downstream tools and services in the data ecosystem.

As part of the analytics workflow, we used BigQuery to perform advanced data analysis and querying. Specifically, we uploaded 7 files in CSV format into BigQuery for processing and analysis.

### IAM ROLES

Identity and Access Management (IAM) is a system in Google Cloud Platform (GCP) that allows administrators to manage who (users) has what access (roles) to which resources. IAM enables fine-grained access control by assigning roles to members, helping ensure that the principle of least privilege is followed. Roles can be basic (Owner, Editor, Viewer), predefined (designed for specific GCP services), or custom (user-defined). Each role is a collection of permissions that grant access to perform specific actions on GCP resources.

#### IAM Roles Used:

1. **BigQuery Data Owner** – Grants full access to all BigQuery resources, including the ability to read and write datasets, create jobs, and manage permissions.
2. **Editor** – Provides broad permissions to modify most GCP resources within a project, excluding the ability to manage IAM policies.
3. **Service Account User** – Allows a user to impersonate and act as a service account, typically used in automated workflows and pipelines.
4. **Storage Object Admin** – Grants full control over objects in Cloud Storage, including uploading, deleting, and listing files within buckets.

### Dataflow:

Google Cloud Dataflow is a fully managed, serverless data processing service designed for both batch and real-time streaming data pipelines. It is based on the Apache Beam programming model, allowing developers to create complex data workflows that are portable across different execution engines. Dataflow also provides built-in monitoring and logging through Cloud Monitoring and Cloud Logging, aiding in observability and debugging. Its strong support for event time processing and exactly once semantics makes it ideal for analytics, ETL, and machine learning pipelines. In essence, Dataflow offers a powerful, scalable, and flexible solution for data engineering tasks in modern cloud environments.

In our project, Google Cloud Dataflow played a central role in automating the ingestion of CSV data from Google Cloud Storage (GCS) into BigQuery. We utilized two pre-built Dataflow templates—"Text Files to BigQuery" and "CSV Files to BigQuery"—to simplify and accelerate the pipeline development. These templates allowed us to define schema mappings and delimiters, enabling seamless parsing and transformation of structured CSV data. Dataflow handled the orchestration, including reading files from GCS, converting the data into table-ready format, and writing it to the target BigQuery tables with high reliability. We defined a schema for each of those 7 files to ensure integrity during the load process. The serverless nature of Dataflow ensured automatic scaling based on the size and number of input files, reducing the need for manual resource provisioning.

### BigQuery:

Google Cloud BigQuery is a fully managed, serverless data warehouse designed for fast and scalable analysis of large datasets using SQL. It enables users to run ad hoc queries on petabytes of data with high performance, thanks to its columnar storage and Dremel-based execution engine. BigQuery supports standard SQL and offers features like automatic data replication, backup, and encryption for robust data security and availability. It integrates seamlessly with various Google Cloud services such as Cloud Storage, Dataflow, Pub/Sub, and Looker, enabling end-to-end data analytics workflows. BigQuery's support for federated queries allows analysis of data across external sources like Cloud Storage and Google Sheets without data movement. BigQuery’s pay-per-query and flat-rate pricing models provide flexibility for different use cases. Overall, BigQuery provides a powerful, scalable, and easy-to-use platform for modern data analytics.

BigQuery served as the central platform for storing and analysing the integrated data from seven different CSV datasets. After ingesting the data into individual BigQuery tables, we leveraged BigQuery’s SQL interface to perform data joins based on common keys and relationships across the datasets. This allowed us to create a unified view of the data, which formed the foundation for deeper analytics. Using standard SQL, we calculated derived fields, applied filters, and created aggregate summaries to extract meaningful insights. BigQuery’s powerful processing engine enabled us to run complex queries efficiently, even on large volumes of data. We also used its capabilities for data exploration, trend analysis, and generating summary tables to support decision-making. The interactive query environment and integration with visualization tools further helped in analysing patterns and drawing business conclusions.

### Looker

Looker was utilized as the primary business intelligence and data visualization tool in this project to derive actionable insights from the processed data in BigQuery. After integrating the data sources, we connected Looker to BigQuery using the appropriate project and dataset configurations, including the use of the “recently used project” and custom SQL queries to explore the data efficiently.

We worked with two distinct data sources and created 13 analytical queries, each tailored to address specific business questions. For each query, we selected the most appropriate visualization type to represent the data effectively. The visualizations included bar charts, column charts, tables with heatmaps, pie charts, line charts, and scatter plots.

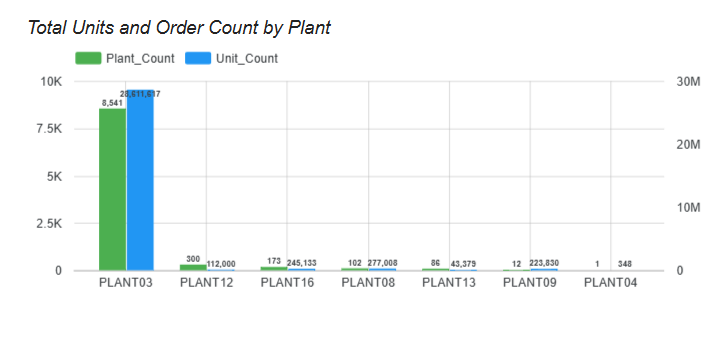
For example, bar charts were particularly effective in comparing categorical data across different groups, while pie charts provided a quick snapshot of percentage distributions. Tables with heatmaps were useful for highlighting key variations in numerical values, and scatter plots helped uncover correlations between variables.

This strategic use of Looker allowed us to turn raw data into meaningful dashboards, facilitating data-driven decision-making and ensuring stakeholders could interact with and interpret the data intuitively.

## 5. Query-by-Query Findings & Dataset Mappings

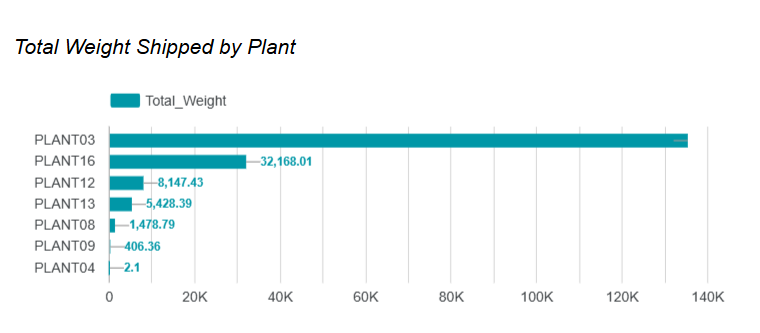
#### 1) Data Insights of Total Units and Order Count by Plant

The chart shows that **PLANT03** is the dominant contributor with 8,541 orders and over 26 million units, indicating it functions as a central hub. In contrast, **PLANT08, PLANT09, and PLANT16** handle significantly fewer orders but ship very high quantities, suggesting their role in bulk or specialized logistics. **PLANT12 and PLANT13** show moderate activity with relatively lower unit efficiency per order. **PLANT04** has minimal presence, possibly indicating limited or specialized use. Overall, the data highlights operational differences across plants, with opportunities to optimize unit efficiency in some locations.



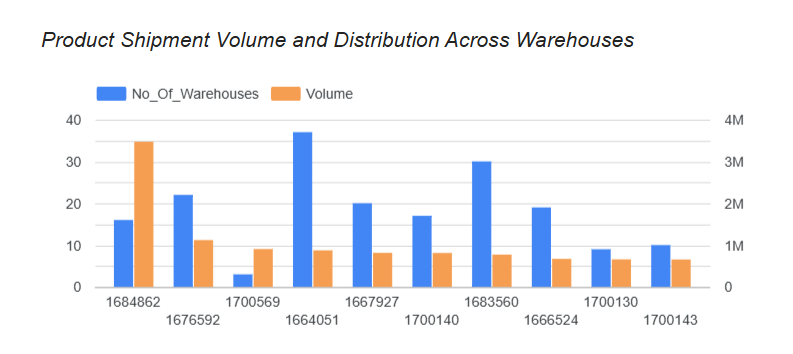
#### 2) Data insights of Total Weight Shipped by Plant.

The bar chart for **Total Weight Shipped by Plant** shows that **PLANT03** is the leading contributor, shipping nearly 140K units of weight, far surpassing all others. **PLANT16** follows with around 32K, indicating a significant but much smaller role. Plants like **PLANT12, PLANT13, and PLANT08** contribute moderately, each shipping between 1K to 8K in total weight. Meanwhile, **PLANT09 and PLANT04** have minimal shipping weights, with PLANT04 showing virtually no activity. This highlights PLANT03’s dominant role in logistics weight distribution, with potential efficiency improvements for lower-performing plants.



#### 3) Data insights of Product Shipment Volume and Distribution Across Warehouses.

The analysis reveals a diverse product distribution landscape. Product 1684862 (and 1676592) shows the highest volume, while 1664051 (and 1700569) leads in the number of warehouses, indicating varying strategies from concentrated high-volume distribution to widespread availability. The SQL query aggregates Unit\_Quantity and counts Plant\_Code by Product\_Id, clearly showing top products by volume and identifying those with extensive distribution networks, even if their individual volumes are lower. The data suggests no consistent correlation between volume and warehouse count across all products.



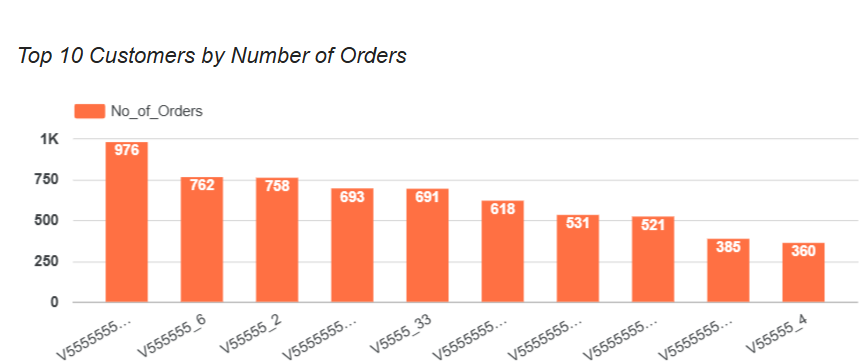
#### 4) Data insights of Average Freight Rate and Total Quantity by Carrier.

The data provided shows two carriers with distinct profiles: V444\_0 handles significantly higher volumes (921M units) at a higher average freight rate (4.597), indicating it's the primary, possibly premium, carrier. In contrast, V444\_1 moves lower volumes (610M units) at a much cheaper rate (1.472), suggesting a cost-effective alternative. The SQL query effectively aggregates total unit quantity and average freight rates by carrier, confirming these operational differences and highlighting the trade-off between cost and volume across the two carriers.

#### 

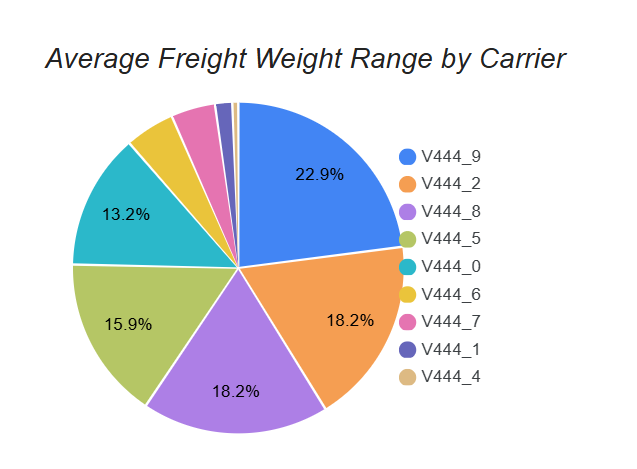
#### 5) Data insights of Top 10 Customers by Number of Orders.

The provided data clearly identifies the top 10 customers by order volume, with V5555555555\_8 leading significantly at 976 orders, showcasing a highly engaged customer base. A noticeable tiered structure exists, where the top 5 customers consistently place over 600 orders, while the latter half of the top 10 ranges from 360 to 531 orders. This strong repeat business indicates high customer loyalty, providing an opportunity for targeted strategies like enhanced loyalty programs for the top tier and engagement initiatives for those with fewer orders. The precise SQL query efficiently extracts this crucial customer activity data, enabling a focused approach to customer relationship management.



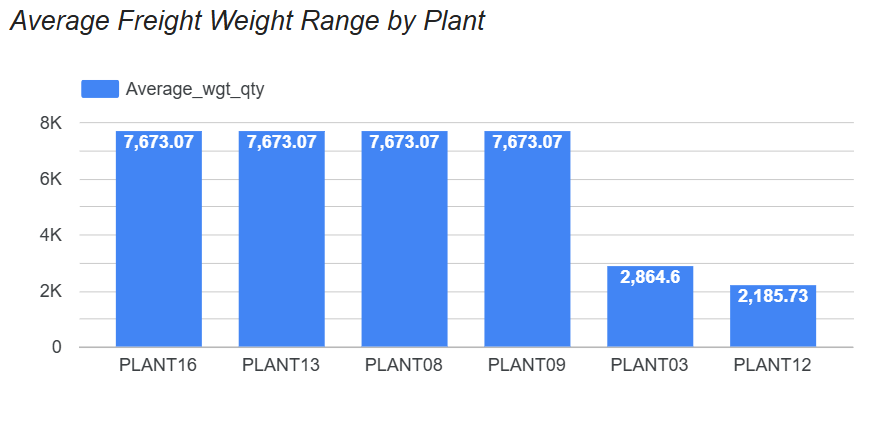
#### 6) Data insights of Average Freight Weight Range by Carrier.

The analysis of average freight weight ranges by carrier reveals distinct specializations. Carrier V444\_9 handles the largest average weight quantity (13,299.995), dominating with 22.9% of the total average weight, closely followed by V444\_2 and V444\_8 (both at 10,569.995), each contributing 18.2%. Conversely, V444\_4 specializes in the lightest shipments (337.895), representing the smallest segment. This distribution, derived from the SQL query calculating the average of the minimum and maximum weight quantities, indicates that different carriers are optimized for varying shipment sizes, allowing for strategic selection based on cargo weight.



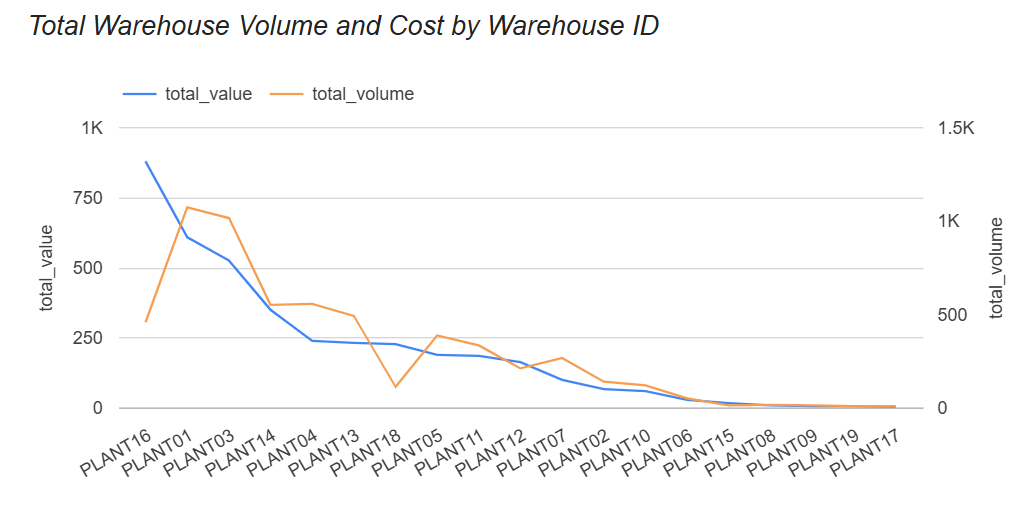
#### 7) Data insights of Average Freight Weight Range by Plant.

Four plants (PLANT16, PLANT13, PLANT08, PLANT09) have a consistently high average freight weight of 7,673.07, indicating they handle the bulk of freight volume. In contrast, PLANT03 and PLANT12 show significantly lower averages at 2,864.60 and 2,185.73, respectively. This highlights a clear division into two groups: high and low average freight weight plants. PLANT12 handles the least, at just 28% of the top group's average.



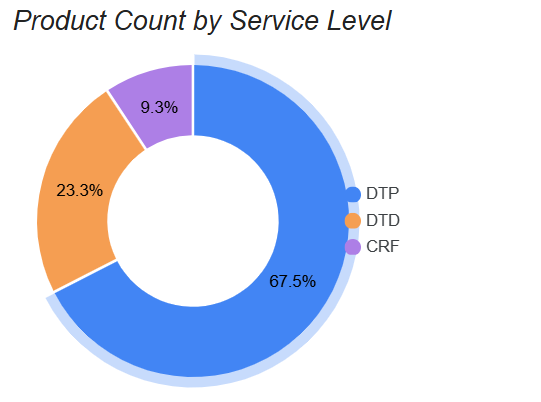
#### 8) Data insights of Total Warehouse Volume and Cost by Warehouse ID.

PLANT01 and PLANT03 are the top performers, leading in both total value and volume, far ahead of other warehouses. PLANT01 has the highest value, while PLANT03 has the highest volume. A sharp drop-in activity follows, with PLANT04 and PLANT14 showing moderate performance. Most warehouses show a positive correlation between volume and value, though PLANT16 stands out with high value despite lower volume, indicating high-value goods. PLANT01’s higher value per unit compared to PLANT03 reinforces this insight. Warehouses like PLANT15, PLANT09, and PLANT19 show minimal activity, possibly indicating specialized or inactive operations.



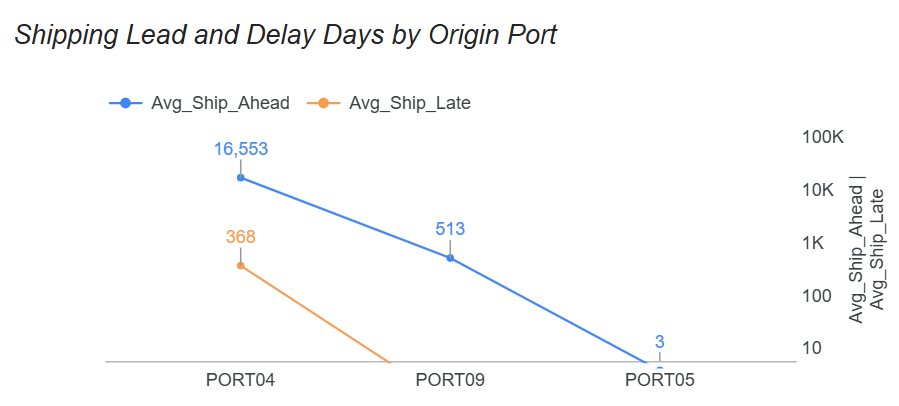
#### 9) Data insights of Product Count by Service Level.

DTP (Door-to-Plant) is the dominant service level, covering 67.5% of products (6,218), making it the most widely used and likely the standard for most logistics' operations. DTD (Door-to-Door) follows with 23.3% (2,143 products), serving as a significant secondary option, possibly for more direct deliveries. CRF (Customer-Requested Freight) is the least used, accounting for just 9.3% (854 products), suggesting it is reserved for specific, customized logistics needs. This distribution highlights a clear preference for standardized shipping models over flexible, on-demand services.



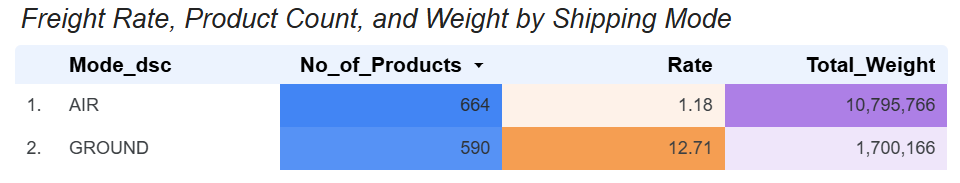
#### 10) Data insights of Shipping Lead and Delay Days by Origin Port.

PORT04 is a clear outlier, with an exceptionally high average of 16,553 days shipped ahead and the only port with delays, averaging 368 days late—indicating inconsistent or complex operations. In contrast, PORT09 shows strong performance with 513 days shipped ahead and zero delays, reflecting efficient and reliable scheduling. PORT05 has minimal lead time, averaging just 3 days ahead, suggesting low activity or tight scheduling, but it also has no delays. Overall, PORT09 emerges as the most consistent and reliable, while PORT04 highlights operational variability.



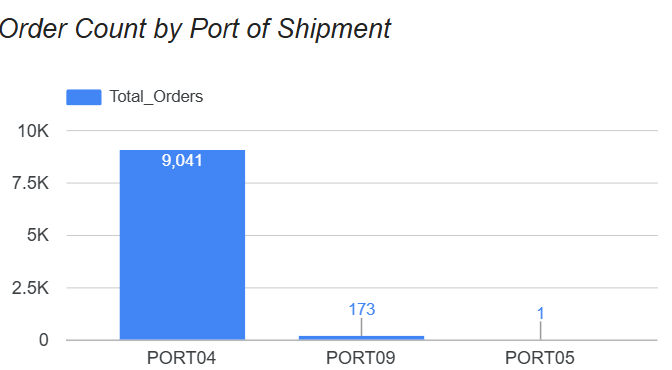
#### 11) Data insights of Freight Rate, Product Count, and Weight by Shipping Mode.

AIR shipping handles more products (664) and significantly more total weight (10.8 million units) than GROUND (590 products, 1.7 million units). Despite this, AIR has a much lower rate per unit (1.18 vs. 12.71), indicating cost efficiency for large, heavy shipments. GROUND shipping, while handling lighter loads, has a much higher rate, possibly due to more frequent, smaller deliveries or a different cost structure. This suggests AIR is optimized for bulk, heavy freight, while GROUND suits lighter, smaller shipments.



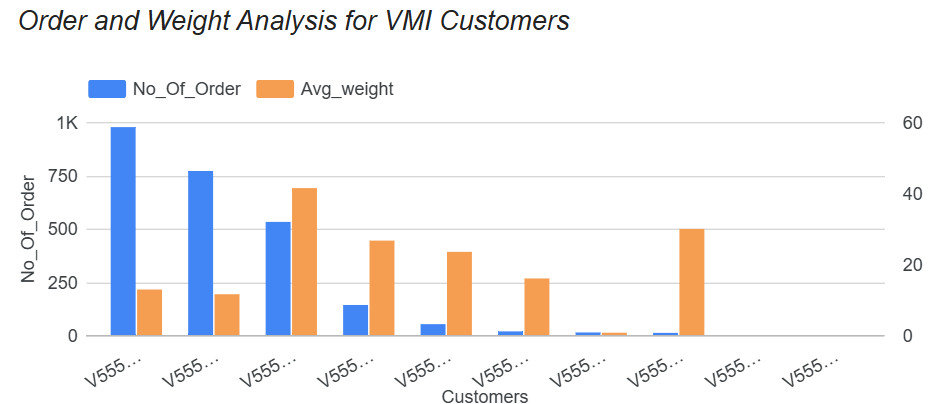
#### 12) Data insights of Order Count by Port of Shipment.

PORT04 is the dominant logistics hub, handling 9,041 orders—about 98.1% of total volume—making it the primary port by far. PORT09 is a minor contributor with 173 orders (around 1.9%), while PORT05 is nearly inactive, processing just 1 order (0.01%). The stark difference in order of volume suggests PORT04 is central to operations; PORT09 has limited but notable use, and PORT05 plays an almost negligible role, possibly reserved for special cases.



#### 13) Data insights of Order and Weight Analysis for VMI Customers.

Customers V55555555555558 and V555555555529 have the highest order counts (976 and 770) with low average weights, indicating frequent, small shipments. V555555555\_5 follows with high order volume (531) and moderate weight (41.38), suggesting significant activity. Mid-range customers like V555555555\_3 and \_10 place fewer orders with moderate weights. Low-frequency customers like V555555555\_34 and \_9 tend toward heavier and bulk shipments. V555555555\_16 has minimal activity with very small shipments, while V5555555555555\_18 and \_54 have placed no orders, showing zero engagement.



## Conclusion

Through this project, we successfully tackled a complex shipping logistics and supply chain problem by leveraging Google Cloud Platform (GCP) tools for data storage, analysis, and visualization. By starting with the creation of a GCP project, IAM roles were appropriately granted to ensure secure access across team members. Data stored in Google Cloud Storage (GCS) was efficiently uploaded into BigQuery for querying, which allowed us to derive actionable insights. We used BigQuery’s powerful query capabilities to aggregate and summarize data, combining multiple datasets to uncover key trends. For visualization, Looker Studio provided an intuitive interface to create interactive dashboards, making it easier for stakeholders to interpret the findings. Ultimately, this project enabled effective data-driven decision-making, streamlined data handling, and fostered collaboration across the team. As a result, we created a dashboard and presented relevant insights for informed decision-making.