Data Engineer – Data Fellowship 12 IYKRA

Assignment 2

Alya Mutiara Firdausyi

Instruction

Task

Hey, data rangers! Our analytics team needs data related to our Supermarket sales for our **monthly reporting project**. Data is in CSV format which can be accessed here, Supermarket sales. The business team also needs you to **generate any insights** we can get from those data. You'll collaborate with a data visualization expert who will create a dashboard for the business team. You need to **provide summary tables** for data visualization experts, such as monthly revenue, etc.

As per governance, you have to **ingest data to the data warehouse in a real-time manner**, then summary tables can be generated with either real-time or batch pipelines depending on the business needs.

Output

You'll need to provide screenshots as the output,

- Apache Kafka, topic list, and topic details.
- Apache NiFi, connectors, relationships, and its configurations.
- Informatica, mapping configs, map task configs, task flow configs, etc

---

Goal:

Build a real-time data pipeline to ingest supermarket sales data, generate insights, and populate summary tables for data visualization.

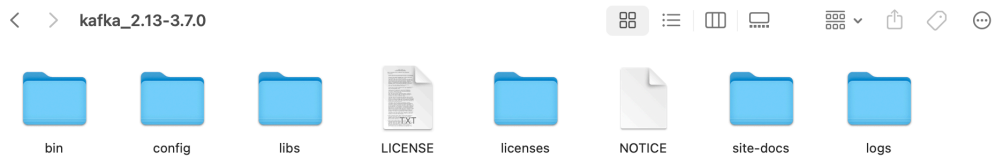
Tools:

- **Apache Kafka**: This will act as the central hub for ingesting the real-time sales data stream.
- **NiFi**: To create a data flow that reads data from a Kafka topic (a category for data streams), transforms the data (e.g., cleanses or filters it), and then routes it to the data warehouse.

- **Informatica**: This commercial platform offers a more comprehensive set of tools for data integration, including pre-built connectors for Kafka and data warehouses. We'd use Informatica's visual interface to build the data pipeline.
- **Data Warehouse**: This is the final destination for the processed data. It could be a cloud-based data warehouse like Google BigQuery.

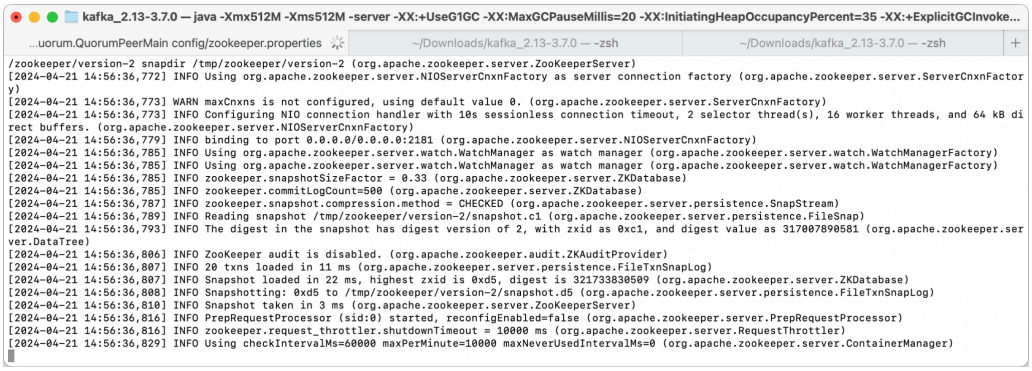
Step:

1. Data Ingestion: setting up Kafka topic to receive the real-time sales data stream by continuously reading the CSV file at regular intervals.
    a. Ensure that Kafka is already downloaded to the local computer. Then open a new blank terminal and locate the directory to the Kafka folder.



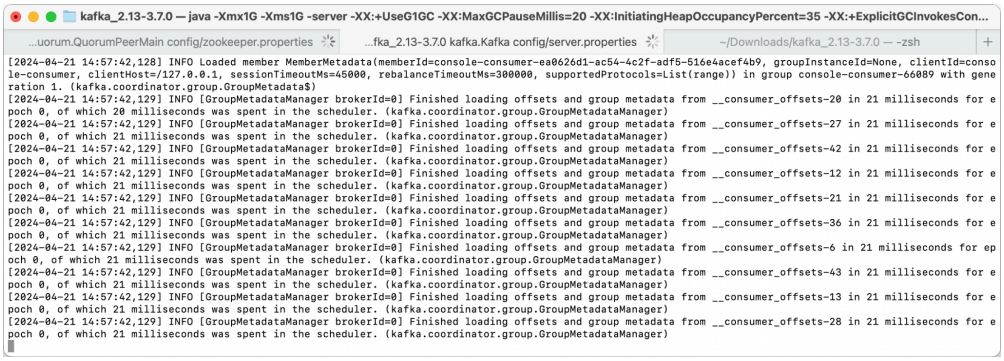
    b. Start the ZooKeeper Server in a new terminal.

```
bin/zookeeper-server-start.sh config/zookeeper.properties
```



    c. Start the Kafka Server in another new terminal.

```
bin/kafka-server-start.sh config/server.properties
```



d. In another new terminal window, create **supermarketSales** topics.

```
bin/kafka-topics.sh --bootstrap-server localhost:9092 --topic supermarketSales --create --partitions 3 --replication-factor 1
```

Then check whether the topic exists or not.

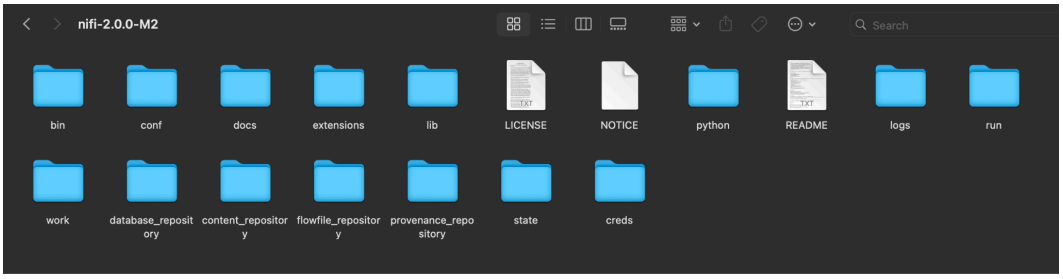
```
bin/kafka-topics.sh --bootstrap-server localhost:9092 --list
```

e. Describe the supermarketSales topic.

```
bin/kafka-topics.sh –bootstrap-server localhost:9092 --topic supermarketSales --describe
```

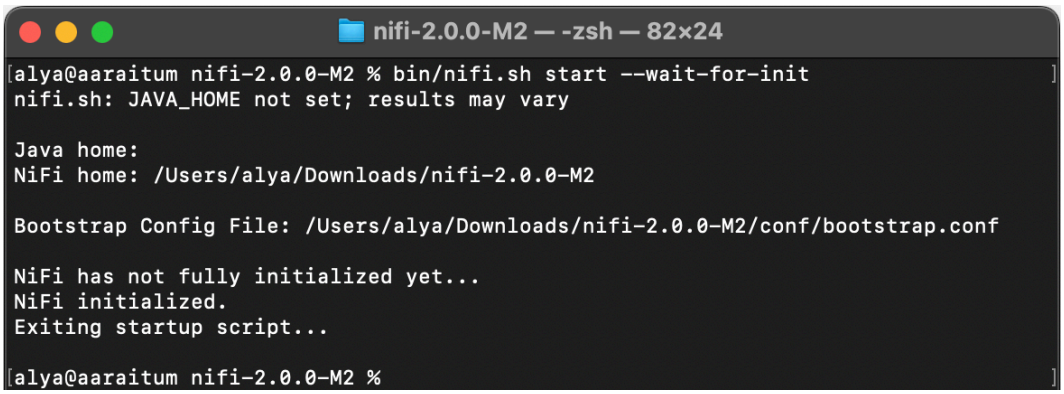
f. Download the Supermarket Sales data from the link provided and change the format into a JSON.

2. Data Processing: use NiFi and Informatica to build a data pipeline that reads data from the Kafka topic, then cleanses, transforms, and aggregates the data to generate summary tables for monthly revenue.
    a. Prepare the NiFi server by ensuring to download the binary from the official NiFi website, then open a new blank terminal and locate the NiFi directory.

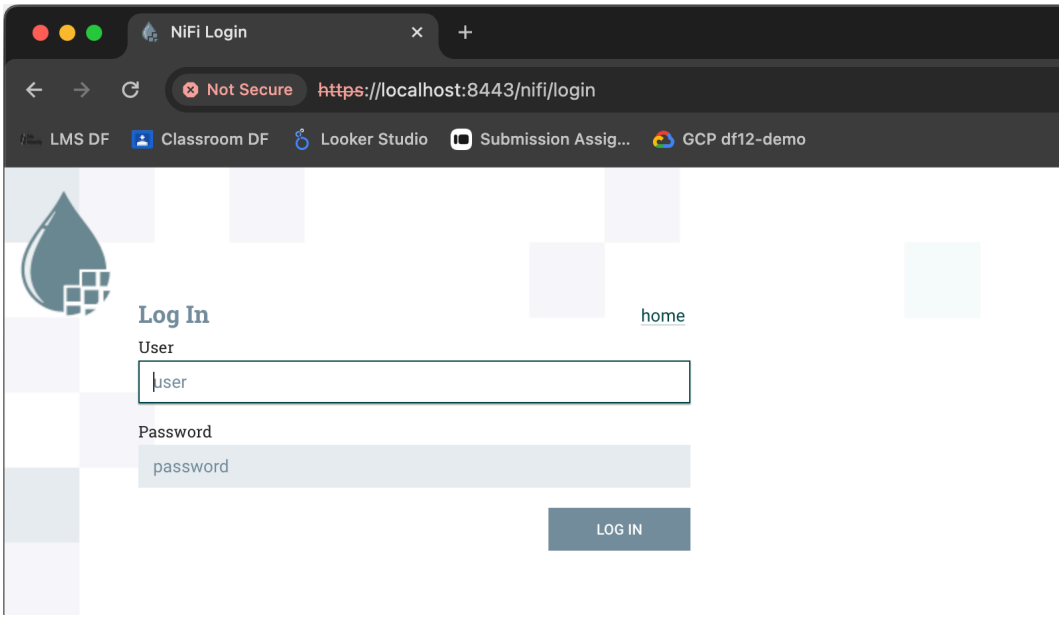


b. Run the NiFi.

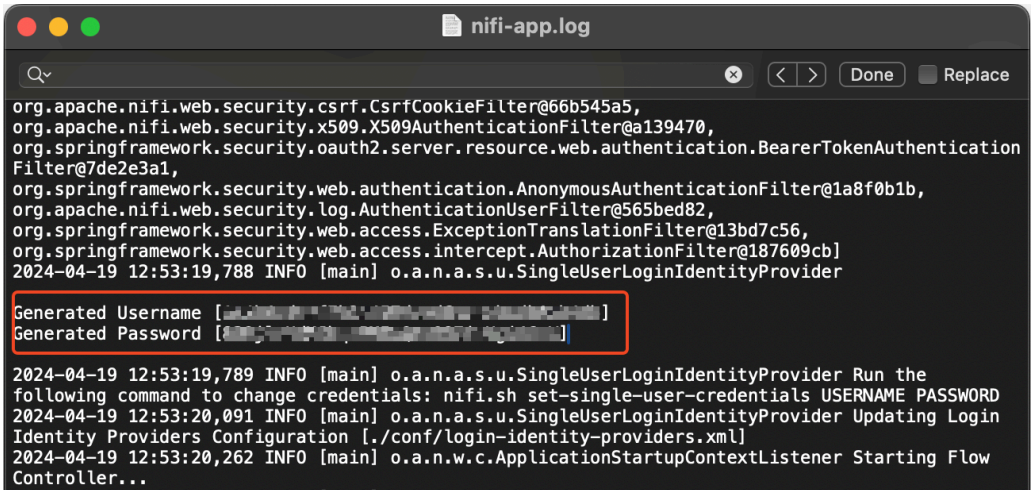
```
bin/nifi.sh start --wait-for-init
```



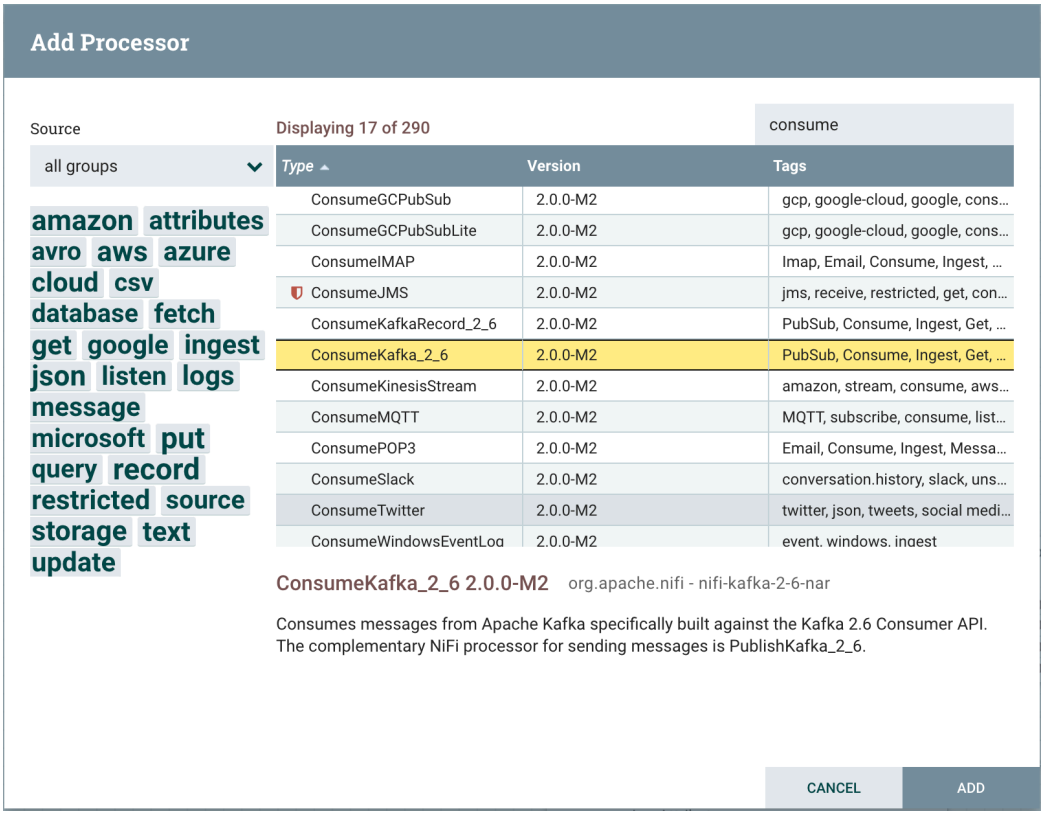
c. Open https://localhost:8443/nifi. Ensure to use HTTPS as NiFi is using a secure connection.



d. Login using the generated credential in the logs/nifi-app.log, then search for the “Generated Username” keyword.



e. After successfully logging in, create a new processor at the top section of the page by dragging and dropping the Processor logo into the canvas. Then choose the ConsumeKafka.



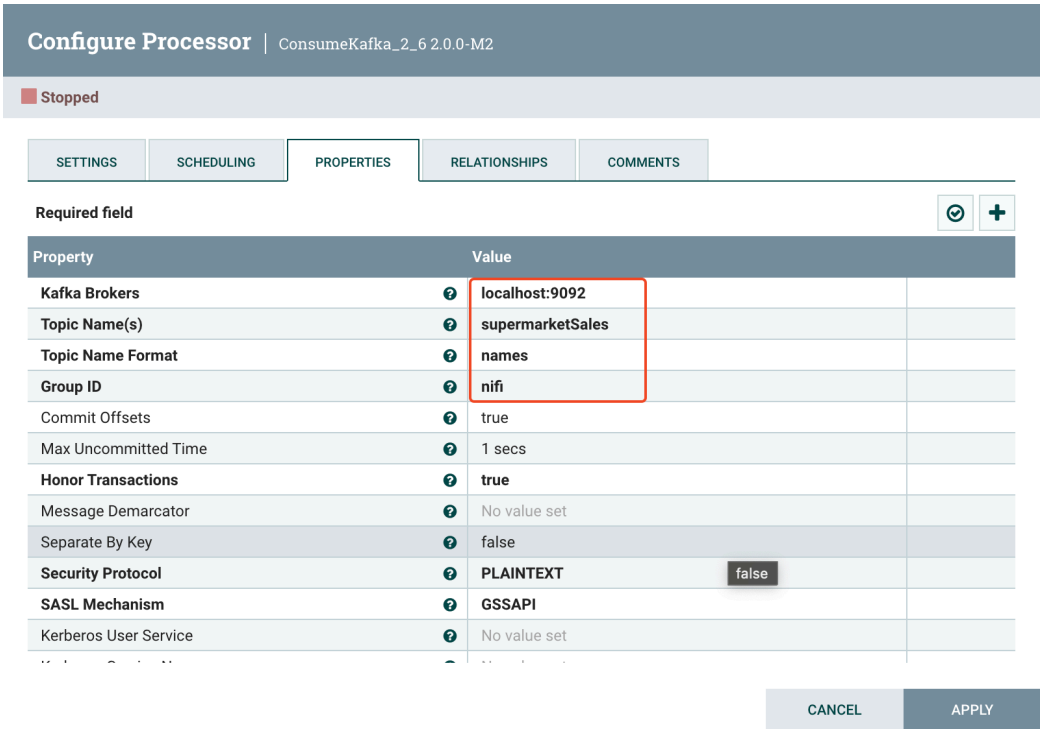
f. Double-click on the ConsumeKafka processor, then go to the Properties tab. Change the value as below:

Kafka Brokers: localhost:9092

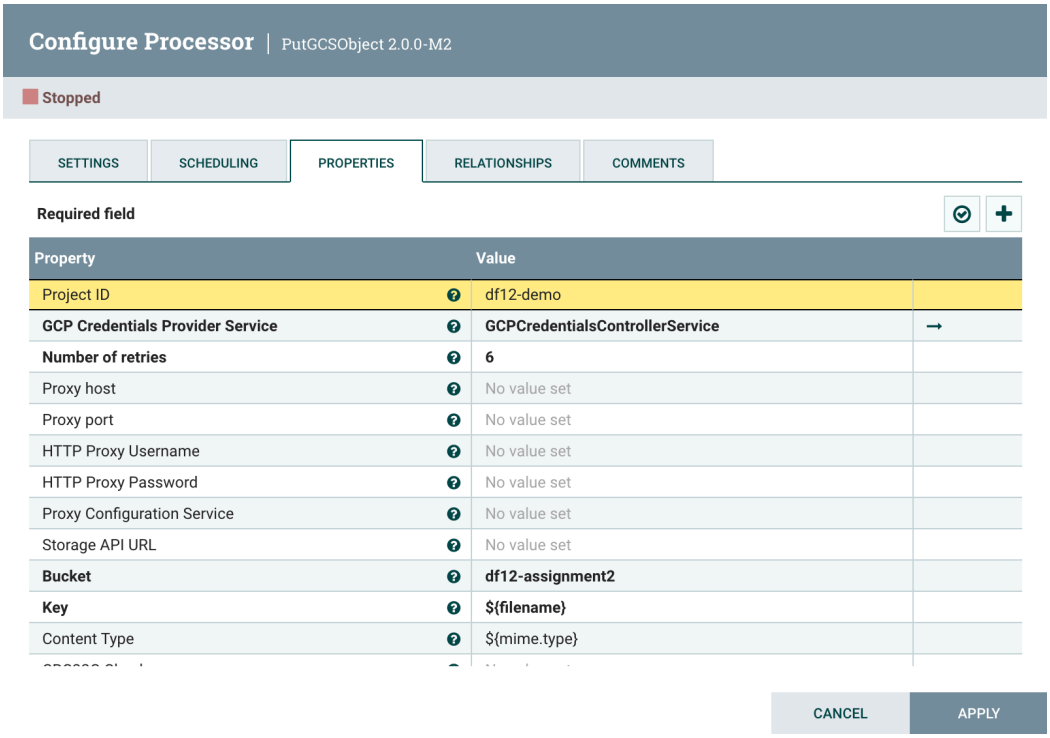
Topic Name(s) : supermarketSales

Topic Name Format: names

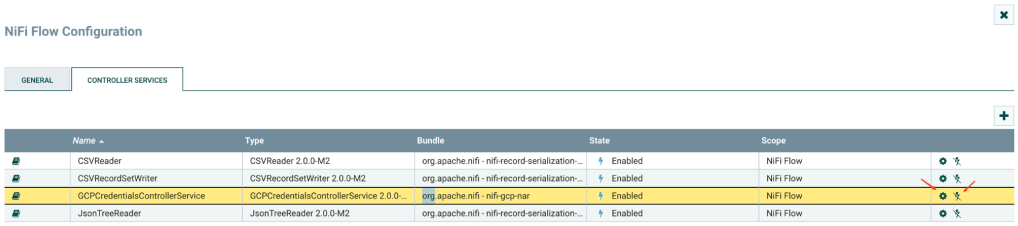
Group ID: nifi



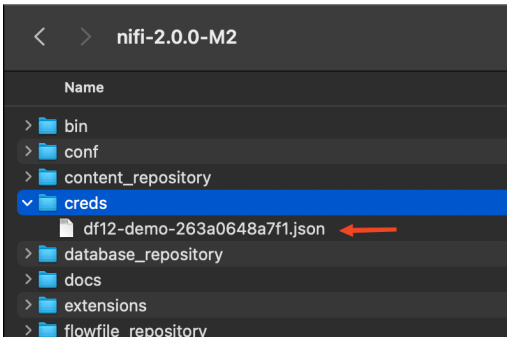
g. Add for another Processor, search for PutGCSObject which has a function to write data into a Cloud Storage bucket. Double-click the processor and go to the Properties tab. Change the value of Project ID to use the desired GCP Project ID, the Cloud Storage bucket that wants to be used in the Bucket property, and choose the GCPCredentialsControllerService in the GCP Credentials Provider Service property.

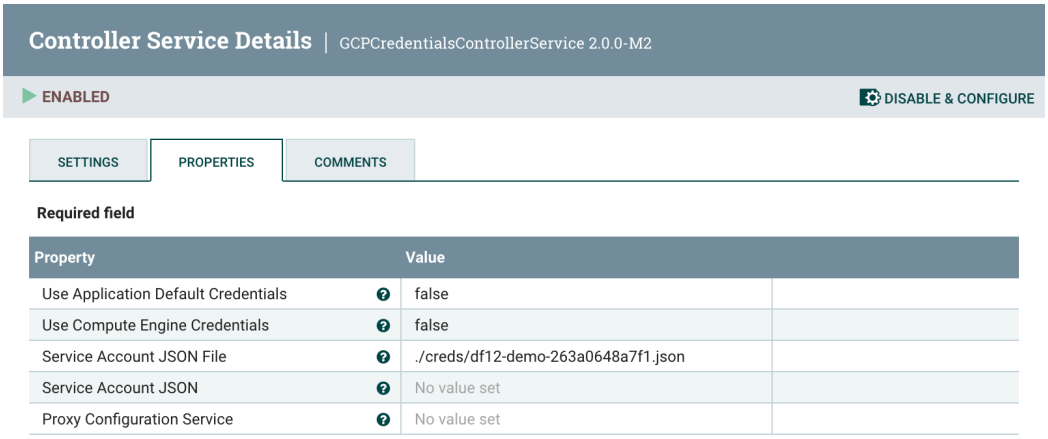


Configure the GCPCredentialsControllerService by clicking the arrow and going to the Controller Services. Then choose to configure the service and go to the Properties tab.

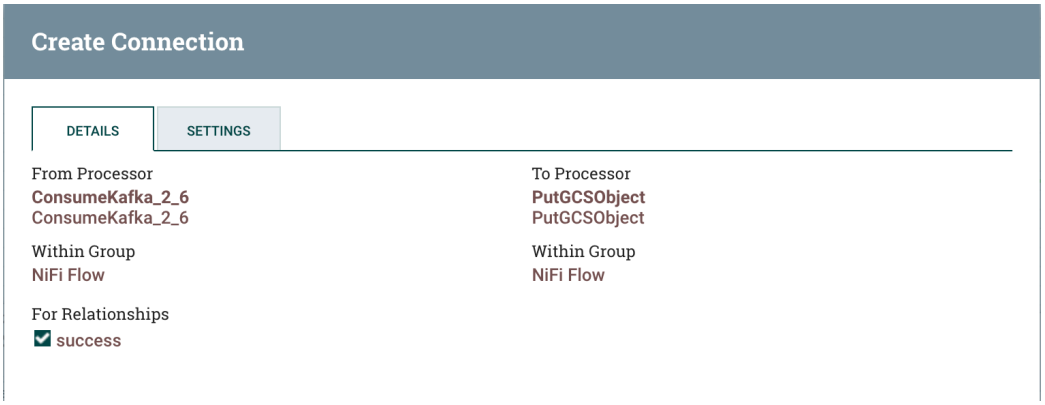


Fill the Service Account JSON File value with the GCP service account key in the JSON format inside the NiFi directory. Then click the lightning symbol to enable the service.

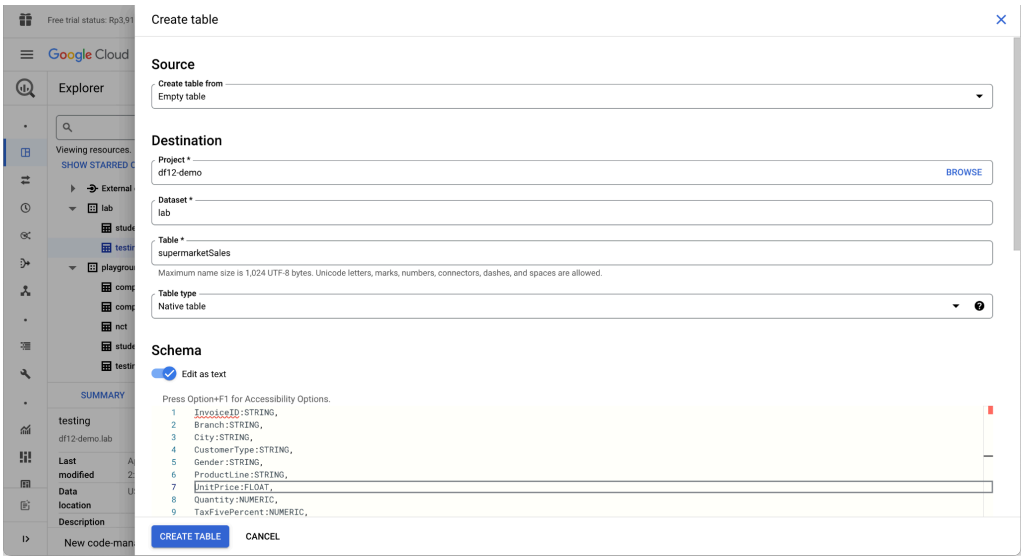




h. Create a Connection from ConsumeKafka to the PutGCSObject.



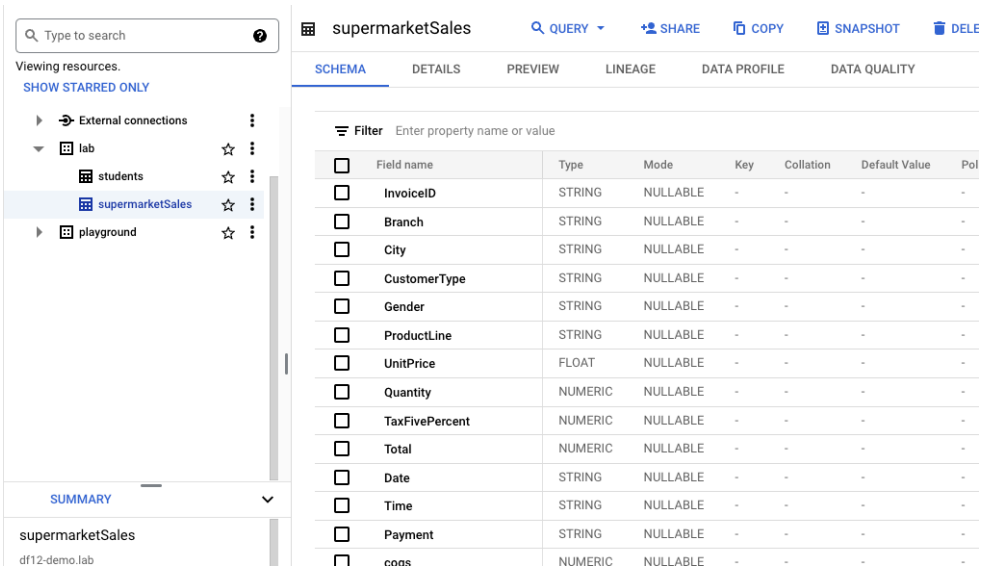
i. Create a new dataset and table in the BigQuery for storing the supermarket sales dataset.



On the Schema section, change to Edit as text and put this into the box.

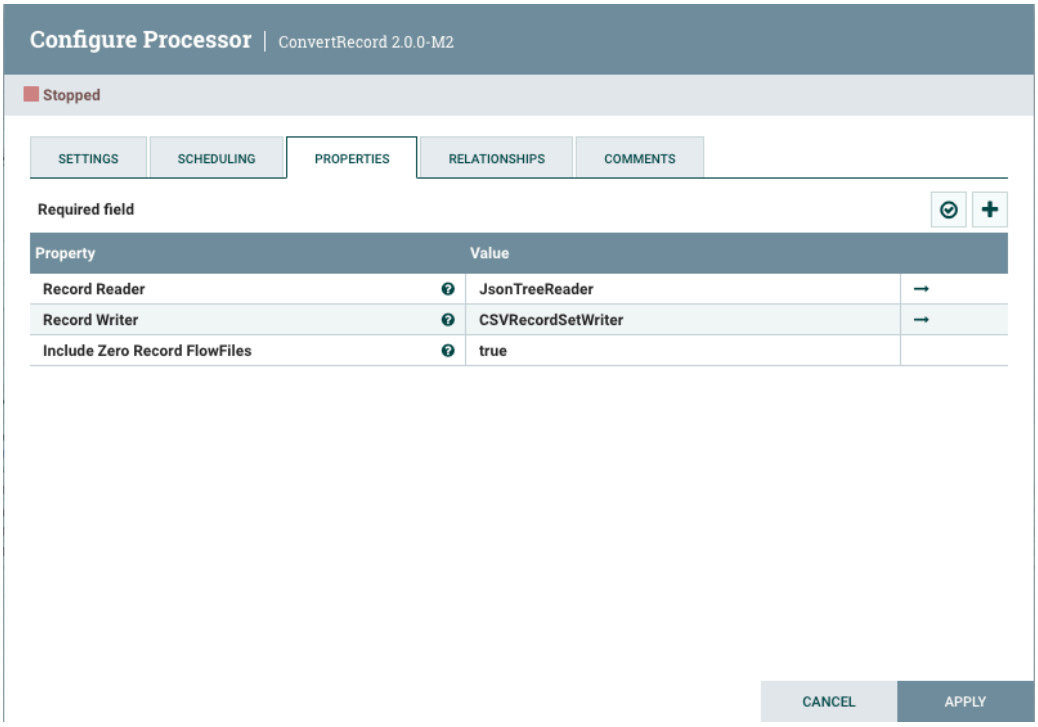
```
InvoiceID:STRING,
Branch:STRING,
City:STRING,
CustomerType:STRING,
Gender:STRING,
ProductLine:STRING,
UnitPrice:FLOAT,
Quantity:NUMERIC,
TaxFivePercent:NUMERIC,
Total:NUMERIC,
Date:STRING,
Time:STRING,
Payment:STRING,
cogs:NUMERIC,
GrossMarginPercentage:NUMERIC,
GrossIncome:NUMERIC,
Rating:NUMERIC
```

Ensure that the table is created.

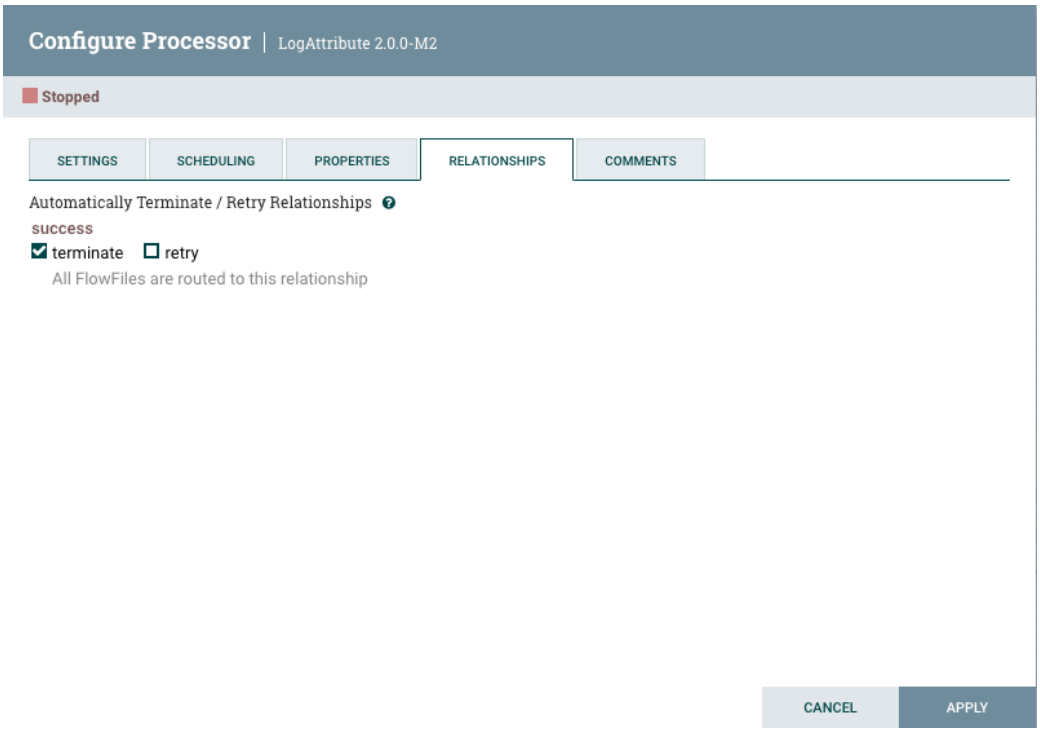


j. Back to NiFi, add a ConvertRecord processor, and create a connection from PutGCSObject for the relationship success.

   On the Properties tab, change the Record Reader value to use JsonTreeReader and the Record Writer value to CSVRecordSetWriter.

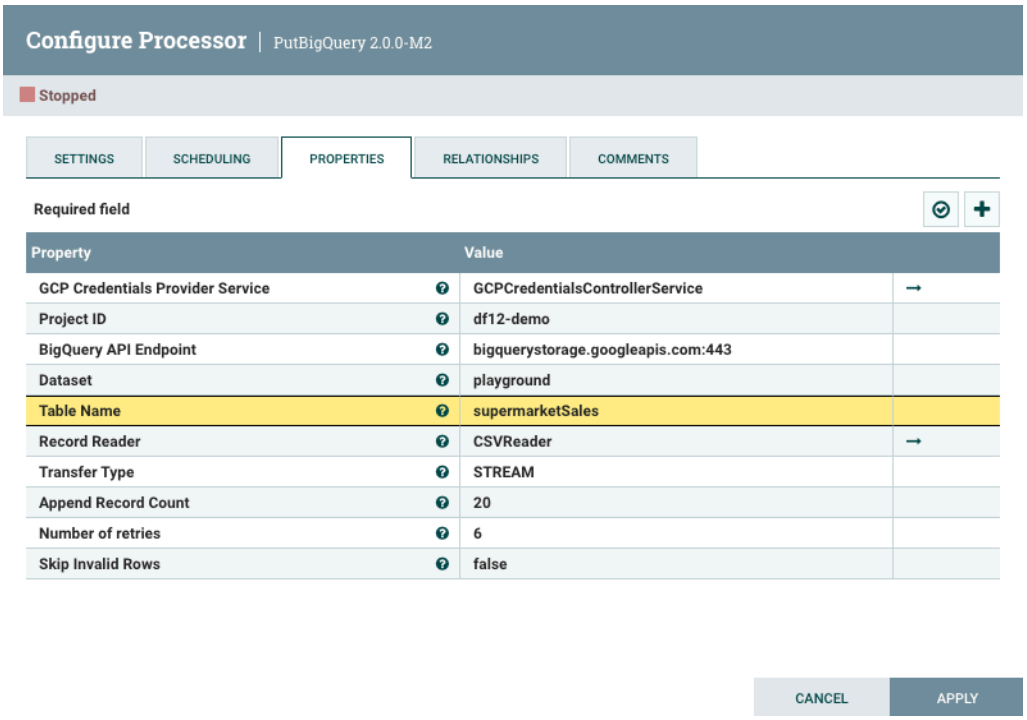


k. Add a new Processor and select LogAttribute. Create a Connection from PutGCSObject to create a relationship failure and from ConvertRecord to create a relationship failure also. On the Relationships tab, tick the terminate for relationship success.

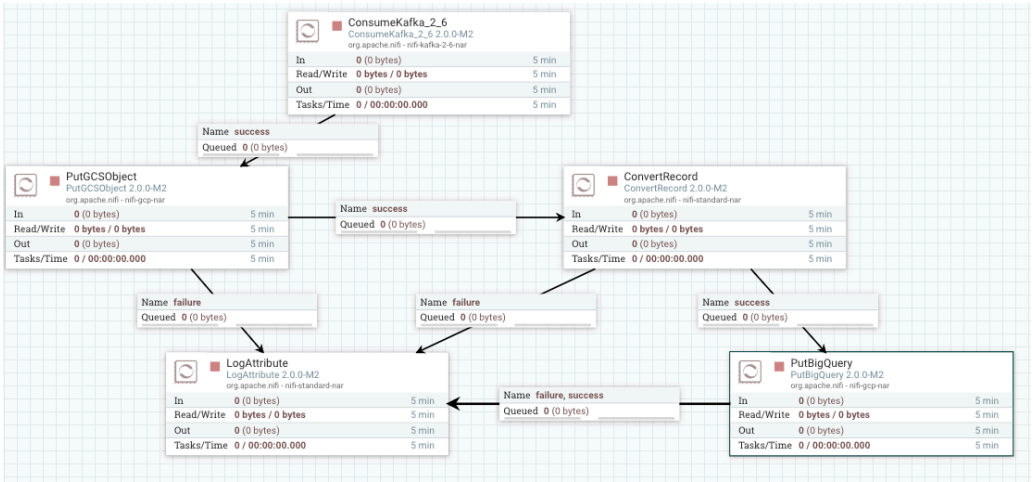


l. Add a new PutBigQuery processor, then create a connection from ConvertRecord for a relationship success. Then create a connection to LogAttribute for both relationship success and failure.

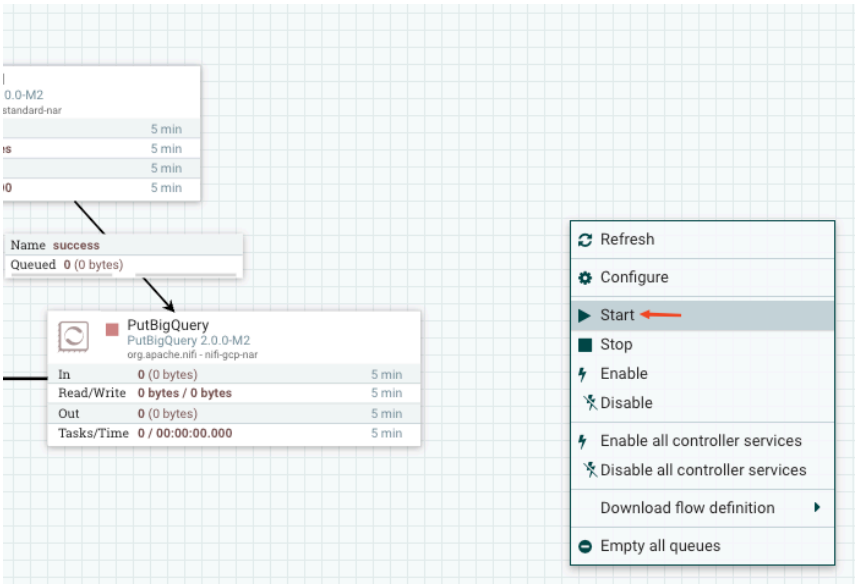
Then go to the Properties tab and change the properties of GCP Credentials Provider Service, Project ID, Dataset, Table Name, and the Record Reader.



m. The NiFi canvas would look like this.



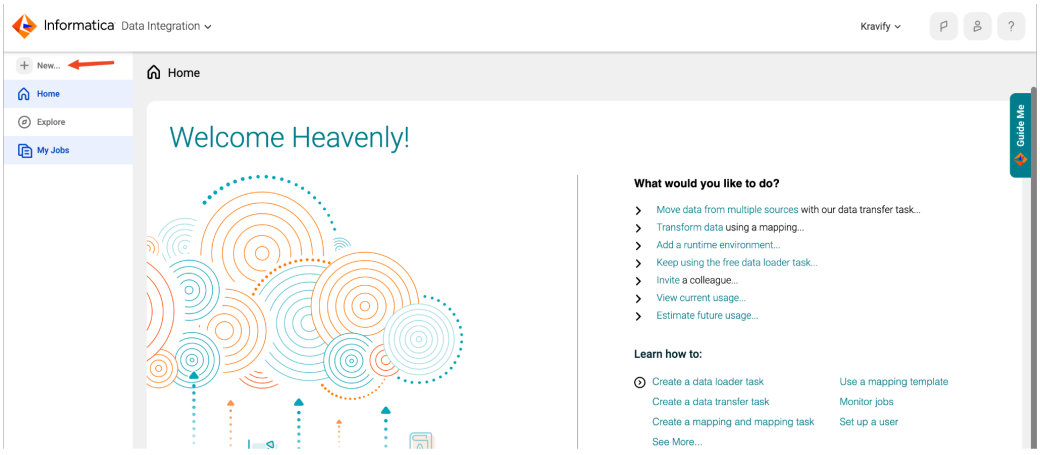
n. Start the NiFi by right-clicking on the canvas.



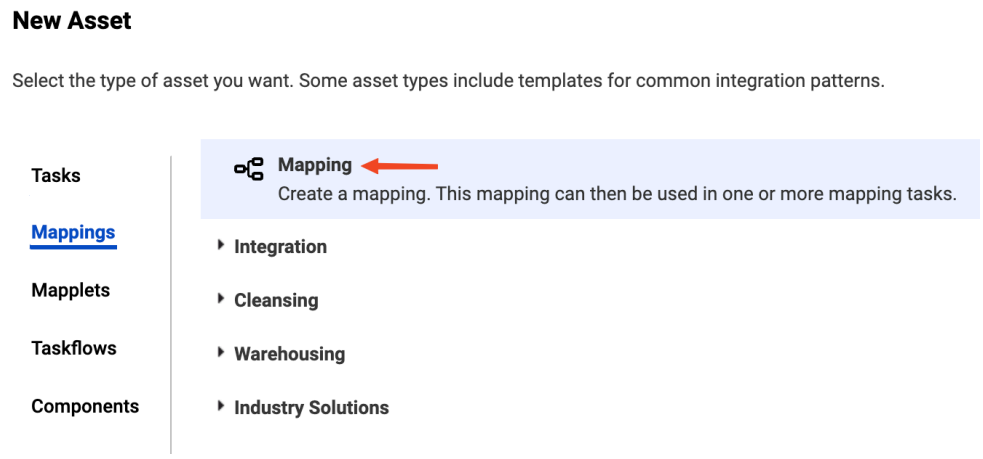
o. Open a new terminal and run this command to send the message to the Kafka Producer.

```
cat file/supermarketSales.json | bin/kafka-console-producer.sh --broker-list localhost:9092 --topic supermarketSales
```

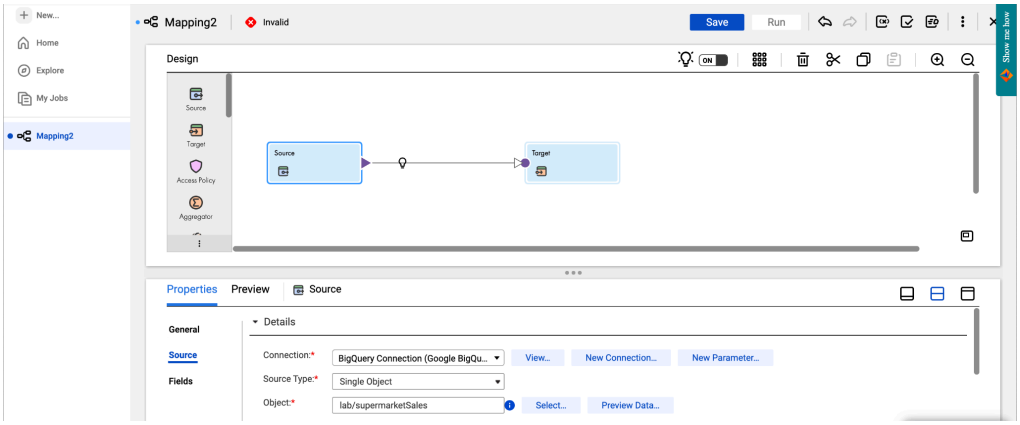
p. Log in to the Informatica Intelligent Cloud Services page. On the left sidebar, select New.

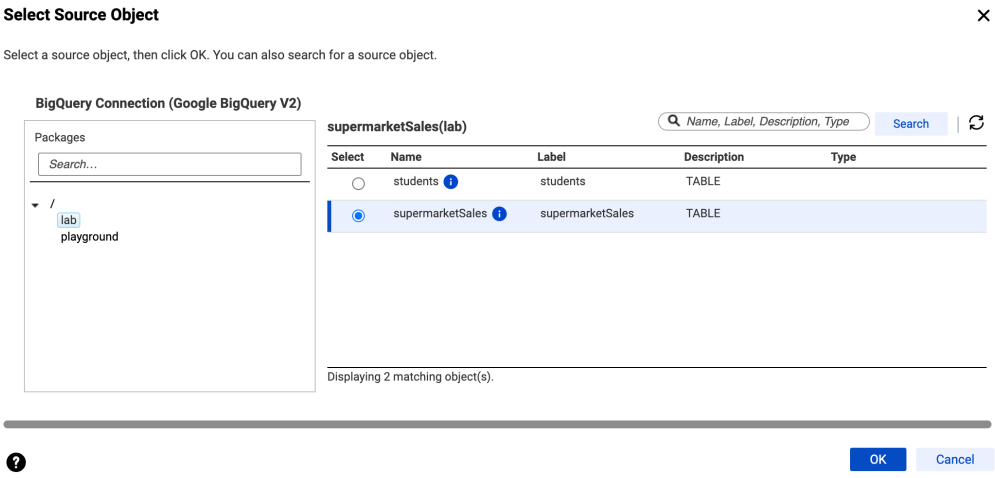


Select Mappings, then create a Mapping.

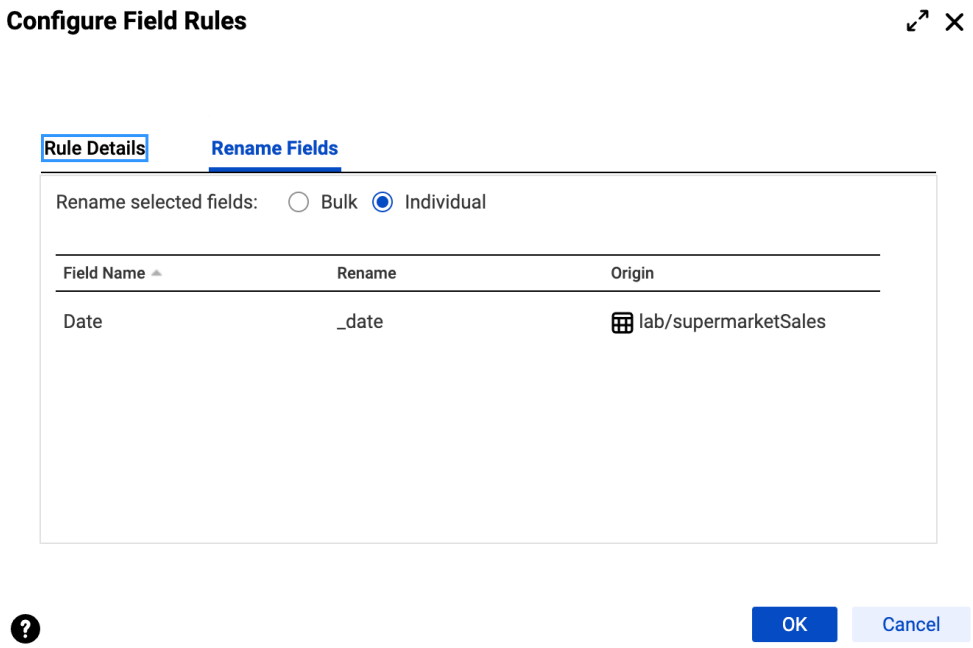


q. Select the source and configure the connection to use BigQuery.

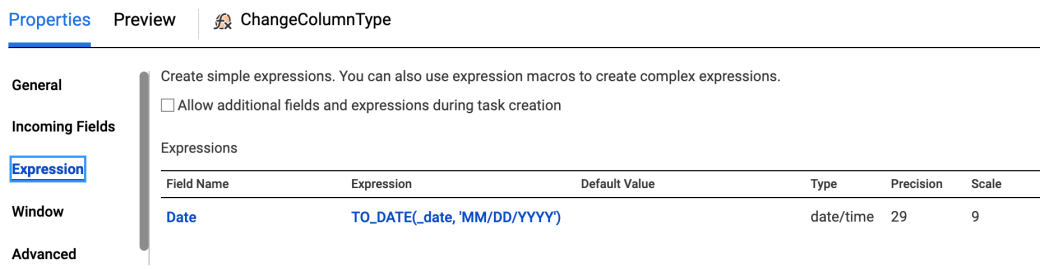




r. We need to change the date format from string to date format by adding the Aggregation. Then in the Incoming Field, add a Field rule to change the current date column into _date so we can add the correct type column with the date name into the table.



Then in the Expression tab, add an expression TO_DATE using _date column and save it as the Date field name.



s. Create two Aggregations to summarize data based on the date (monthly) and Product Line. For the monthly summary, we need to use the Group By Date column and for the Product Line use the Product Line column.

Edit Field

Create new output field, variable field, input macro field or output macro field.

Field Type: Output Field

Name:* total_sales

Type:* double

Precision:* 15

Scale: 0

Description: monthly sales from Total column

OK Cancel

Field Expression: total_sales(double, 15, 0)

Configure the expression by adding fields, functions, and variables.

Expression: Not Parameterized

The expression is valid.

Fields

Expression

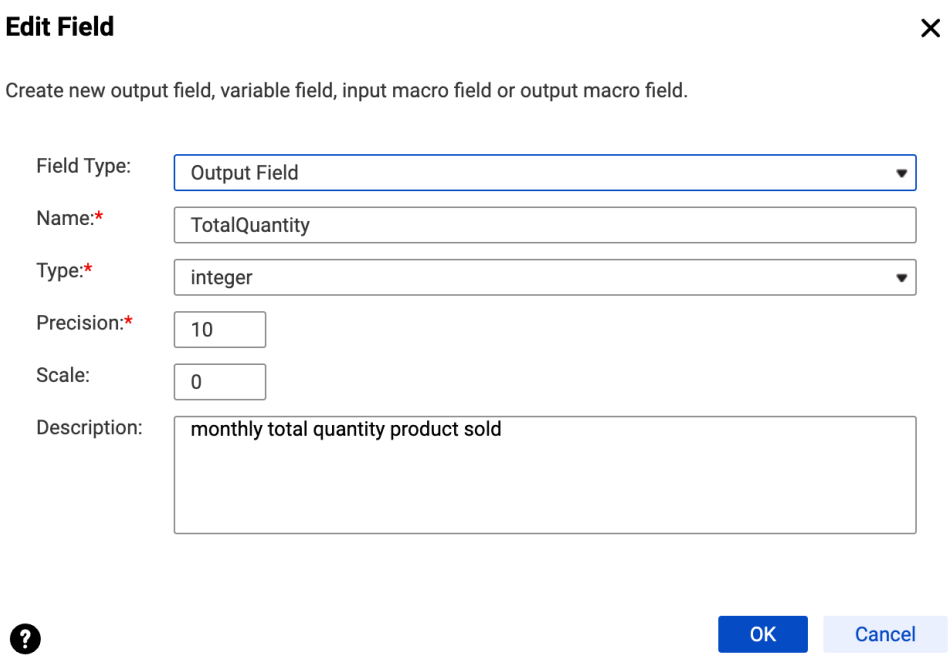
Validate

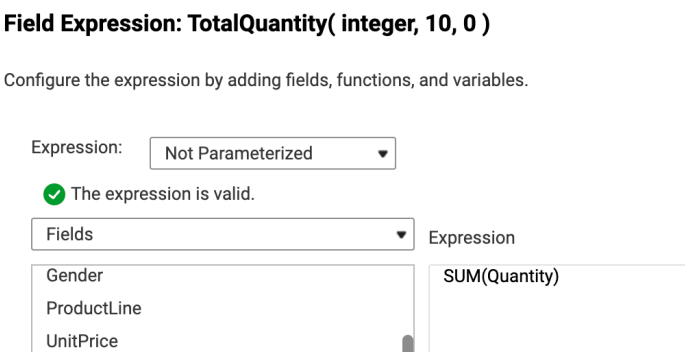
- Date
- InvoiceID
- Branch
- City
- CustomerType
- Gender
- ProductLine

SUM(Total)

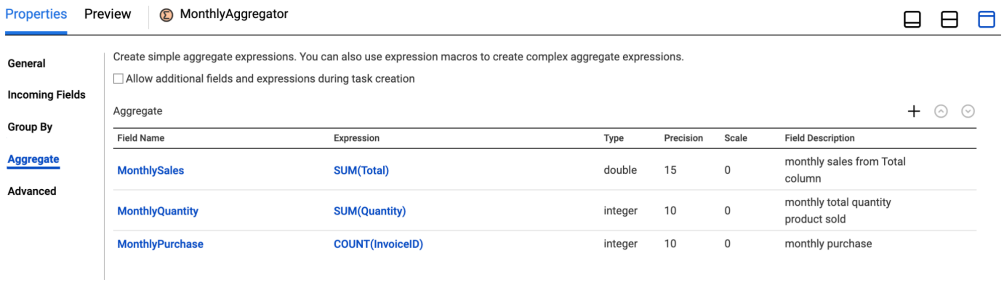
Select a field to see detailed information about it.

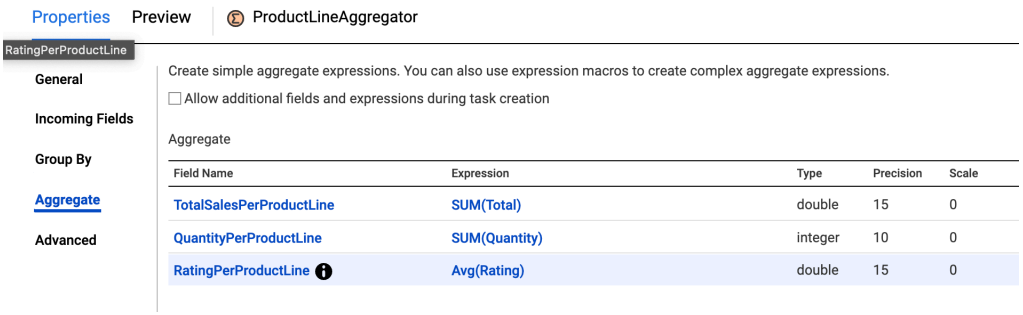
OK Cancel



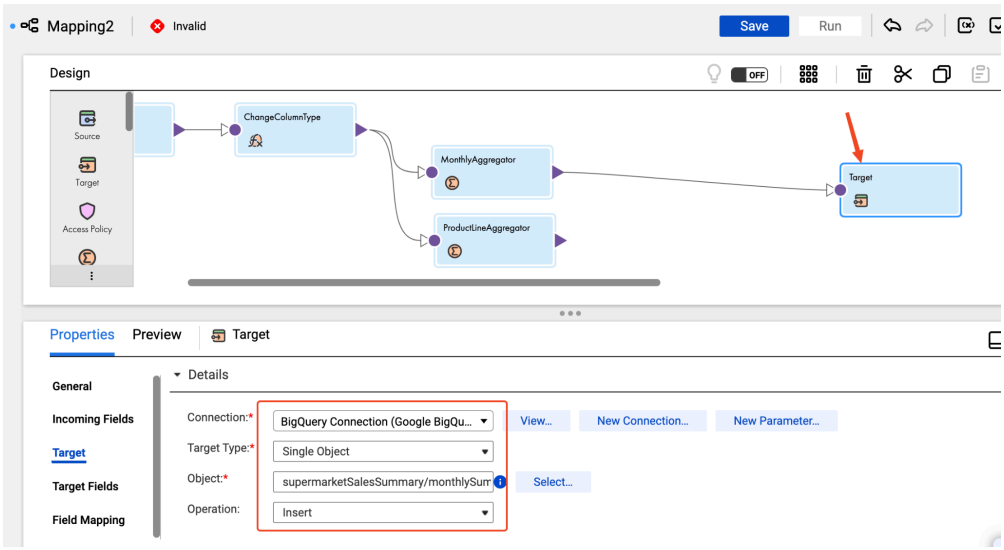


We can create seveaggregationstion based on the data.

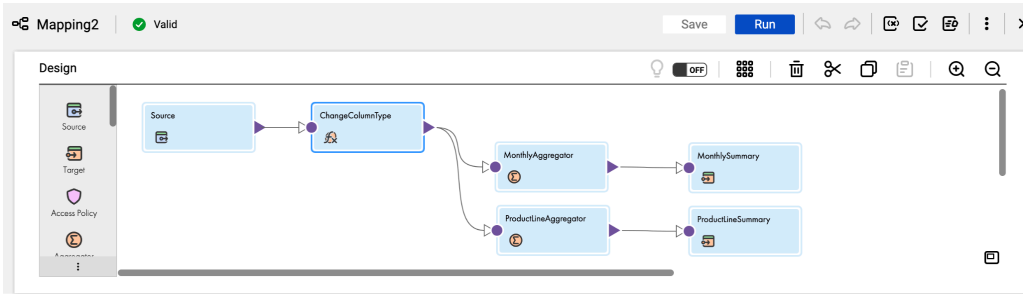


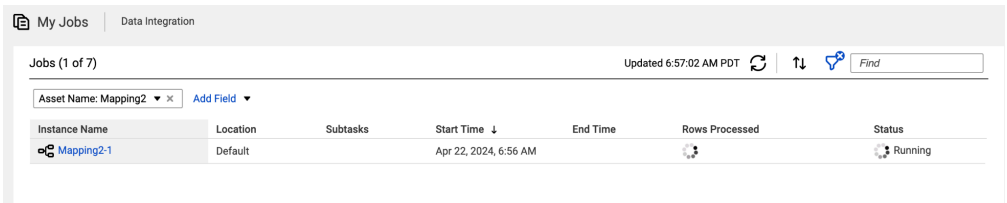


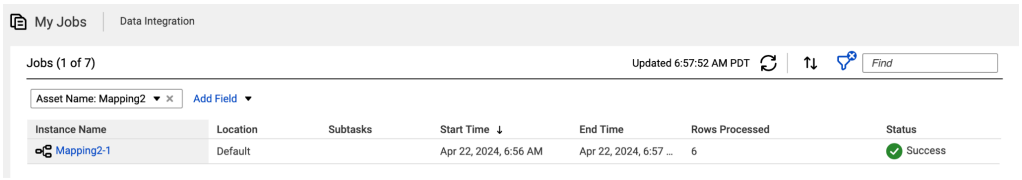
t. Set the Target in a new dataset in BigQuery.



u. Run the query and wait until it is successful.

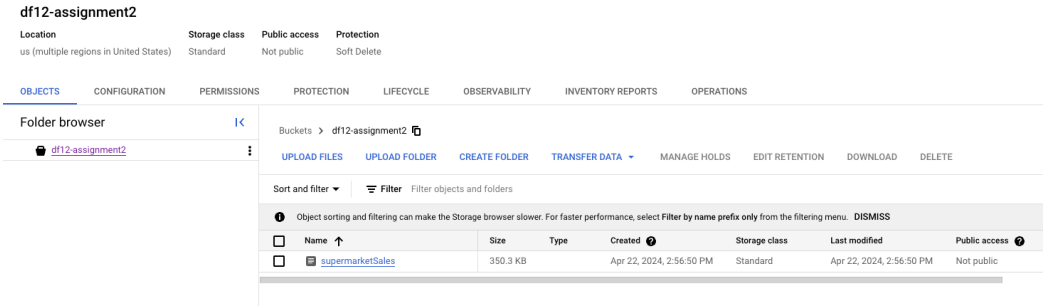




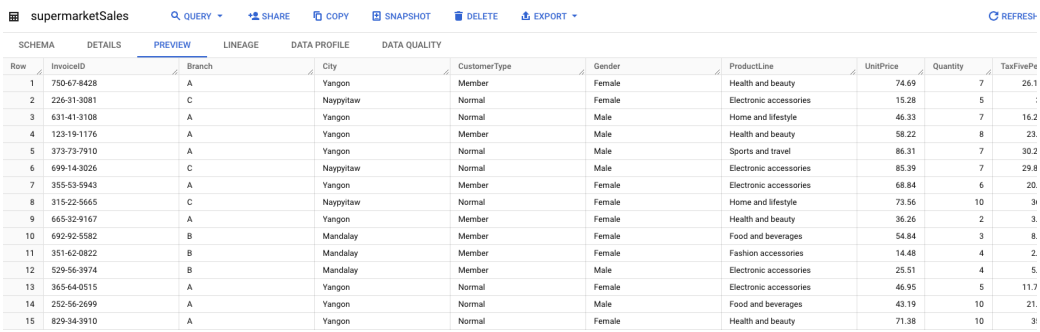


v.

3. Data Storage: route the processed data to the data warehouse (BigQuery).
    a. Ensure the data is available in the GCS.



    b. Ensure the table is created in the BigQuery.



Additional Questions

1. I see lots of manipulation to the input data which I assume the condition in real life would be far from ideal, how a data engineer would handle that?

    In this case, the CSV file collected from Kaggle was converted into a single-line JSON format, then streamed into Kafka, and NiFi would convert it back to a CSV file. The question is, what for? Why should we convert to JSON just to convert it back to CSV?

2. Is there any best practice for handling errors that are caused by different column types in the table so it wouldn't cause a null value?
3. On a company scale, is it needed to route the data from source to Kafka, then ingest it into NiFi which would insert the data into BigQuery, then transform using Informatica

from BigQuery data and put the result in another BigQuery table? Can we actually use Kafka to ingest data straight to BigQuery? And also, what's the point of putting the data into GCS?