

WHITE PAPER

How To Build a Better Data Platform Using SnapLogic

Authors:

Pradeep Sathyanarayan, Software Architect, SnapLogic Software Engineering

Prathyusha Kanala, Manager, SnapLogic Software Engineering



Table of Contents

1. Abstract	3
2. Problem Statement	4
3. Solution Overview	6
4. Introduction	7
5. SnapLogic Platform	8
6. Solution Architecture	9
6.1. Simplified explanation of EtLT	11
6.2. Data Extraction from MongoDB	12
6.3. Data Extraction from Salesforce	13
6.4. Data Extraction from Google Worksheets	14
6.5. Data Extraction from Zendesk	15
6.6. Data Extraction from Aha!	16
6.7. Writing Data into Google Cloud Storage and Loading into Google BigQuery	17
6.8. Writing Data into Amazon S3	18
6.9. Running Analytics on Google BigQuery	19
6.10. Integration with Looker Dashboard	23
6.11. Integration with Amazon SageMaker	26
7. Common Data Problems Solved by the SnapLogic Platform	27
8. Conclusion	29
9. References	30

1. Abstract

“Data is the new oil” as eloquently quoted by the famous mathematician Clive Humby. Just like oil is valuable, so is data, but if it is not refined properly, it cannot be used.^{9.1} Adding context to data provides information which can produce meaningful decisions and actions.^{9.2}

An Enterprise could be at a different level of evolution and maturity driven by economic, business, and regulatory needs. A single legacy source of data is not enough to compete in the fierce marketplace. Business opportunities could be hidden across multiple sources of data, providing contextual insights. Business leaders not only have to overcome the technical constraints of data sources. They must also manage the cost of data processing and compliance, while depending on institutional knowledge amidst depleting workforces.

The bottom line is an enterprise must act fast on the opportunities and do that responsibly and efficiently.

A data platform built on SnapLogic iPaaS helps connect enterprise data and applications. In this manner a raw data which is unorganized can be converted into a big repertoire of information that is organized, presents context and relationship between the various data elements to aid in quick decision making.

This paper discusses the architecture solution of a customer 360 dashboard that has been implemented internally for SnapLogic’s internal consumption and strategic decision making.

2. Problem Statement

As with any enterprise organization, at SnapLogic the customer data required for the 360 dashboard is available in silos spread across disparate systems like Salesforce, MongoDB, Aha!, Zendesk and Google worksheets. There is no unified view of individual customer data.

If one needs to know all the different pipelines created and executed by the customer, they would need to connect to MongoDB. On the other hand, if one needs to know all the details of different tickets and their resolution for a given customer, one would need to login into Zendesk.

Therefore, it was important to unify the data across all these systems and be able to run analytics needed for strategic decision making and actions.

Following are some of the questions that the customer 360 dashboard was expected to answer for each customer:

1. What are the number of open opportunities?
2. What is the annual recurring revenue (ARR)?
3. Who is the current Account Executive from SnapLogic?
4. Who is the Customer Success Manager for this customer?
5. What is the CSAT score for the customer?
6. What is the pricing model used for the customer?
7. What is the support level offered to the customer?
8. When is the next renewal date for the subscription?
9. How many support tickets were raised by the customer?
10. How many support tickets are still open?
11. How many of them have been resolved?
12. What are the new feature requests made by the customer?
13. How is the customer using SnapLogic Platform?
14. How many pipelines are being created?
15. How many pipeline executions are being done?
16. How many documents are processed by the SnapLogic Platform for the customer?
17. How many active users are there in the customer's organization?

Given below is a screenshot of how the customer 360 dashboard would look like for a given customer.

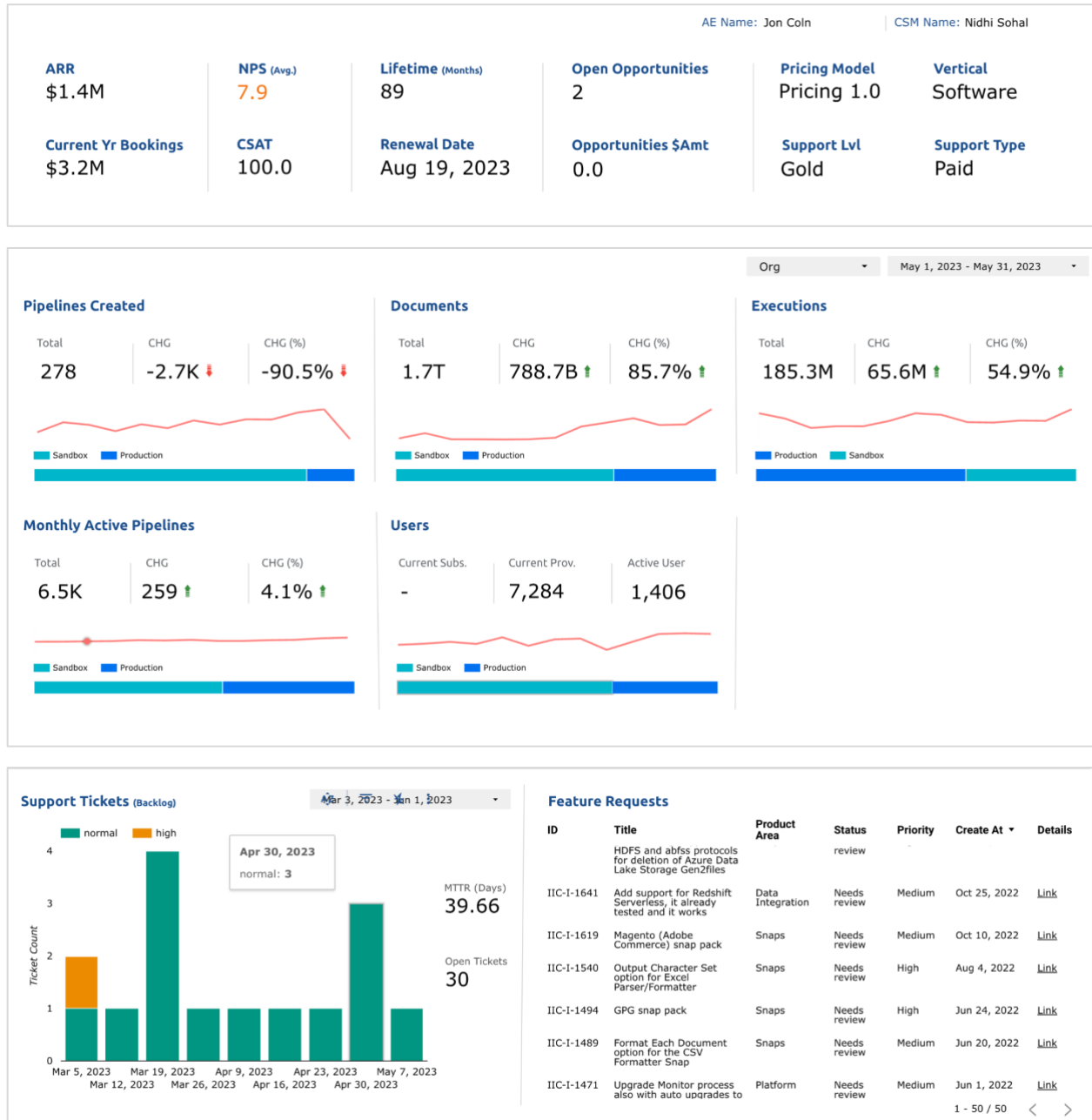


Figure 1: Customer 360 Dashboard

3. Solution Overview

At a very high level, the solution brings in all the disparate data sources mentioned under one umbrella by integrating them using the SnapLogic iPaaS infrastructure. This provides a unified view of a customer 360.

The data would be extracted from all sources, using either REST API calls, SOQL or NOSQL, and then minimally cleansed and transformed on the SnapLogic iPaaS, to land in cloud data storage. Finally, the data is loaded into the cloud data warehouses (CDWs) for further transformation. It is then integrated with Google Looker Studio to provide the unified view.

4. Introduction

Traditionally the data used to be stored in big on prem data warehouses where the data is very structured and has a specific schema. With the advent of Big Data, this paradigm shifted where enterprises started to get data from various sources in all sorts of formats wherein the data was no longer structured. Then came the concept of a data lake, which allows storage of massive amounts of data in its raw format.

The term Data Lake (DL) originated in 2011 from data vendor Pentaho (now Hitachi) as a way to reduce data silos that were forming in Data Warehouse-based ecosystems.^{9.3}

Now there is another term called Data Lakehouse. Databricks defines Data Lakehouse as a new, open data management architecture that combines the flexibility, cost-efficiency, and scale of data lakes with the data management and ACID transactions of data warehouses, enabling business intelligence (BI) and machine learning (ML) on all data. Putting it simply, it is a data warehouse operating on a data lake.

This paper describes how SnapLogic can be used to implement a data platform. It is based on an internal use case Customer 360. It extracts data from disparate sources like Salesforce, Zendesk, MongoDB, Aha! and Google worksheets and brings them all together into Google Cloud storage and Amazon S3 to be loaded into Google BigQuery for analytics and AWS SageMaker for data science and machine learning purposes.

5. SnapLogic Platform

SnapLogic's iPaaS empowers enterprises by automating application, data, and cloud integration.

It consists of two main components:

1. Control Plane
2. Data Plane

The “Control Plane” controls where and how data is processed based on user configuration and preferences.^{9,4}

In other words, it is here where the user can design and construct their integration pipelines using the SnapLogic User Interface and the available snaps catalog.

There are custom snaps that can extract and load data from different source and target endpoints which make data integration tasks simplified for the user.

The “Data Plane” (aka the Snaplex) does the actual processing of data as per the pipelines received from the control plane. No data is stored in the SnapLogic Integration Cloud, instead data is streamed between systems via the Snaplex.^{9,4}

6. Solution Architecture

Following is the high-level architecture of the workflow (Figure 2) designed for implementing a data lakehouse ecosystem for customer 360.

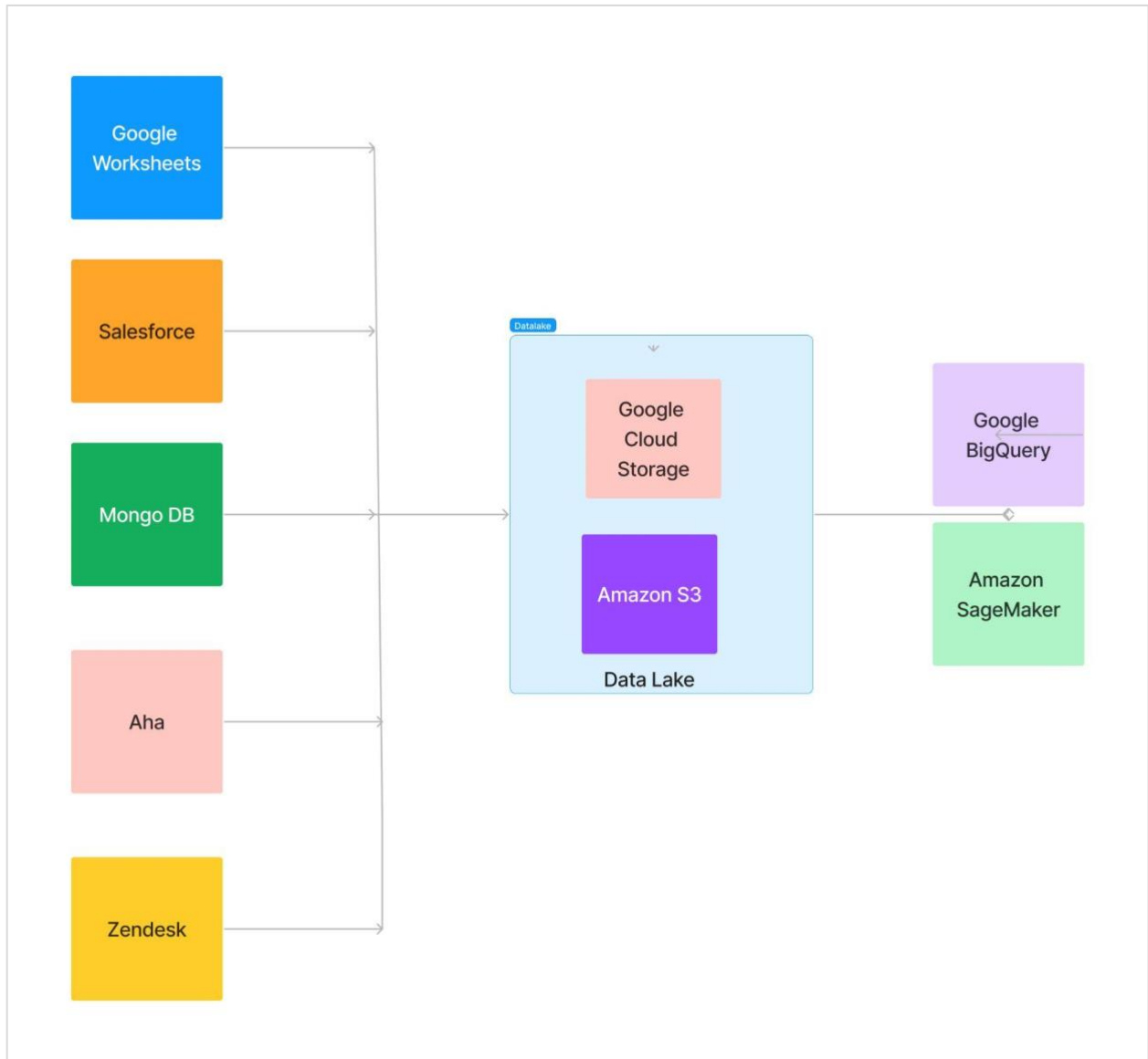


Figure 2: High level workflow for implementing a data platform for customer 360

As can be observed in the above figure (Figure 2), data is extracted from 5 different source endpoints, namely:

1. Salesforce
2. Zendesk
3. Aha!
4. MongoDB
5. Google worksheets

All these data are cleansed, transformed, and landed on to the data lake constituting Google Cloud Storage and Amazon S3.

The data from Google Cloud Storage is then uploaded into Google BigQuery for further transformation and analytics.

The data from Amazon S3 is then loaded into Amazon SageMaker for further data science and machine learning operations.

Google BigQuery has been chosen as a platform for all the analytics for the Customer 360 dashboard because of the following reasons:

1. SnapLogic uses Google Suites across the organization
2. Google BigQuery does not charge the users to load data into BigQuery

Salesforce contains all the customer purchase and other licensing related data.

Aha! carries all the information pertaining to the customers' feature requests and enhancements.

Zendesk hosts all the customers' support related requests.

MongoDB is the database that hosts all the various assets created by the users in SnapLogic that include pipelines, accounts, plexes, tasks, files, pipeline runtime information etc.

Google worksheets contain various customer satisfaction surveys and other customer-specific details.

6.1. Simplified explanation of EtLT

EtLT stands for "Extract transform Load Transform." Typically, most workflow operations either do ETL or ELT, but in this implementation, we have embraced the advantages of both ETL and ELT world to get the best possible performance, functionality, and cost effectiveness for the implementation of the data lakehouse ecosystem.

SnapLogic as a platform has both functionalities of ETL and ELT built into it.

In the EtL part, initially the data is extracted, cleansed, and transformed to a certain format and then loaded into the target Cloud Data Warehouse (CDW). The reason the second "t" is in lower case is to indicate that the transformations are done on a smaller scale on the SnapLogic Platform. Once the data has landed onto the CDW, remaining portions of transformation are done inside the CDW which constitutes the final "T" part in the "EtLT" process.

There are significant advantages to this approach:

1. It speeds up data ingestion
2. It improves data quality
3. Helps in Data compliance and security
4. Reduces Data warehousing costs for compute and storage

Yet again, it really depends upon where the lower case "t" and uppercase "T" is done. In some cases, switching the position of "t" and "T" in the above acronym may make sense if the target CDW processing costs are higher.

6.2. Data Extraction from MongoDB

The following pipeline (Figure 3) is one of the several pipelines that are used in the implementation that extracts data from MongoDB. The workflow helps in determining all the pipelines that were created by a given customer in a given month.

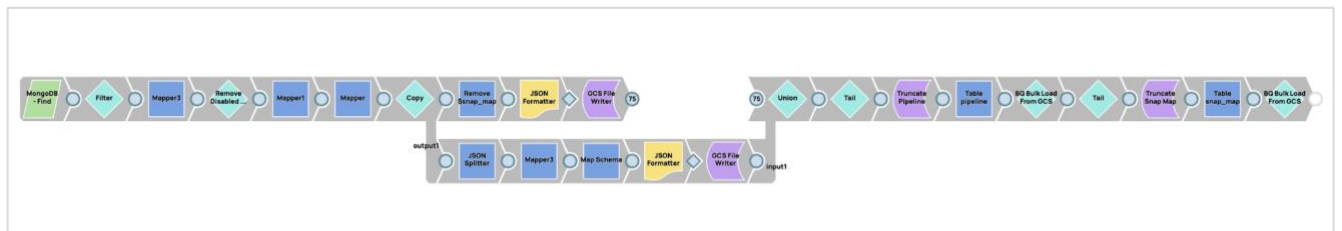


Figure 3: Pipeline that extracts data from MongoDB

For the sake of brevity, only the high-level logic of the workflow has been explained here:

1. Data is extracted from MongoDB using the MongoDB Find Snap
2. For each of the extracted records for an individual pipeline, the complete snap map (indicates the snaps comprising the pipeline) and the link map (that indicates how the snaps are connected) are extracted and transformed
3. This data is written by splitting them into two separate tables in BigQuery by staging the data on Google Cloud Storage

6.3. Data Extraction from Salesforce

The following 6 different pipeline fragments (Figure 4) extract data from different Salesforce Objects. This workflow helps in generating data about the customer details like the name of the customer, support offered to them, the current ARR, Customer Support manager from SnapLogic, the type of billing, software licenses etc.

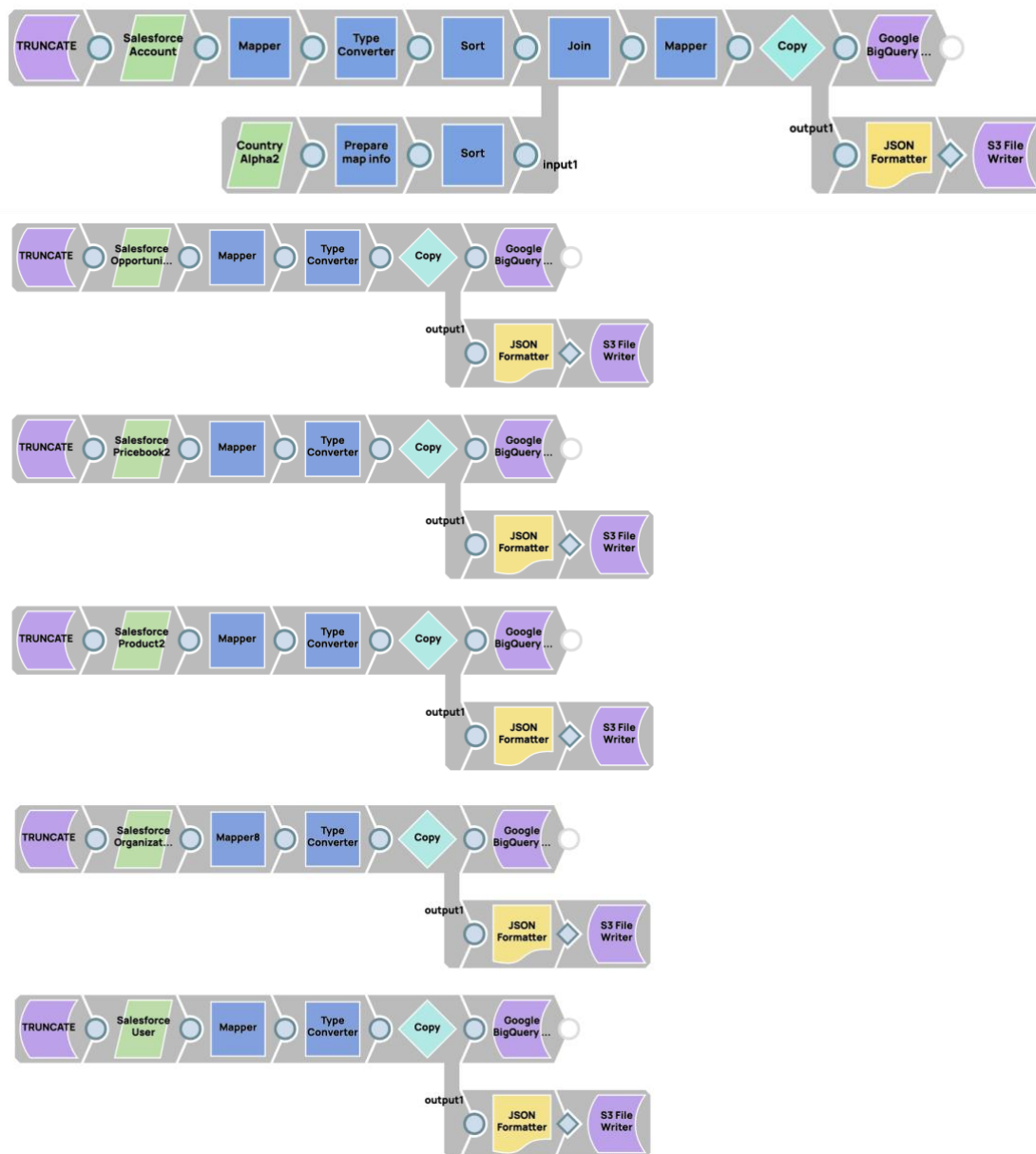


Figure 4: Pipeline fragments that extract data from Salesforce

To keep things simple and succinct, following is the high-level flow of the above pipeline fragments:

1. Data is extracted in parallel from 6 different Salesforce objects
2. Since the data needs to be written into BigQuery and the schema of BigQuery and Salesforce are not fully compatible, the Salesforce Data types are converted into BigQuery Data Types as a part of the transformation phase using the Type Converter Snaps of MLData Preparation snap pack
3. The transformed data is written into the Google Cloud Storage and Amazon S3 using the S3 writer snap
4. Finally, the data is uploaded into BigQuery using the Google BigQuery Bulk Load Snap

6.4. Data Extraction from Google Worksheets

The following workflow (Figure 5) indicates how the data is gathered for the NPS score for every customer.



Figure 5: Pipeline that extracts data from Google worksheets

Below is a high-level summary of how the data is collected from Google worksheets, transformed, and loaded into BigQuery:

1. Initially the target table is truncated on BigQuery in order to upload fresh data from the Worksheets at regular intervals
2. The data is read from the worksheet using the Worksheet reader snap and its data types are converted into the BigQuery using the Type Converter snap
3. It is then sorted and aggregated on the SnapLogic Platform using the Aggregate snap
4. Finally, the data is loaded into BigQuery using Google Cloud Storage as a staging area

6.5. Data Extraction from Zendesk

The following workflow (Figure 6) indicates how the data is gathered for tracking all the technical support related tickets requests for every customer. This information may include the customer name, ticket number, person assigned to work on the ticket, status of the ticket, the date the ticket was last updated, the date the ticket was created etc.



Figure 6: Pipeline that extract data from ZenDesk

Below is a brief description of how the data is collected from Zendesk, transformed, and loaded into BigQuery:

1. Initially the target table is truncated on BigQuery in order to upload fresh data from Zendesk at regular intervals
2. The data is read from Zendesk using the REST GET snap and its data types are converted into the BigQuery using the Type Converter snap
3. Finally, the data is loaded into BigQuery using Google Cloud Storage as a staging area

6.6. Data Extraction from Aha!



Figure 7: Pipeline that extracts data from Aha!

By utilizing the REST API provided by the Aha! portal, it is possible to extract and analyze ideas efficiently, transforming raw data into valuable insights. Once the ideas are split into pages, it then proceeds to extract valuable details about each idea. This includes attributes such as idea title, description, submission date, and votes.

The script in the Script Snap reads input documents, wraps them in a HashMap for easy processing downstream and writes the output documents. The script can be customized to perform more complex transformations based on specific requirements within a SnapLogic pipeline.

The extracted idea details can be further processed and analyzed to identify trends, evaluate the popularity of certain ideas, or perform sentiment analysis. This information can guide product development decisions, enable prioritization of ideas, and provide insights into customer preferences and needs.

6.7. Writing Data into Google Cloud Storage and Loading into Google BigQuery

The following (Figure 8) shows the ease at which SnapLogic connector or Snap Google BigQuery Bulk Load can be configured and used.

In the snap configuration, the user can simply configure the destination table name, its dataset id and Project ID where the extracted data can be loaded into. It also specifies the Google Cloud Storage Bucket name wherein the staging data can be landed onto. The user has the ability to specify the name and the format of the file where the staging data is stored on the Google Cloud Storage bucket. In this example, the format of the input file is JSON. This particular snap has been used across all the pipelines that have been explained before.

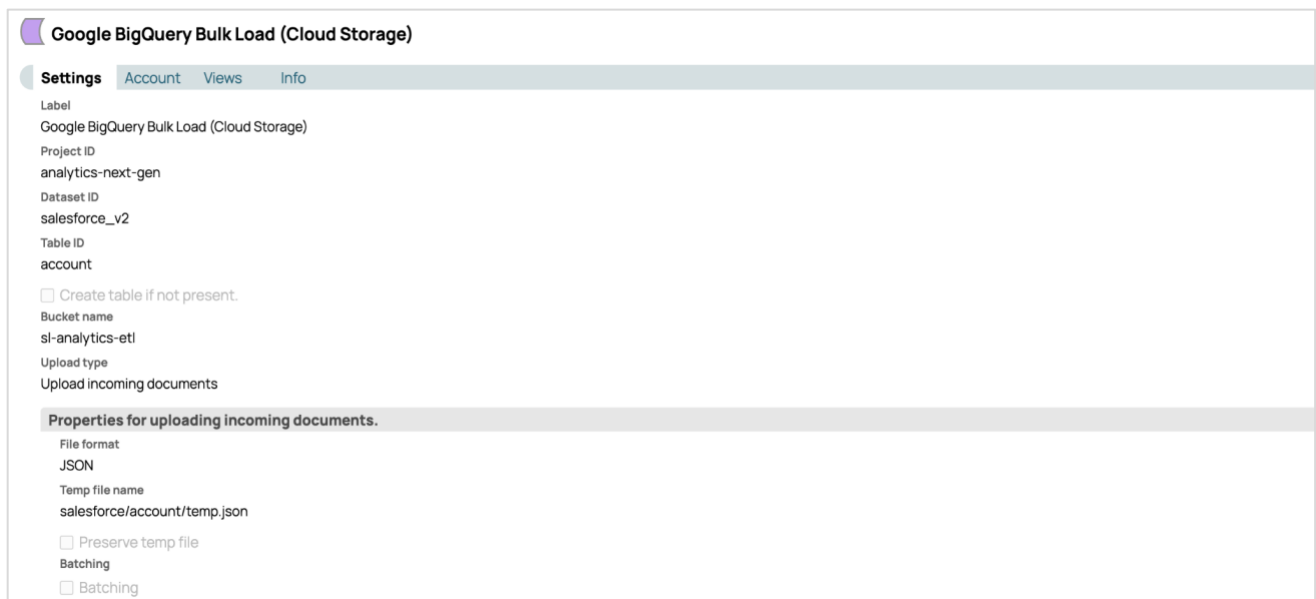
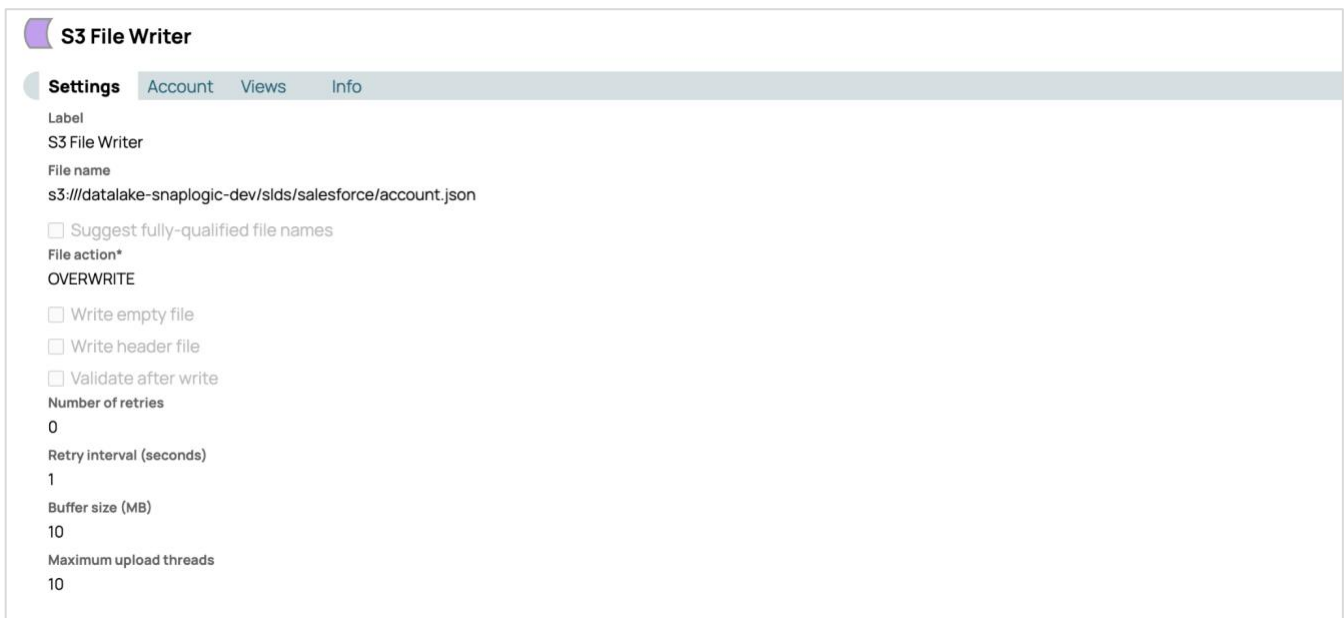


Figure 8: Configuration for Google BigQuery Bulk Load Snap

6.8. Writing Data into Amazon S3

The following (Figure 9) demonstrates the simplicity with which SnapLogic connector or Snap S3 File Writer can be configured and used.

The S3 contents are used further by AWS SageMaker for data science and machine learning. The S3 File Writer comes with a very easy to use interface where the data coming in the form of documents is written directly from the upstream snaps into the chosen S3 bucket and file as seen in the snap configuration. It also offers performance optimization options provided by S3 of multi-part upload allowing multiple threads to make the writes faster and efficient. It also provides buffer-size tuning options in order to provide a throttle control on the write operations.



S3 File Writer

Settings Account Views Info

Label
S3 File Writer

File name
s3:///datalake-snaplogic-dev/slds/salesforce/account.json

Suggest fully-qualified file names

File action*
OVERWRITE

Write empty file

Write header file

Validate after write

Number of retries
0

Retry interval (seconds)
1

Buffer size (MB)
10

Maximum upload threads
10

Figure 9: Configuration for S3 File Writer Snap

6.9. Running Analytics on Google BigQuery

The following SQL query exemplifies the analytics performed in BigQuery as part of the Transformation (T) phase, as described in Section 5.1. Its purpose is to generate pipeline execution statistics by combining data from the "executions per day" and "org_metadata" tables.

```
(
  WITH temp_execution_day AS (
    SELECT A.*, org.org, org.org_type, org.customer FROM (
      SELECT
        DATE(time) AS date,
        doc,
        exec,
        pipe_count_map,
        org_snode_id
      FROM `analytics-next-gen.execution_v4.execution_day`
      WHERE DATE(time) >= DATE_SUB(CURRENT_DATE(), INTERVAL 2 YEAR)
    ) A
    LEFT OUTER JOIN `org_metadata.org_metadata_v2` org ON A.org_snode_id = org.org_snode_id
    WHERE lower(org.customer) != 'snaplogic' AND org.org != '/'
  ),
  temp_exec_pipeline AS (
    SELECT date, org_snode_id, COUNT(DISTINCT JSON_VALUE(pipeline, '$.key')) as pipe FROM (
      SELECT
        date, org_snode_id, pipe_count_map AS pipeline
      FROM temp_execution_day
    ), UNNEST(pipeline) AS pipeline
    GROUP BY date, org_snode_id
  ),
  exec_daily AS(
    SELECT t.*, temp_exec_pipeline.pipe FROM (
      SELECT
        date,
        org_snode_id,
        ANY_VALUE(org) AS org,
        ANY_VALUE(org_type) AS org_type,
```

```

        ANY_VALUE(customer) AS customer,
        SUM(doc) AS doc,
        SUM(exec) AS exec,
    FROM temp_execution_day
    GROUP BY date, org_snode_id
) t
LEFT OUTER JOIN temp_exec_pipeline ON t.date = temp_exec_pipeline.date AND t.org_snode_id =
temp_exec_pipeline.org_snode_id
),
tempable AS (
    SELECT
        org_snode_id, org, org_type, customer,
        -- doc
        SUM(IF(DATE_DIFF(CURRENT_DATE(), date, DAY) <= 30, doc, 0)) AS doc_30d,
        SUM(IF(DATE_DIFF(CURRENT_DATE(), date, DAY) > 30 AND DATE_DIFF(CURRENT_DATE(), date, DAY) <=
60, doc, 0)) AS doc_60d,
        SUM(IF(DATE_DIFF(CURRENT_DATE(), date, DAY) > 60 AND DATE_DIFF(CURRENT_DATE(), date, MONTH)
<= 3, doc, 0)) AS doc_3m,
        SUM(IF(DATE_DIFF(CURRENT_DATE(), date, MONTH) > 3 AND DATE_DIFF(CURRENT_DATE(), date, MONTH)
<= 3, doc, 0)) AS doc_6m,
        SUM(IF(DATE_DIFF(CURRENT_DATE(), date, MONTH) > 6 AND DATE_DIFF(CURRENT_DATE(), date, YEAR)
<= 1, doc, 0)) AS doc_1y,
        SUM(IF(DATE_DIFF(CURRENT_DATE(), date, YEAR) > 1 AND DATE_DIFF(CURRENT_DATE(), date, YEAR)
<= 2, doc, 0)) AS doc_2y,
        -- exec
        SUM(IF(DATE_DIFF(CURRENT_DATE(), date, DAY) <= 30, exec, 0)) AS exec_30d,
        SUM(IF(DATE_DIFF(CURRENT_DATE(), date, DAY) > 30 AND DATE_DIFF(CURRENT_DATE(), date, DAY) <=
60, exec, 0)) AS exec_60d,
        SUM(IF(DATE_DIFF(CURRENT_DATE(), date, DAY) > 60 AND DATE_DIFF(CURRENT_DATE(), date, MONTH)
<= 3, exec, 0)) AS exec_3m,
        SUM(IF(DATE_DIFF(CURRENT_DATE(), date, MONTH) > 3 AND DATE_DIFF(CURRENT_DATE(), date, MONTH)
<= 3, exec, 0)) AS exec_6m,
        SUM(IF(DATE_DIFF(CURRENT_DATE(), date, MONTH) > 6 AND DATE_DIFF(CURRENT_DATE(), date, YEAR)
<= 1, exec, 0)) AS exec_1y,
        SUM(IF(DATE_DIFF(CURRENT_DATE(), date, YEAR) > 1 AND DATE_DIFF(CURRENT_DATE(), date, YEAR)
<= 2, exec, 0)) AS exec_2y,
        -- pipe
        SUM(IF(DATE_DIFF(CURRENT_DATE(), date, DAY) <= 30, pipe, 0)) AS pipe_30d,
        SUM(IF(DATE_DIFF(CURRENT_DATE(), date, DAY) > 30 AND DATE_DIFF(CURRENT_DATE(), date, DAY) <=
60, pipe, 0)) AS pipe_60d,
        SUM(IF(DATE_DIFF(CURRENT_DATE(), date, DAY) > 60 AND DATE_DIFF(CURRENT_DATE(), date, MONTH)
<= 3, pipe, 0)) AS pipe_3m,

```

```

        SUM(IF(DATE_DIFF(CURRENT_DATE(), date, MONTH) > 3 AND DATE_DIFF(CURRENT_DATE(), date, MONTH)
<= 3, pipe, 0)) AS pipe_6m,
        SUM(IF(DATE_DIFF(CURRENT_DATE(), date, MONTH) > 6 AND DATE_DIFF(CURRENT_DATE(), date, YEAR)
<= 1, pipe, 0)) AS pipe_1y,
        SUM(IF(DATE_DIFF(CURRENT_DATE(), date, YEAR) > 1 AND DATE_DIFF(CURRENT_DATE(), date, YEAR)
<= 2, pipe, 0)) AS pipe_2y,
    FROM exec_daily
    GROUP BY org_snode_id, org, org_type, customer
),
time_array AS (
    SELECT ['30D', '3M', '6M', '1Y'] as x
)
SELECT org_snode_id, org, org_type, customer, time,
-- doc
CASE time
    WHEN '30D' THEN doc_30d
    WHEN '3M' THEN doc_30d + doc_60d + doc_3m
    WHEN '6M' THEN doc_30d + doc_60d + doc_3m + doc_6m
    WHEN '1Y' THEN doc_30d + doc_60d + doc_3m + doc_6m + doc_1y
END AS doc,
CASE time
    WHEN '30D' THEN doc_60d
    WHEN '3M' THEN doc_6m
    WHEN '6M' THEN doc_1y
    WHEN '1Y' THEN doc_2y
END AS doc_past,
-- exec
CASE time
    WHEN '30D' THEN exec_30d
    WHEN '3M' THEN exec_30d + exec_60d + exec_3m
    WHEN '6M' THEN exec_30d + exec_60d + exec_3m + exec_6m
    WHEN '1Y' THEN exec_30d + exec_60d + exec_3m + exec_6m + exec_1y
END AS exec,
CASE time
    WHEN '30D' THEN exec_60d
    WHEN '3M' THEN exec_6m
    WHEN '6M' THEN exec_1y
    WHEN '1Y' THEN exec_2y

```

```

END AS exec_past,
-- pipe
CASE time
    WHEN '30D' THEN pipe_30d
    WHEN '3M' THEN pipe_30d + pipe_60d + pipe_3m
    WHEN '6M' THEN pipe_30d + pipe_60d + pipe_3m + pipe_6m
    WHEN '1Y' THEN pipe_30d + pipe_60d + pipe_3m + pipe_6m + pipe_1y
END AS pipe,
CASE time
    WHEN '30D' THEN pipe_60d
    WHEN '3M' THEN pipe_6m
    WHEN '6M' THEN pipe_1y
    WHEN '1Y' THEN pipe_2y
END AS pipe_past
FROM tempable CROSS JOIN time_array, UNNEST(time_array.x) AS time
)
    
```

This query retrieves the number of distinct executions per day for each organization by joining the "executions per day" and "org_metadata" tables. It allows us to analyze the frequency and distribution of executions within each organization on a daily basis.

Additionally, the query performs aggregation on the "executions per day" data, computing the sum of executions at different time intervals. This enables us to understand the overall volume of executions during specific time periods, providing valuable insights into execution patterns and trends.

6.10. Integration with Looker Dashboard

Organizations continuously face the challenge of efficiently accessing and querying large volumes of data to derive actionable insights. Executives often make strategic business decisions based on customer usage data, and customer support heavily relies on real-time information to identify and triage major issues. Additionally, engineering teams strategically allocate resources to address critical and high-impact areas of operation based on this valuable data. These needs necessitate a robust infrastructure and visualization tools such as Looker/Google Data Studio become crucial in enabling ad hoc queries to gather insights.

Currently, Google Cloud Storage and Google BigQuery serve as data sources for dashboards developed in Google Data Studio. This integration empowers users to create visually appealing and interactive dashboards that pull data directly from the data warehouse, ensuring accuracy and real-time information. By consolidating all necessary data within the data warehouse, there is flexibility to perform queries tailored to specific needs, at will. Scheduled jobs can be set up to generate daily, weekly, or monthly reports in easily consumable formats.

Looker provides connectors to a myriad of sources. In this case, it would connect to the BigQuery instance and pull in the appropriate datasets for the report. Reports can also be generated by aggregating data from different datasets and multiple sources if there is such a use case.

Consider the example of generating a pipeline execution statistics report. Once the data is extracted from MongoDB into Google BigQuery (see Figure 3), Looker can be set up to pull in that data to generate a report.

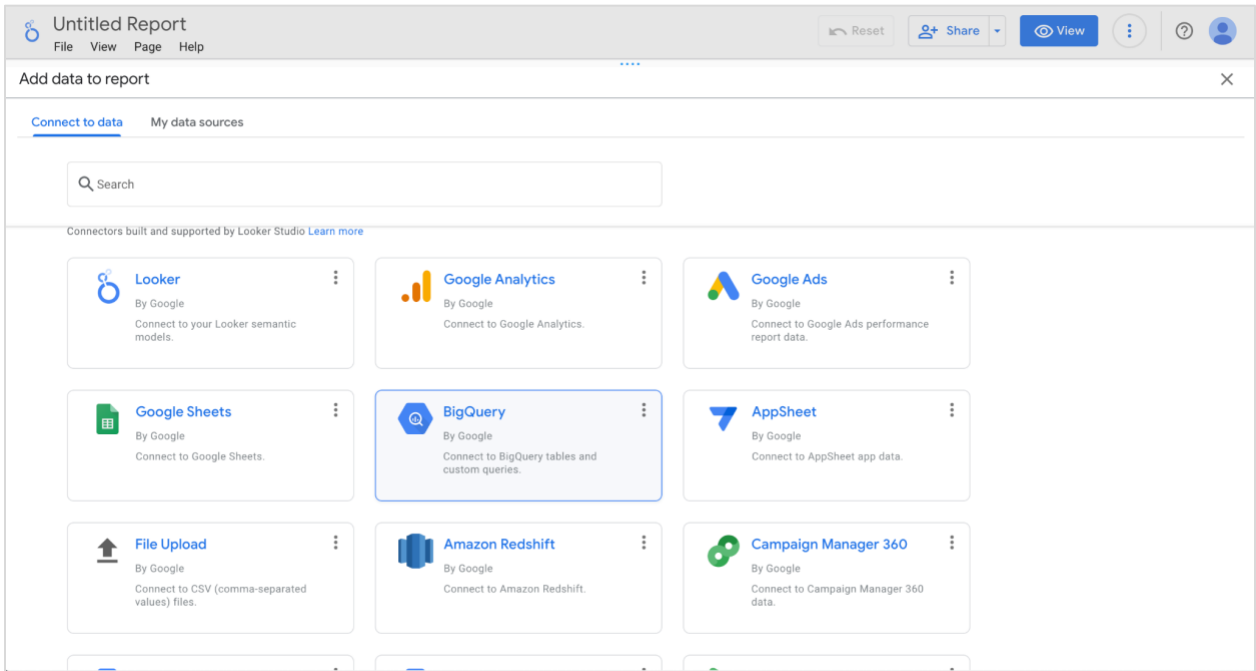


Figure 10: Picking a data source in Looker Dashboard

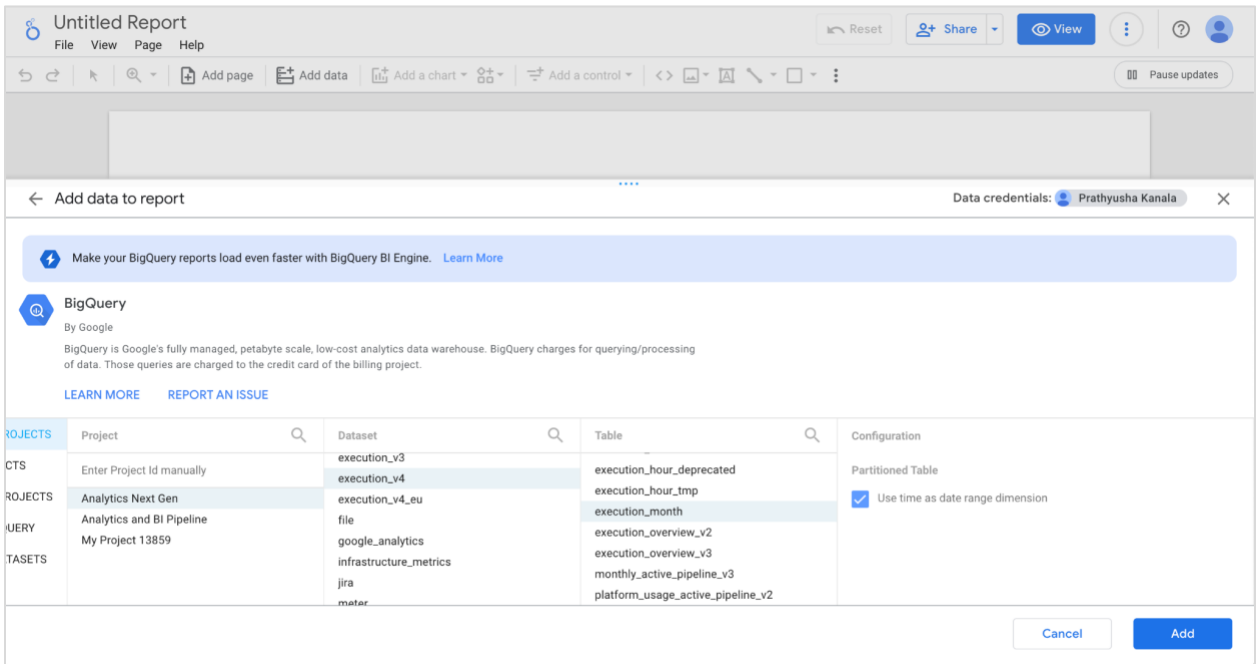


Figure 11: Picking the dataset and the tables to pull data from

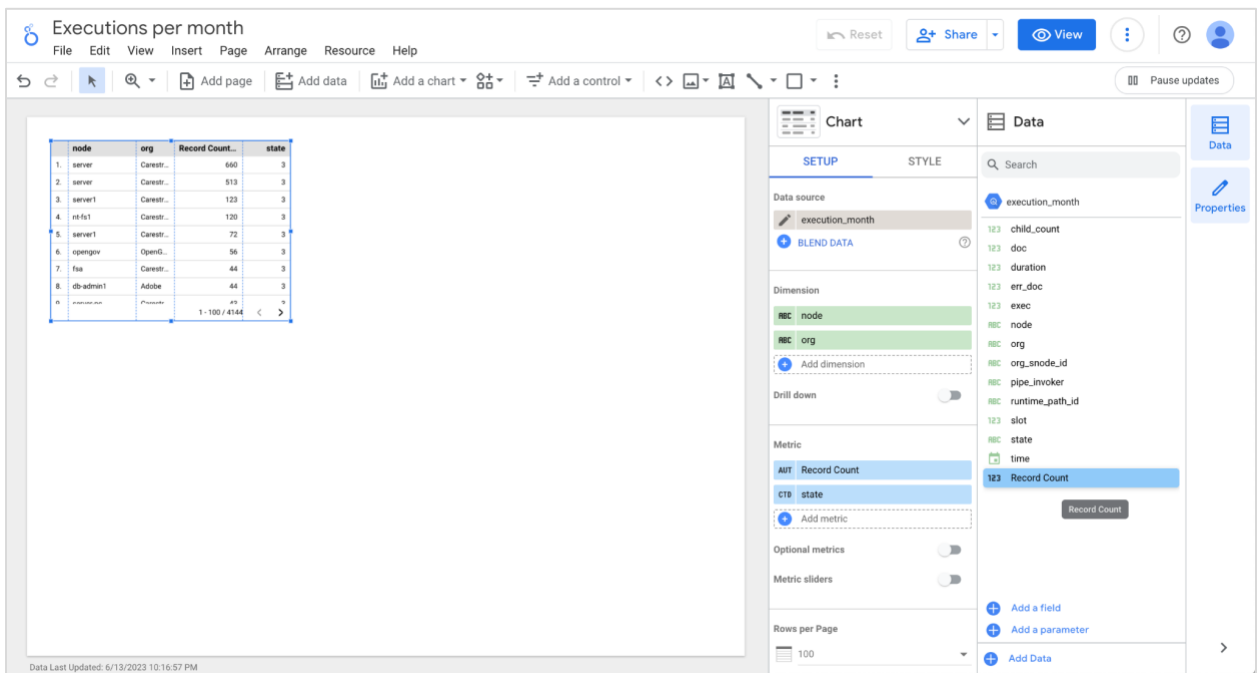


Figure 12: Configuring parameters to generate a report in form of a graph or a pie chart



Figure 13: Example of number of pipeline executions per org and number of documents processed

6.11. Integration with Amazon SageMaker

The runtime data from MongoDB is replicated to S3 in near real-time. With the help of Amazon SageMaker, we leverage the data stored in S3 to train powerful machine learning models. These models are trained using a comprehensive dataset of high-quality pipelines. The dataset undergoes continuous updates to include the latest pipelines executed in the SnapLogic environment, ensuring that the models stay current. This ML based model is used in our recommendation system, which we call Iris AI. Iris AI enhances the pipeline building experience, by offering intelligent snap recommendations tailored to the user's specific needs and contextual information. Once the training is complete, SageMaker provides the capability to easily turn our model into an API endpoint that can efficiently process real-time data from S3.

7. Common Data Problems Solved by the SnapLogic Platform

Typically, when integrating data from disparate systems, several data problems are bound to arise, some of which are listed below:

1. **Schema of the source and the target system are different and/or incompatible:** The SnapLogic Platform provides an easy way to infer the schema from the source systems with its pre-built connectors or snaps and then accordingly aid in converting the schema to the target system using pre-built comprehensive SnapLogic Expression library infrastructure. AutoSync would be an easy alternative if the source and target endpoints are currently supported by it.
2. **Sources which don't carry any schema like CSV and JSON files:** There are many sources like CSV, JSON and AVRO file formats that don't have a pre-built schema in them. But most of the Cloud Data warehouses are strongly typed databases which work on a strongly defined schema. SnapLogic Platform offers connectors like Auto Prep and inbuilt schema inference logic in its ELT Load Snap that can infer the schema of these schema-less sources before loading the data into the target table.
3. **Incompatibilities between source and target data values:** It is not just the schema that could be incompatible, but data values themselves between the source and target systems can be incompatible. For example, date and timestamp formats may not be compatible between source and target systems or float and double data types may not be compatible either because of the precision support each of them carries. SnapLogic Platform offers different options through its plethora of SnapLogic expression capabilities to do these transformations to make the data values compatible.
4. **API limits on the source systems:** Data needs to be extracted from multiple sources while performing such complex integrations. Several data sources impose limits on the number of API calls that can be made to extract data from them. SnapLogic Platform offers capabilities by offering scheduling capabilities of the various data extraction tasks to avoid such data contention issues imposed by API limits.

5. **Data security compliance:** Security is the key and compliance to Data security is an implicit requirement of the day. Data Masking or eliminating data columns containing sensitive data during the extraction of the data are some of the capabilities that SnapLogic Platform offers through its plethora of connectors. SnapLogic offers connectors which can encrypt the data during transit and decrypt them at the destination.
6. **Performing complex analytics with SQL:** Even though SQL is a widely used language for all data analytics, it becomes extremely complex to write SQLs which have complex transformations and debugging them is even more hard. The SnapLogic Platform offers the ELT Snap pack that provides an easy and a very convenient way for the end users to write complex SQL in a simplified GUI.
7. **Capturing historical data using SCD2:** One of the key requirements of any workflow is to capture historical data which describes the changes happening to the data in terms of what row was updated and storing all the prior updated values of that row. This is very useful for auditing and for historical reference. SnapLogic provides automated connectors that can be used in the pipelines which would capture this data in the most seamless manner.

8. Conclusion

The number of disparate data sources with time. Each source application is expected to produce data in its own format, as each of these applications would address specific business needs. The need for a unified view of all the different data will continue to grow, along with the need to bring this raw, unstructured data into data lakes.

As businesses fiercely compete in this ever-changing economic climate, it becomes even more important to be able to integrate all the different facets of business in the most fast, efficient manner in order to make strategic decisions and execute these decisions quickly.

This is where platforms like SnapLogic will play a pivotal role in the rapidly changing enterprise automation environment.

The topics discussed here provide just a sneak peak of the extensive and deep capabilities that the SnapLogic iPaaS can offer for building a better data platform. Only imagination can be the limit.

9. References

- 9.1. <https://www.forbes.com/sites/nishatalagala/2022/03/02/data-as-the-new-oil-is-not-enough-four-principles-for-avoiding-data-fires/?sh=3bb856a3c208>
- 9.2. <https://bloomfire.com/blog/data-vs-information/>
- 9.3. <https://medium.com/quantumblack/lakes-warehouses-lakehouses-a-short-history-of-data-architecture-bc942b0ed463>
- 9.4. <https://www.snaplogic.com/blog/software-defined-integration>