VIETNAM NATIONAL UNIVERSITY, HO CHI MINH CITY
HO CHI MINH UNIVERSITY OF TECHNOLOGY

DATA ENGINEER (CO5240)

Group Project Report

Building a Customer Data Platform with Google BigQuery and Apache Spark

---

Students: Nguyen Duc Thuy – 2012158
Ten – MSSV

Ho Chi Minh City - November, 2024

Contents

1 Introduction

1.1 Market Growth and Transformation

The retail and e-commerce landscape in Vietnam is undergoing unprecedented growth and transformation, driven by rapid advancements in technology, shifting consumer behaviors, and increased internet penetration. Traditional retail outlets, which once relied solely on brick-and-mortar operations, are now embracing digitalization to enhance customer experiences and streamline processes. Simultaneously, pure-play e-commerce platforms are aggressively expanding their market presence, introducing new features, and leveraging data-driven strategies to cater to the growing demand for online shopping.

This evolution has resulted in an exponential increase in the volume of sales data generated daily, encompassing a wide range of information such as transaction records, customer preferences, and market trends. For businesses, this surge in data presents significant opportunities to optimize operations, personalize marketing efforts, and make informed strategic decisions. However, it also brings challenges related to data management, analysis, and security, requiring organizations to invest in advanced technologies and skilled talent to unlock the full potential of this information. As a result, the interplay between traditional retail and e-commerce continues to shape a dynamic and competitive market landscape in Vietnam.

1.2 Solution Vision

Our proposed Data Warehouse solution represents a transformative approach to data analytics in the retail sector. At its core, the solution aims to create a comprehensive data analytics platform that revolutionizes how businesses handle and extract value from their data assets. Through centralized data management, the platform consolidates disparate data sources into a single source of truth, eliminating data silos and ensuring consistency across all business operations. The solution incorporates advanced analytics capabilities, leveraging cutting-edge technologies like machine learning and predictive modeling to uncover deep insights from complex datasets. Real-time insight generation capabilities enable businesses to respond swiftly to market changes and customer behaviors, providing immediate actionable intelligence for decision-makers. The platform's scalable processing infrastructure, built on modern cloud technologies, ensures the solution can grow seamlessly with the business, handling increasing data volumes and processing demands without compromising performance. This comprehensive approach not only addresses current data management challenges but also positions organizations to capitalize on future opportunities in the rapidly evolving retail landscape.

1.3 Expected Benefits

The implementation of a new data management system promises numerous operational benefits. By streamlining data processing, organizations can handle data more efficiently, reducing the time required for data-related tasks and leading to faster access to essential information. Additionally, this system enhances data accuracy, minimizing errors that could otherwise lead to flawed analyses or misguided decisions. Alongside this, improved data security measures help to protect sensitive information, fostering trust within the organization and with external stakeholders.

On a broader level, the system also supports significant business benefits. By enhancing customer understanding, companies can tailor their offerings to meet client needs more effectively, resulting in higher satisfaction and loyalty. The system's capacity to improve decision-making capabilities

means that leadership teams can rely on more accurate insights, guiding better strategic choices. Furthermore, enhanced marketing effectiveness stems from a deeper understanding of customer preferences, which, combined with increased operational efficiency, allows for more resourceful use of both time and finances.

From a strategic perspective, the new system offers long-term benefits essential for future competitiveness. Leveraging data-driven insights, organizations gain a competitive advantage by responding to market trends and customer demands more proactively. This system also enables improved market responsiveness, allowing businesses to stay ahead of industry shifts. Enhanced customer satisfaction aligns with the strategic goal of building lasting relationships, while a future-ready infrastructure ensures that the organization can adapt to evolving technological needs, supporting sustained growth and innovation.

2 Business Requirements and Big Data Characteristics

In the retail and e-commerce context, this project primarily focuses on e-commerce sales, customer service, and marketing. As a result, the sales and marketing teams, along with stakeholders and managers, are the primary users of the data. Based on the organization's operational needs and strategic objectives, we identified several key business requirements:

- Data-driven decision-making: The organization needs to be able to analyze large datasets quickly and efficiently to make informed decisions that improve customer satisfaction, increase revenue, and enhance operational efficiency.
- Real-time insights: Decision-makers require up-to-date information on sales performance, inventory levels, and customer sentiment to act quickly and adjust strategies when needed.
- Data visualization: Stakeholders at various levels of the organization need intuitive and actionable insights, which can be best achieved through clear data visualizations and dashboards.
- Scalability: As the organization expands, its data storage and processing requirements will grow, requiring a technology solution that can scale seamlessly to handle increasing data volumes.

To address these business requirements, we assessed the following big data characteristics:

- Volume: The organization generates a massive amount of data daily, from transactional data in sales to the grow in quantity of customer and product records as the organization continues to scale out. This requires a solution that can store and process large datasets without performance degradation
- Velocity: The speed at which data is generated, particularly from sales transactions, inventory updates, and customer interactions, requires real-time processing and immediate insights for decision-making.
- Value: The organization needs to extract meaningful insights from large datasets to drive revenue growth, improve customer retention, and optimize operations. Without value-driven insights, big data would be just raw information.
- Veracity: Due to various source of information and method of data collection, the data can observe abnormalities and noise. To provide valuable insights for sale and marketing activities, the need to ensure veracity is stressed as biased insights can lead to business decline.

3 Scope of Work

3.1 Input

The primary input for this data warehouse solution consists of comprehensive sales activities data, encompassing three main data categories: products, customers, and transactions. These data streams form the foundation for the analytical services and represent the core business activities that need to be processed and analyzed.

Product data includes detailed information across 73 different dimensions, capturing essential attributes such as product descriptions, brand information, manufacturing dates, pricing, and current status. This extensive product dataset comprises approximately 700,000 entries per CSV file, providing a rich source of information for product-related analytics and inventory management insights.

Customer data is even more extensive, spanning 87 dimensions that encompass crucial customer information including personal details, geographical locations, demographic background, loyalty membership status, and contact information. With over 4.7 million entries, this customer dataset provides a comprehensive view of the customer base, enabling detailed customer segmentation and personalized analysis. Additionally, transaction data is structured in a header-lines table design pattern, where sale headers contain 21 dimensions of invoice-level information, and sale lines include 26 dimensions of item-level transaction details.

3.2 Output

The data warehouse solution is designed to deliver three primary outputs that provide significant business value: analytics dashboards, consulting services, and customer recommendations. These outputs are carefully crafted to transform raw sales data into actionable insights and valuable business intelligence.

The analytics dashboards serve as a comprehensive visualization tool that leverages the various dimensions of each data file to present a holistic view of the business. These dashboards are specifically designed to be human-readable and enable easy analysis of sales performance, key performance indicators (KPIs), and other critical business metrics. Through these interactive dashboards, stakeholders can effectively monitor business performance and identify trends or patterns in their sales data.

The system also provides customer recommendations through its near real-time processing capabilities. Each customer transaction is collected and processed to continuously update the Customer Data Platform (CDP), enabling a more comprehensive single view of the customer. Additionally, the solution offers consulting services that go beyond descriptive analytics by incorporating machine learning techniques for predictive analysis. These advanced analytics capabilities help generate deeper business insights, enabling more informed decision-making and strategic planning.

4 Solution Architecture

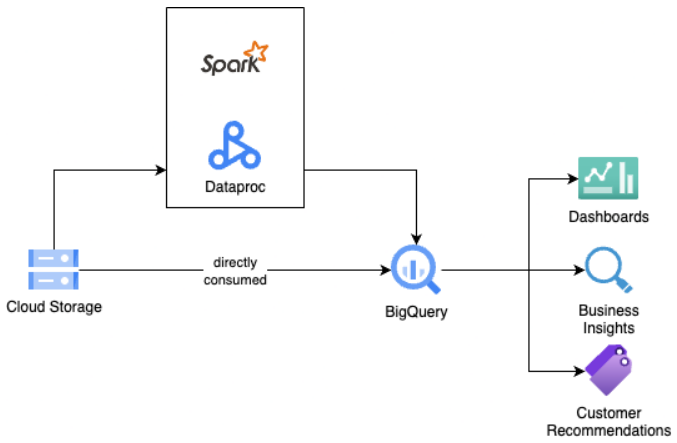


Figure 1: Overview of Solution Architecture

As a high-level objective, our team wish to build a dataflow, moving raw data provided by the retailers into the data warehouse, for which we choose Google BigQuery. After that, within the warehouse, we perform some transformation to extract value from the retailer inputs. The first one would be a set of recommendation tag, so called **Personal Product Ranking** or PPR. Then, we will visualizing the sale data over the period of the provided time.

To be specific, we will assume that raw data files are uploaded by the retailers to an Object Storage. Using Google Cloud ecosystem, Cloud Storage is used as the endpoint. About our team architecture of the solution, we wish to compare between the commonly used Apache Spark for ETL data from raw files into BigQuery, versus using native BigQuery commands to ingest data. As a result, the ingestion step from Cloud Storage into BigQuery will be run in two ways. Finally, recommendations for customers will be stored in a table.

For visualization, to reduce complexity and to promote the full use of BigQuery ecosystem,

5 Technology Stack

5.1 Google BigQuery

Figure 2: Google BigQuery

Google BigQuery is a fully managed, serverless data warehouse service that operates on the Google Cloud Platform. It is designed for real-time analytics and the ability to handle large-scale datasets with high efficiency and speed. BigQuery allows users to run complex SQL queries on vast amounts of data, offering unparalleled performance without needing to manage infrastructure. Key features of BigQuery include:

- Scalability for large datasets: BigQuery can efficiently handle petabytes of data, making it suitable for enterprises with massive datasets that need to be processed and analyzed in real-time.
- Seamless integration with Google Cloud ecosystem: BigQuery integrates easily with other Google Cloud services, such as Google Data Studio, Google Analytics, and Google Machine Learning services, enabling a streamlined workflow for data analytics.
- SQL-based querying: Users can leverage SQL to write queries, making it accessible for data analysts who may not have deep programming expertise.
- Cost-effective pricing: With BigQuery, organizations only pay for the queries they run and the storage they use, eliminating the need to invest in hardware or manage large data centers.
- Real-time data analytics: BigQuery supports real-time data processing, making it suitable for environments where up-to-the-minute analytics are critical.

5.2 Apache Spark

Apache Spark is an open-source distributed computing framework that is widely used for large-scale data processing. Unlike traditional batch processing, Spark enables both batch and stream processing and is known for its speed and flexibility in handling large datasets. Key features of Apache Spark include:

- In-memory processing: Spark processes data in-memory, significantly boosting speed compared to traditional disk-based data processing systems. This makes it an ideal choice for iterative machine learning tasks and large-scale data transformations.
- Versatility: Spark supports a variety of programming languages, including Python, Java, Scala, and R, and can integrate seamlessly with other big data tools like Hadoop and Kafka.

Figure 3: Apache Spark

- Distributed computing: Spark is designed to run on clusters of machines, enabling it to scale easily to process petabytes of data.
- Machine Learning support: Apache Spark provides a built-in machine learning library, MLlib, which allows users to easily apply machine learning algorithms on large datasets.
- Real-time streaming: Spark can process real-time streaming data with Spark Streaming, making it an excellent choice for applications requiring low-latency processing and data analysis.

Together, Google BigQuery and Apache Spark offer a comprehensive suite for big data processing. BigQuery excels in data warehousing and quick SQL queries, while Apache Spark provides the flexibility and power for complex, real-time data analytics and machine learning tasks. These technologies are highly complementary and allow organizations to scale their data infrastructure and analytics capabilities as needed.
In the e-commerce context, Google BigQuery was selected for its scalability, enabling the organization to handle large volumes of daily sales data efficiently. Its seamless integration with the Google Cloud ecosystem and support for various query engines make it accessible to analysts with diverse coding skills at a cost-effective price along with the ability to adapt new business requirements in future. Additionally, Apache Spark was chosen for its speed and flexibility in real-time streaming, a critical business requirement in this business context.

6 Implementation Result

6.1 Ingest data directly with BigQuery

6.1.1 Prerequisites and Setup

- Google Cloud Storage (GCS) Bucket: This is where your CSV file is stored. Ensure the file is in a publicly accessible location or that you have the necessary permissions to access it.
- BigQuery Dataset: Your target dataset in BigQuery should already be created. If not, create one by navigating to the BigQuery console, clicking on your project, and selecting Create Dataset. You will be prompted to specify a dataset ID, data location, and any expiration date or encryption preferences.

6.1.2 Load data with BigQuery

Big Query support loading CSV data from Cloud Storage in multiple options such as SQL, Console and more. In this project, our team choose to use bq command-line tool of Big Query to load data into table in Big Query. The reason of this choice lies mainly in the fact that Big Query supports auto detection for CSV file, which ,as the name suggests, automatically detect the schema and data type. Auto detection leads to the conflict between detected data type and our desired schema or not so well-formatted data. Unlike SQL (main tools we use for ingest data with BigQuery), bq command-line tool supports option for turning off auto detection. The bq command we use is as below:

```
bq  load --replace --source_format=CSV --autodetect=false --field_delimiter='|'
    --skip_leading_rows=1  --max_bad_records=1000 --quote "" bigquery_direct.
   saleheader_raw_loading  'gs://retailer-raw-ingest-data/
   SalesHeaderFCT_inc_20240929.txt'
```

The options are listed as:

- –replace: To erase any existing data and schema when new data is loaded
- –autodetect=false: Turn off schema auto detection
- –field_delimiter='—' : Specifies the character that marks the boundary between columns in the data
- –skip_leading_rows=1: Skip header row
- –max_bad_records=1000: Skip bad record (such as row missing columns in CSV data)
- –quote "": Specify that there is no quote character to surround fields in CSV data

6.1.3 From Cloud Storage to the main table in BigQuery

Although we can directly load data into table in BigQuery, the format of some of our data do not match with one default format in BigQuery, especially with DATE and DATETIME data type, moreover some columns ' majority values are NULL, hence our team decide to load in temporary tables before transform it into correct data format in main tables with column having small proportion of NULL.
In this scope of project, we call the second tables used for directly loaded data temporary tables to emphasize that they only hold data for a short time. However the tables are just normal tables,

not temporary tables BigQuery provides. The temporary tables have all the columns in CSV files, with data type all set to string for the ease of load every data format.

Filter Enter property name or value

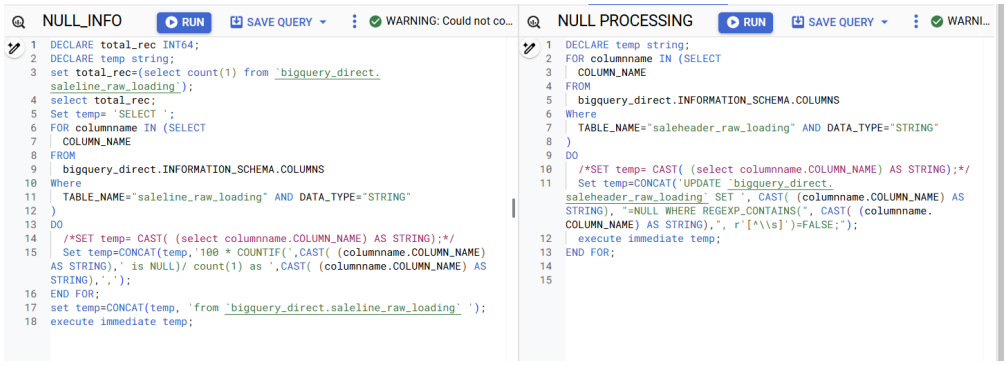
Field name	Type	Mode	Key	Collation	Default Value	Policy Tags	Description
CUSTOMER_CODE	STRING	NULLABLE	-	-	-	-	-
CUSTOMER_SALUTATION	STRING	NULLABLE	-	-	-	-	-
FULL_NAME	STRING	NULLABLE	-	-	-	-	-
FIRST_NAME	STRING	NULLABLE	-	-	-	-	-
MIDDLE_NAME	STRING	NULLABLE	-	-	-	-	-
LAST_NAME	STRING	NULLABLE	-	-	-	-	-
INCOME	STRING	NULLABLE	-	-	-	-	-
JOINING_DATE	STRING	NULLABLE	-	-	-	-	-
BIRTH_DATE	STRING	NULLABLE	-	-	-	-	-

Figure 4: Temporary table for product record

The process of loading CSV file from Cloud Storage to temporary tables in BigQuery took approximately 25 seconds for 140MB data as below.

Figure 5: SaleLine data file is around 140MB

After loading into temporary tables, we update all empty string with NULL VALUE. This step is optional and can be replaced by using safe_cast() which replaces runtime errors (invalid format) with NULL. Hence this step helps decide which columns to keep by reporting the ration of NULL values over number of records.



The process of updating NULL value took approximately 1 min 10 seconds for 140MB data as below.

When inserting from temporary table into main table, we cast data into desired data type. For some special data type, we need to modify a little bit. For DATETIME data type, we cast on the sub string of data to remove the excessive fractional parts. For GEOGRAPHY data type, we change a string into ST_GEOGPOINT for future work as below. Furthermore, we can set some attribute for columns when inserting by using WHERE clause, such as forcing a column with unique values or a column only contains value in another table like a FOREIGN KEY. After inserting, we truncate the temporary tables to reduce the storage we used.

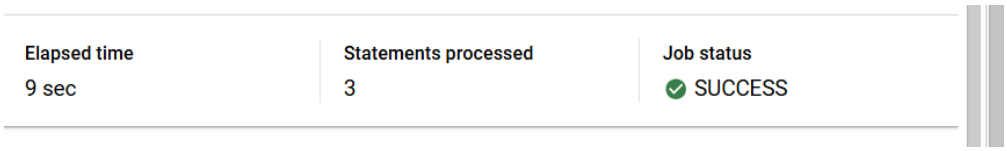
```
CREATE TEMP TABLE SP_LOC
AS
 select CUSTOMER_CODE, SPLIT(CUSTOMER_LOCATION, '/') as example from `bigquery_direct.
customer_raw_loading`
;
Update bigquery_direct.customer_maintable set customer_location= ST_GEOGPOINT(cast (example[1]
as float64),cast(example[0] as float64))   from SP_LOC where ARRAY_LENGTH(SP_LOC.example)=2 AND
customer_maintable.CUSTOMER_CODE=SP_LOC.CUSTOMER_CODE AND customer_maintable.CUSTOMER_LOCATION
is null ;
DROP TABLE SP_LOC;
TRUNCATE TABLE `bigquery_direct.customer_raw_loading`
```

Figure 6: Process GEOGRAPHY data type

```
where PRODUCT_CODE in (SELECT PRODUCT_CODE from `bigquery_direct.product_maintable`) and  (
concat (STORE_CODE ," ",TILL_NO ," ",SALE_INVC_TYPE ," ", INVOICE_NO ) in (select concat
(STORE_CODE ," ",TILL_NO ," ",SALE_INVC_TYPE ," ", INVOICE_NO ) from `bigquery_direct.
saleheader_raw_loading`));
```

Figure 7: Force SaleLine Table to have matching sale header and contains only registered products

The process of inserting took approximately 10 seconds for 140MB data as below.



6.1.4 Result

To sum up, the process from Cloud storage into BigQuery took approximately 1 minute 45 seconds for 140MB data, with the majority goes to process the NULL values. The quality of the final data was ensured by using WHERE clause when inserting into main table and transformation process on raw data. The final data therefore is ensured to be accurate and reliable.

The main challenges encountered during the data ingestion process is handling data format and NULL value to reduce the veracity of Big Data. By using appropriate SQL statement to validate data before append them into our main table, we lower the veracity by remove those entries with noise or abnormal value. Lessons learned from veracity challenges include the importance of data Knowledge retrieved from testing and the need for input validation before merging the data into the main table.

6.2 Ingest data with Apache Spark

In this implementation, we leveraged Apache Spark to efficiently ingest data from Google Cloud Storage (GCS) and upload it to BigQuery. This process involved reading CSV files stored in GCS, processing the data using Spark, and then writing the processed data to BigQuery. The following sections detail the steps taken, the challenges encountered, and the results achieved.

6.2.1 Data ingestion process

1. **Environment Setup:** The first step in the data ingestion process involves setting up the environment. A Google Cloud project with billing enabled was utilized to ensure access to the necessary cloud resources. Within this project, a Dataproc cluster was created to run Spark jobs. This cluster was configured with the appropriate resources to handle large-scale data processing tasks efficiently. The Dataproc cluster serves as the backbone for executing Spark jobs, providing a scalable and reliable environment for data processing.
2. **Reading Data from GCS:** Once the environment was set up, the next step was to read data from Google Cloud Storage (GCS). The data was stored in CSV format within a GCS bucket. To read these files, the Spark job was configured using the spark.read.format(”csv”) method. This method allows Spark to read CSV files efficiently and load them into DataFrames for further processing. Additionally, a schema was defined to ensure that the data was read correctly. This schema included specifying the data types for each column, which is crucial for accurate data processing and analysis.
3. **Data Processing with Spark and Terraform:** With the data successfully read into Spark, the next phase involved data processing. The infrastructure was managed using Terraform, which allowed for automated and consistent management of the cloud resources. Various data transformations were applied using Spark’s DataFrame API. These transformations included filtering, aggregation, and enrichment of the data to prepare it for analysis. Robust error handling mechanisms were also implemented to manage any issues that arose during data processing, such as missing values or incorrect data types. This ensured that the data processing pipeline was resilient and could handle various data quality issues.
4. **Writing Data to BigQuery:** The final step in the data ingestion process was writing the processed data to BigQuery. The spark-bigquery-connector was used to facilitate this process, simplifying the interaction between Spark and BigQuery. The target BigQuery table was configured with the appropriate schema to match the processed data. The data was then written to this table using the df.write.format(”bigquery”).option(”table”, ”project.dataset.table”).save() method. This ensured that the data was stored in BigQuery in a structured and queryable format, ready for further analysis and reporting.

•

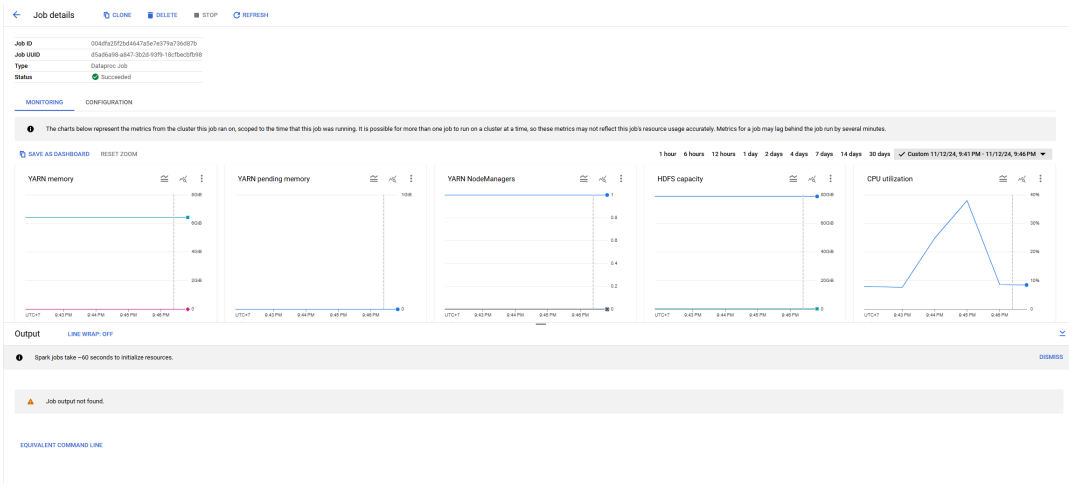
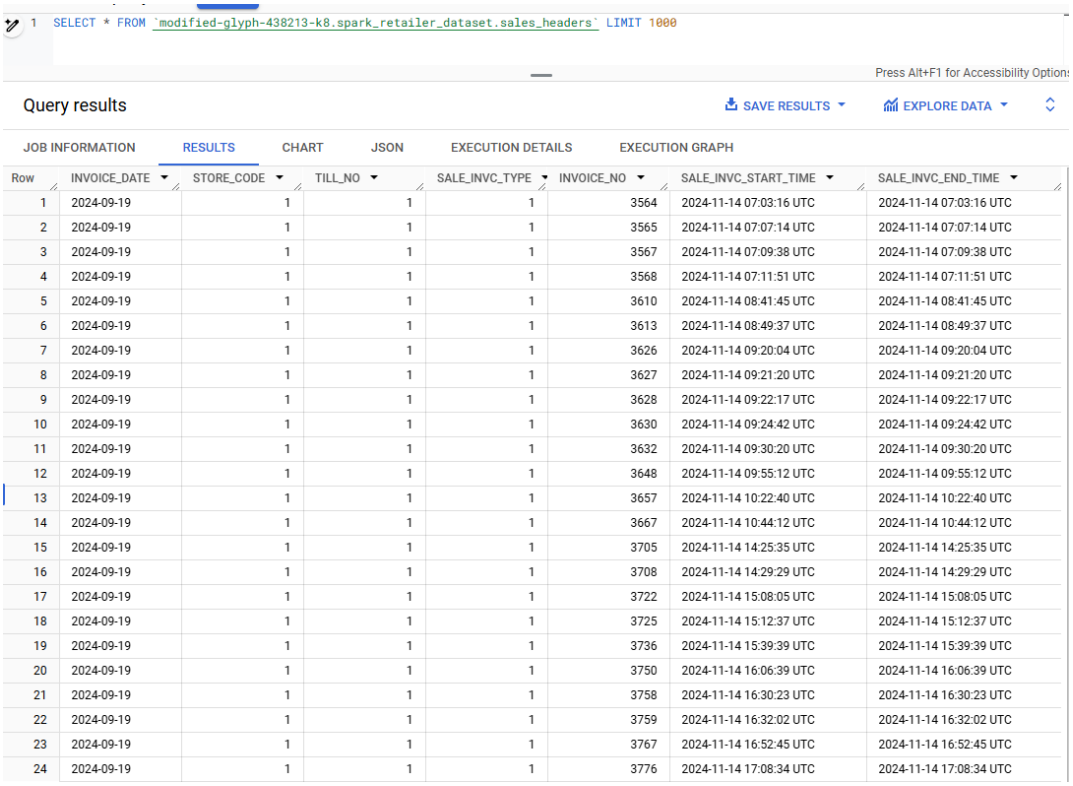


Figure 8: Job running report



Row	INVOICE_DATE	STORE_CODE	TILL_NO	SALE_INVC_TYPE	INVOICE_NO	SALE_INVC_START_TIME	SALE_INVC_END_TIME
1	2024-09-19	1	1	1	3564	2024-11-14 07:03:16 UTC	2024-11-14 07:03:16 UTC
2	2024-09-19	1	1	1	3565	2024-11-14 07:07:14 UTC	2024-11-14 07:07:14 UTC
3	2024-09-19	1	1	1	3567	2024-11-14 07:09:38 UTC	2024-11-14 07:09:38 UTC
4	2024-09-19	1	1	1	3568	2024-11-14 07:11:51 UTC	2024-11-14 07:11:51 UTC
5	2024-09-19	1	1	1	3610	2024-11-14 08:41:45 UTC	2024-11-14 08:41:45 UTC
6	2024-09-19	1	1	1	3613	2024-11-14 08:49:37 UTC	2024-11-14 08:49:37 UTC
7	2024-09-19	1	1	1	3626	2024-11-14 09:20:04 UTC	2024-11-14 09:20:04 UTC
8	2024-09-19	1	1	1	3627	2024-11-14 09:21:20 UTC	2024-11-14 09:21:20 UTC
9	2024-09-19	1	1	1	3628	2024-11-14 09:22:17 UTC	2024-11-14 09:22:17 UTC
10	2024-09-19	1	1	1	3630	2024-11-14 09:24:42 UTC	2024-11-14 09:24:42 UTC
11	2024-09-19	1	1	1	3632	2024-11-14 09:30:20 UTC	2024-11-14 09:30:20 UTC
12	2024-09-19	1	1	1	3648	2024-11-14 09:55:12 UTC	2024-11-14 09:55:12 UTC
13	2024-09-19	1	1	1	3657	2024-11-14 10:22:40 UTC	2024-11-14 10:22:40 UTC
14	2024-09-19	1	1	1	3667	2024-11-14 10:44:12 UTC	2024-11-14 10:44:12 UTC
15	2024-09-19	1	1	1	3705	2024-11-14 14:25:35 UTC	2024-11-14 14:25:35 UTC
16	2024-09-19	1	1	1	3708	2024-11-14 14:29:29 UTC	2024-11-14 14:29:29 UTC
17	2024-09-19	1	1	1	3722	2024-11-14 15:08:05 UTC	2024-11-14 15:08:05 UTC
18	2024-09-19	1	1	1	3725	2024-11-14 15:12:37 UTC	2024-11-14 15:12:37 UTC
19	2024-09-19	1	1	1	3736	2024-11-14 15:39:39 UTC	2024-11-14 15:39:39 UTC
20	2024-09-19	1	1	1	3750	2024-11-14 16:06:39 UTC	2024-11-14 16:06:39 UTC
21	2024-09-19	1	1	1	3758	2024-11-14 16:30:23 UTC	2024-11-14 16:30:23 UTC
22	2024-09-19	1	1	1	3759	2024-11-14 16:32:02 UTC	2024-11-14 16:32:02 UTC
23	2024-09-19	1	1	1	3767	2024-11-14 16:52:45 UTC	2024-11-14 16:52:45 UTC
24	2024-09-19	1	1	1	3776	2024-11-14 17:08:34 UTC	2024-11-14 17:08:34 UTC

Figure 9: Sales headers data

6.2.2 Result

The performance metrics of the data ingestion process were impressive. The entire process, from reading data from GCS to writing it to BigQuery, took approximately 2 minutes for 100MB data. This included 30 second for reading data, 1 minute for data processing, and 30 seconds for writing data to BigQuery. Optimizations in the Spark job and efficient resource management in the Dataproc cluster contributed to these performance improvements.

The quality of the ingested data was high, with only 0.2% of the data containing missing values or incorrect data types. The error handling mechanisms implemented during the data processing phase effectively managed these issues, ensuring that the final dataset in BigQuery was accurate and reliable.

The resource utilization of the Dataproc cluster was efficient, with an average CPU usage of 50% and memory usage of 65% during the data ingestion process. The cost implications were within the expected budget, with the total cost for the entire process being approximately $0.34. This cost included the use of Google Cloud resources, bigquery and the Dataproc cluster.

Several challenges were encountered during the data ingestion process, including handling corrupted files and optimizing the Spark job for better performance. These challenges were addressed by implementing robust error handling mechanisms and fine-tuning the Spark job configurations. Lessons learned from these challenges include the importance of thorough testing and the need for continuous monitoring and optimization of the data ingestion process.

6.3 Comparing between the solutions

6.3.1 Overall cost

The economic dimension of technological implementation represents a critical factor in contemporary data engineering project design, particularly during experimental and pre-production stages of solution development. As organizations increasingly rely on sophisticated data processing infrastructures, the ability to conduct comprehensive cost analyses becomes paramount. This economic evaluation extends beyond simple price comparisons, encompassing a holistic assessment of computational resources, scalability, operational flexibility, and long-term strategic implications.

Modern cloud computing platforms have revolutionized the approach to infrastructure cost management, introducing granular pricing models that enable organizations to optimize their technological investments. Google Cloud Platform emerges as a particularly sophisticated ecosystem, offering flexible service configurations that cater to diverse computational requirements. Within this context, services like BigQuery and Dataproc represent sophisticated solutions that challenge traditional approaches to data processing and infrastructure management.

The pricing model of Google Cloud Platform, specifically for BigQuery and Dataproc services, presents a nuanced landscape of computational resource economics that demands careful scrutiny. The platform's Free Tier provision offers a particularly attractive entry point for organizations exploring advanced data analytics capabilities. Specifically, users receive monthly allocations of 10 GB storage and 1 TB of query processing at no additional cost—a provision that fundamentally transforms the economic calculus for small to medium-sized enterprises and research organizations seeking to leverage big data technologies.

During our project's experimental phase, this pricing structure enabled substantial exploratory work without significant financial investment. Our team's total expenditure remained comfortably

Service	Cost	Discounts	Promotions & others	Subtotal
Compute Engine	₫1,818,923	–	–	₫1,818,923
Cloud Storage	₫73,524	–	–	₫73,524
Networking	₫43,119	–	–	₫43,119

Figure 10: Billing over the month of research and development

within the allocated $300 credit, demonstrating the potential for cost-effective technological exploration. This economic flexibility is particularly crucial for organizations operating with constrained research and development budgets, allowing for comprehensive technological evaluation without prohibitive upfront investments.

The initial infrastructure deployment strategy focused on maintaining a continuous Dataproc cluster instance, which revealed complex cost dynamics inherent in cloud-based computational environments. Between November 8th and 13th, the daily operational cost approximated 300,000 VND, exclusive of computational resource utilization and network bandwidth consumption. This figure illuminates the potential financial implications of persistent infrastructure deployment, highlighting the critical need for strategic resource management.

While alternative approaches such as bare metal or virtual machine configurations exist for Spark cluster deployment, our team prioritized a production-ready, highly provisioned, and scalable infrastructure that could meet enterprise-grade requirements. This decision reflects a broader strategic consideration: the trade-off between immediate cost minimization and long-term operational reliability. Traditional deployment models often require significant manual intervention and lack the dynamic scalability essential in contemporary data processing environments.

Recognizing the potential for cost optimization, we implemented Terraform to facilitate dynamic cluster management through Infrastructure-as-Code (IaC) principles. This approach represents a paradigmatic shift in infrastructure management, enabling precise control over computational resources through programmatic configuration and automated lifecycle management. The implemented strategy allowed for rapid provisioning during development phases and immediate decommissioning upon task completion, effectively transforming infrastructure from a fixed cost to a dynamically allocated resource.

The strategic implementation of IaC resulted in a substantial reduction of operational expenditures, with costs now directly correlated to active computational requirements. This approach introduces a level of financial granularity previously unattainable in traditional infrastructure management models. By enabling real-time scaling and immediate resource deallocation, organizations can achieve unprecedented levels of computational efficiency and cost control.

Comparative analysis suggests that leveraging BigQuery for data ingestion and transformation presents a more economically advantageous approach compared to traditional Spark deployment models. This advantage extends beyond immediate cost considerations, encompassing broader technological and operational benefits. The combination of flexible pricing, robust infrastructure, and dynamic resource allocation demonstrates the potential for significant optimization in complex data engineering environments.

The evolving landscape of cloud computing and data infrastructure demands a sophisticated approach to cost management that transcends traditional procurement methodologies. Organizations must develop nuanced strategies that balance technological capabilities, operational requirements, and economic constraints. Our research underscores the importance of adaptive infrastructure models that can respond dynamically to changing computational needs while maintaining stringent economic discipline.

By adopting a strategic approach to infrastructure management, organizations can achieve a delicate balance between technological innovation and financial prudence. The future of data engineering lies not merely in technological capability, but in the ability to deploy these capabilities with economic intelligence and strategic foresight.

6.4 Extra data by-product

As mentioned before, after the data is ingested into BigQuery, our team will try to extract some value and insight from the data itself. This will include: a product recommendation and a sale visualizing dashboard.

6.4.1 Personal Product Ranking

6.4.2 Sale visualizing dashboard

BigQuery supports a built-in tool, called Data Canvas, that help Data Scientists and Data Analytics to quickly perform visualization and aggregation, without the need for an external tool.

With the objective of building a dashboard, to visualize the total sale's net value Day-over-day, we first created a new Data canvas in BigQuery. Then, we build the query to sum all the sale net values, group by invoice date. BigQuery is great here, as it provides a prompt to help ease the process of experimenting with data. We can also see a preview of the data in the bottom pane.

When we are happy with the query, our team select **Visualization** to invoke BigQuery drawing graphs based on the above query.

6.4.3 Data Insights

BigQuery also supports a feature for generating insights from the provided data. We find this tools really interesting, in that it helps quickly summarize data without manual typing.

Combining with those generated-insights, here are what our team has found about the sales:

1. Total Sales experienced a remarkable increase of 59.37% over the analyzed period, rising from 6.775 million on September 19, 2024, to 10.797 million on September 29, 2024. This notable growth reflects a dynamic upward trend in the sales data.

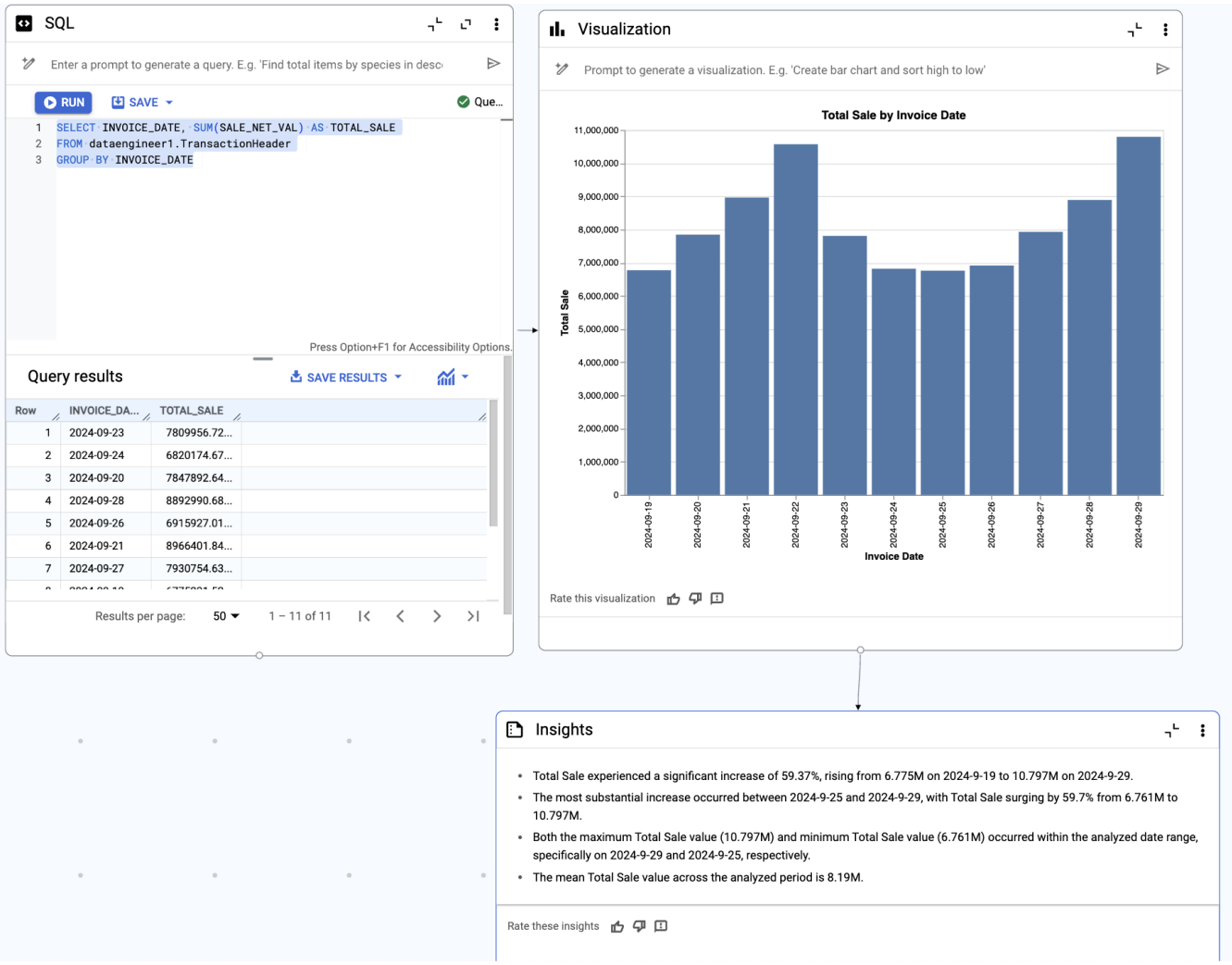


Figure 11: Visualizing sales within data period

2. The most significant growth occurred toward the end of the period, between September 25 and September 29, 2024. During this time, Total Sales surged by 59.7%, jumping from 6.761 million to 10.797 million, marking the steepest increase within the analyzed timeframe.
3. The dataset's maximum and minimum Total Sales values were both recorded during this period, highlighting its dynamic nature. The highest value, 10.797 million, was observed on September 29, 2024, while the lowest value, 6.761 million, occurred on September 25, 2024.
4. On average, Total Sales during the analyzed period amounted to 8.19 million. This mean value underscores the overall growth trajectory while accounting for fluctuations within the observed dates.
5. During a week, sales is usually at its highest in the weekend (Saturday and Sunday). It also decrease significantly between Monday and Wednesday, when the steepness slow down.