What is google cloud dataflow?

1. What is data flow?

Google Cloud Dataflow is a managed service for building data processing pipelines. It allows you to write code that defines how data should be processed, and then Dataflow takes care of running that code on a scalable and reliable infrastructure.

Here's a practical example of how Dataflow can be used:

Use Case: Real-time analytics for website traffic

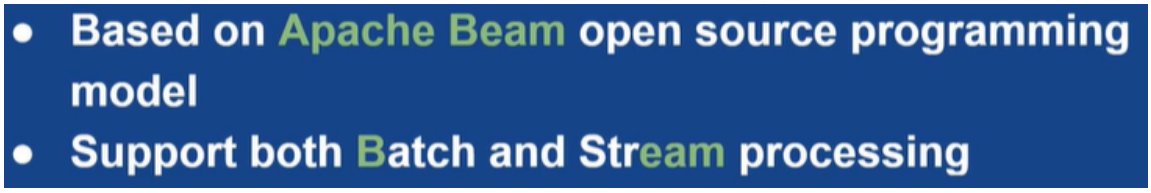
Imagine you run an e-commerce website and you want to analyze website traffic in real-time. You might collect data about user clicks, page views, and purchases. This data can be streamed into Google Cloud Pub/Sub, a real-time messaging service.

Dataflow can then be used to create a pipeline that processes this data stream. The pipeline could:

1. **Extract** data from Pub/Sub.
2. **Transform** the data by filtering out unwanted information and formatting it for analysis.
3. **Load** the transformed data into BigQuery, a data warehouse service for large datasets.

Once the data is in BigQuery, you can use business intelligence (BI) tools to create dashboards that show real-time insights into your website traffic. This can help you understand what products are popular, how users are navigating your site, and identify any potential issues.

2. It is Serverless too
3. ANd it is based on



beam is nothing but batch and stream

4. Next we have templates

In Google Cloud Dataflow, templates are pre-built or user-defined pipelines that offer a convenient way to run data processing jobs without writing code from scratch. They act as blueprints for data processing tasks.

There are two main types of Dataflow templates:

- **Flex templates:** These are the recommended approach for new templates. They package the pipeline code as a Docker image and utilize a separate template specification file. This allows for greater flexibility and isolation of dependencies.
- **Classic templates:** These are the older type of template and contain the entire pipeline code within the template file itself. They are simpler to set up but less flexible compared to Flex templates.

Example: Using the WordCount Template

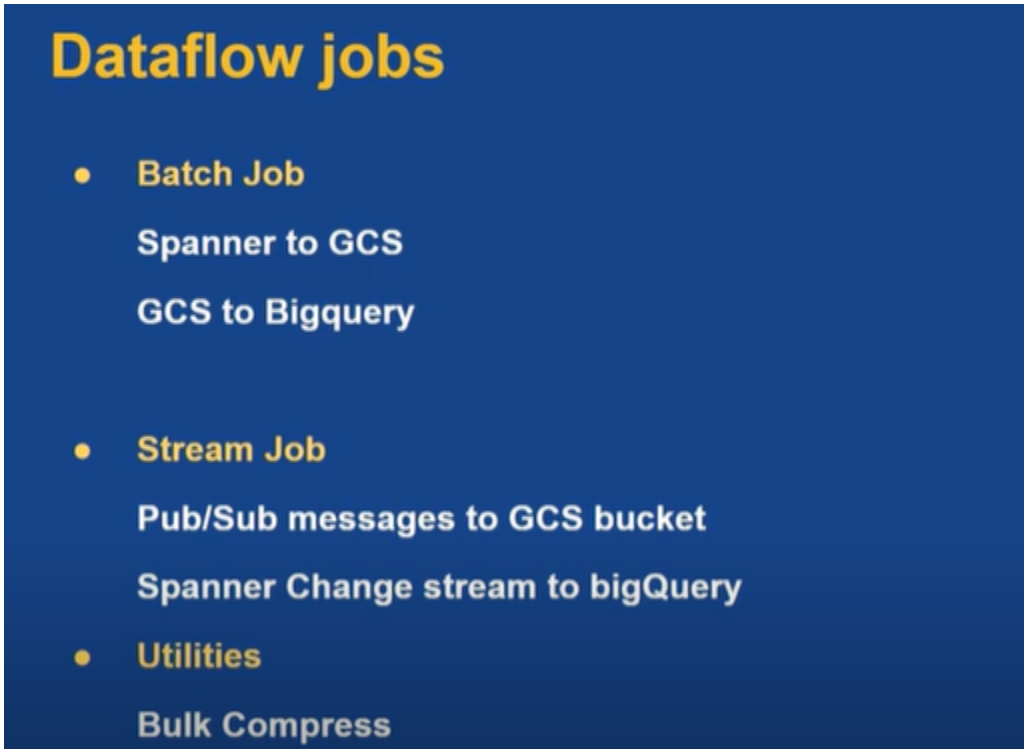
Dataflow offers a pre-built Flex template for a simple word count job. This template takes text data from a Cloud Storage bucket, counts the occurrences of each word, and writes the results to another Cloud Storage bucket.

Here's how you can use it:

1. Go to the Dataflow **Create job from template** page in the Google Cloud Console.
2. Enter a unique job name.
3. Select the **WordCount** template from the dropdown menu.
4. Configure the template parameters:
    - **Input GCS bucket:** Specify the Cloud Storage bucket containing the text data.
    - **Output GCS bucket:** Specify the Cloud Storage bucket where the word count results will be written.
5. Click **Run job**.

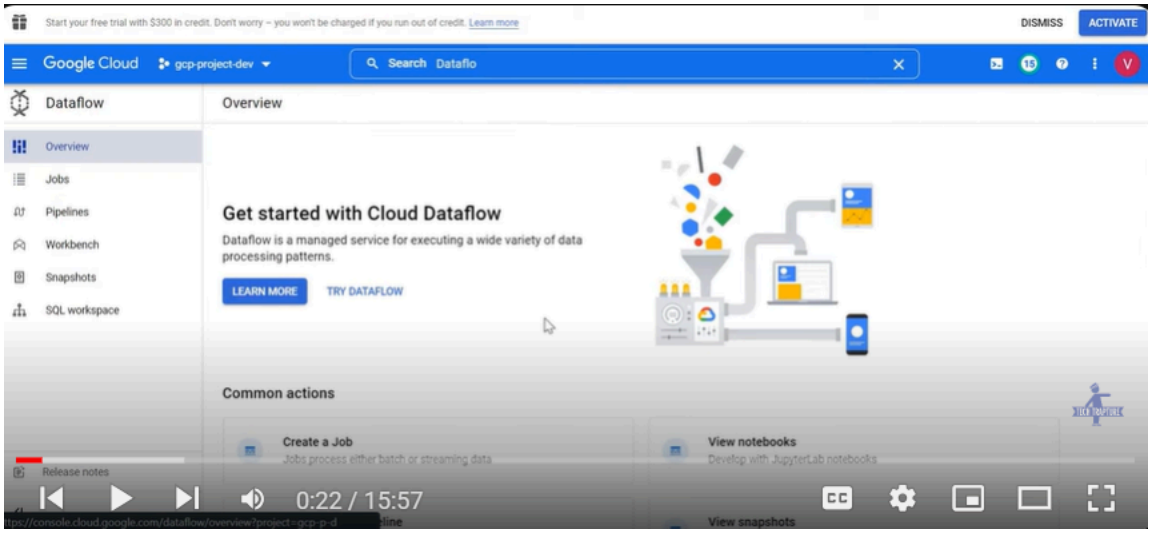
5. Practical

6. Next we will create batch job and stream job



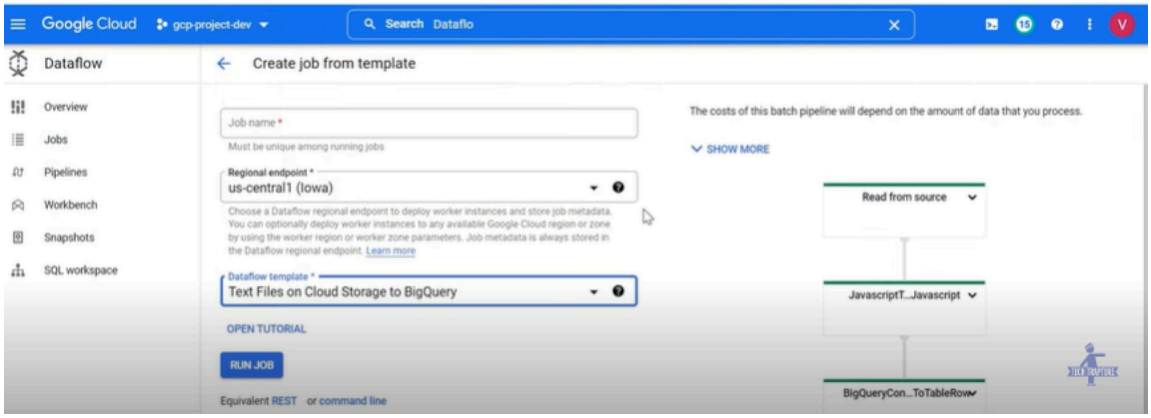
Load data from GCS to BigQuery using DataFlow

1. Here this is the dataflow UI



2. Next we go to jobs and create job from template
3. Here first we focus on batch templates..

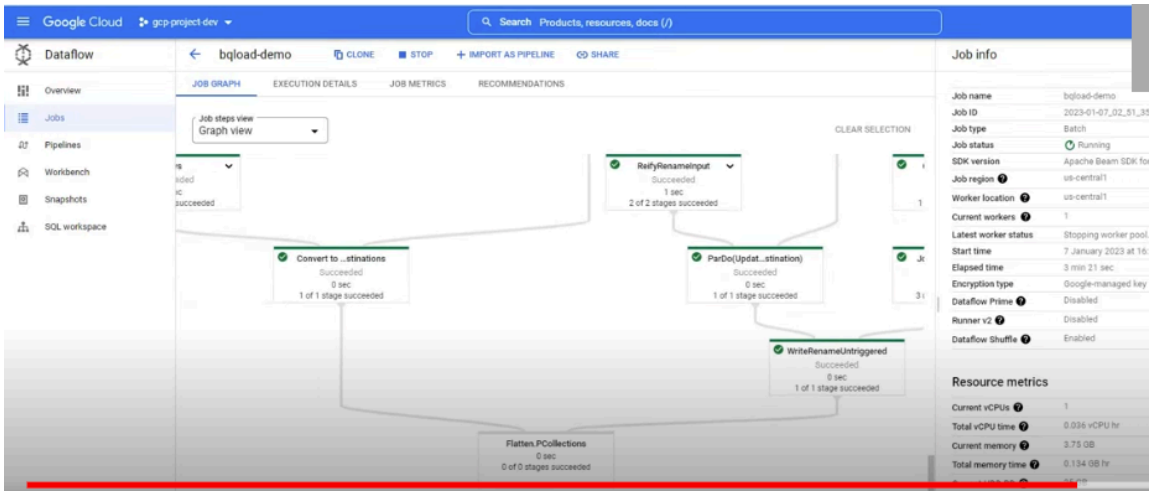
4. First we used text files on cloud storage to bigquery template



5. We have this data…and we load few cols to bigquery
6. Next we have to give required parameters
7. And we create a new cloud storage bucket..and later we upload the js,csv and json files
8. Next we pass this values in parameters

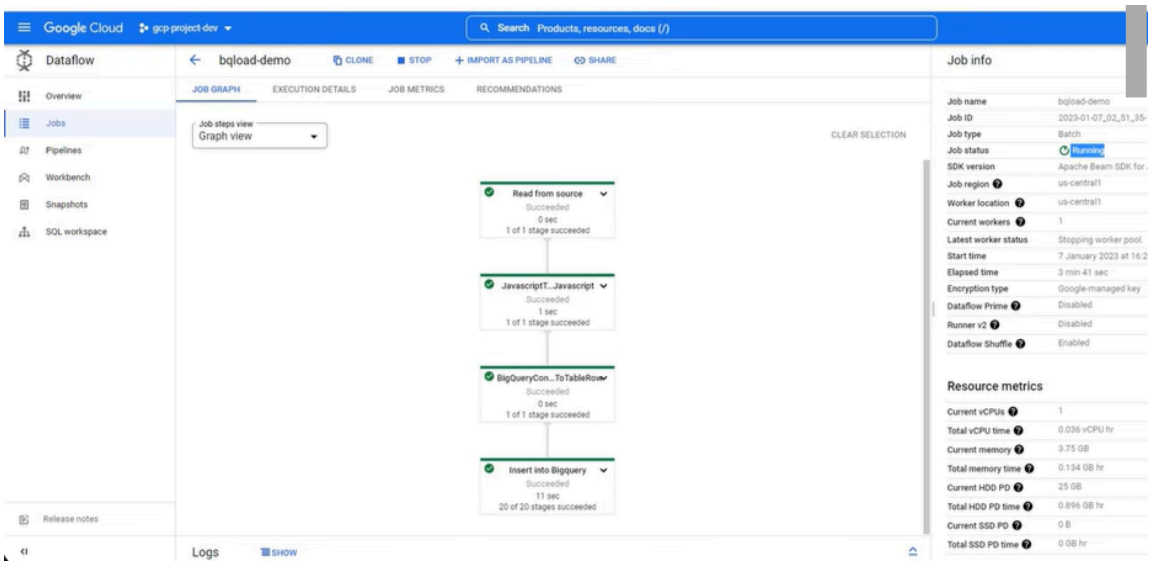
9. And also we have to create a dataset in bigquery
10. Later we run the job
11. Now in the job section…we can see the details for our job

12. This is our job graph

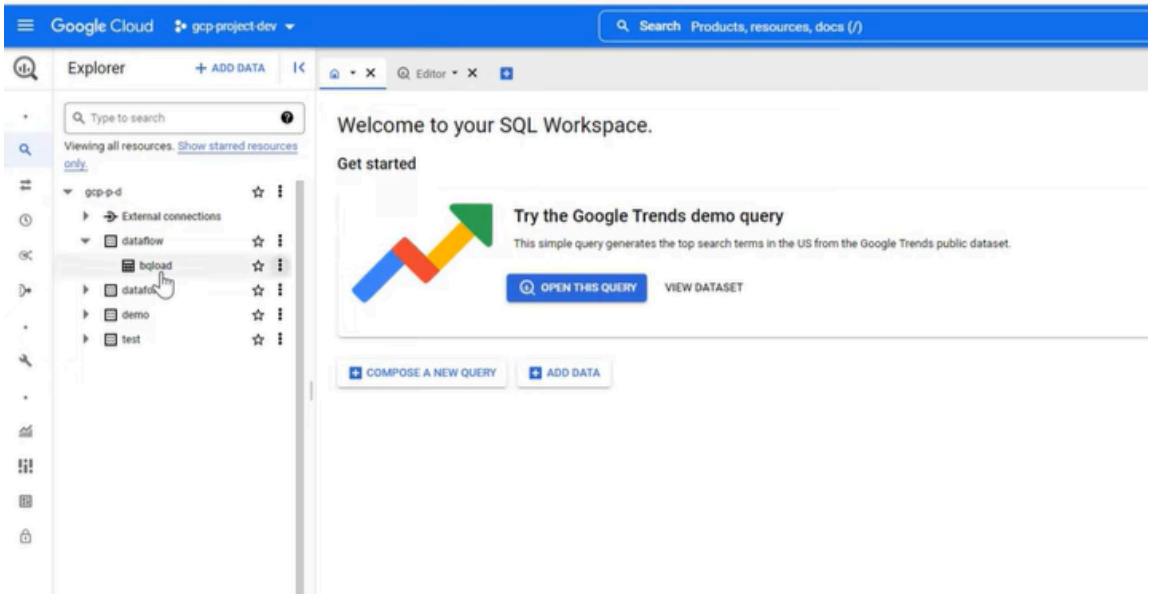


if there are any errors..then we can rectify at which stage our jon failed

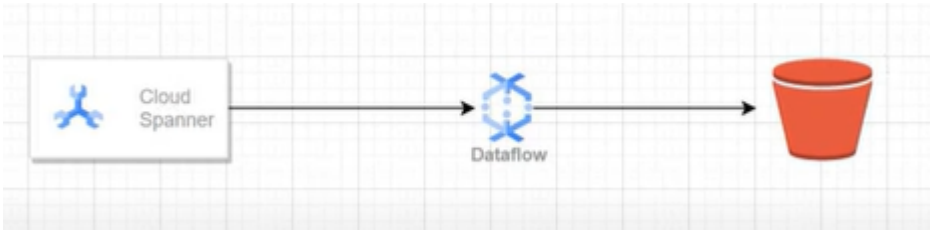
13. Now every thing is completed



we can our data in bigquery



Extract Data from cloud spanner to GCS using DataFlow

1. Here dataflow will extract the data from cloud spanner and store the data in cloud storage 

Basically dataflow retrieves the data from source and loads in destination

2. What is cloud spanner?

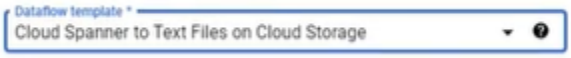
Google Cloud Spanner is a relational database service built specifically for the cloud. It combines the familiar features of relational databases, like SQL queries and strong consistency, with the scalability and flexibility of non-relational databases. Here's a breakdown of what Cloud Spanner

Imagine a large retail chain with stores across different countries. They need a database to track their inventory levels in real-time across all locations. A traditional relational database might struggle with this due to scalability limitations and potential consistency issues if data is spread across multiple servers.

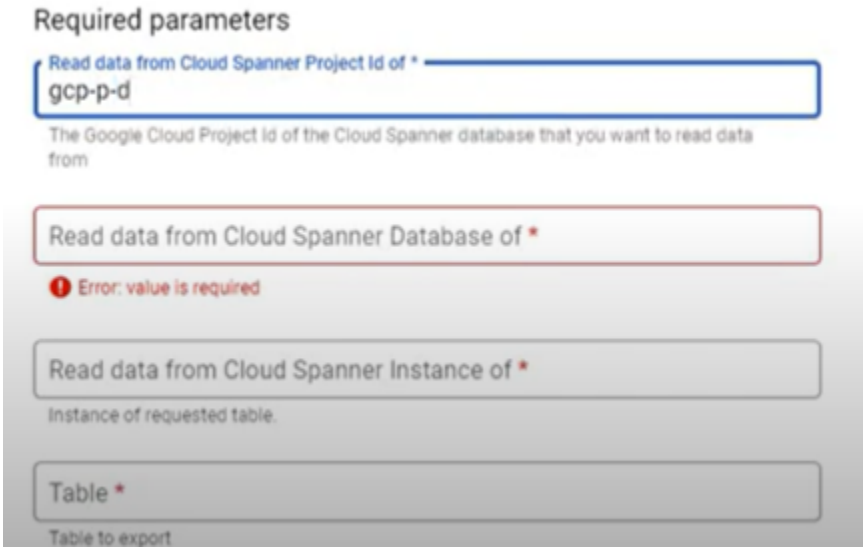
Cloud Spanner can be a perfect solution for this scenario. Here's how:

1. **Data Model:** The retail chain can design a schema in Spanner to store information like product IDs, quantities, and store locations.
2. **Global Distribution:** Spanner can store the inventory data across multiple regions where the stores are located. This ensures low latency for queries and high availability in case of regional outages.
3. **Real-time Updates:** Whenever a sale occurs in a store, the inventory level for that product in that location can be updated in Spanner.
4. **Strong Consistency:** Due to strong consistency, the updated inventory level is immediately reflected across all replicas, ensuring all stores have the latest information.
5. **Scalability:** As the retail chain grows and their inventory volume increases, Spanner can automatically scale to accommodate the additional data.

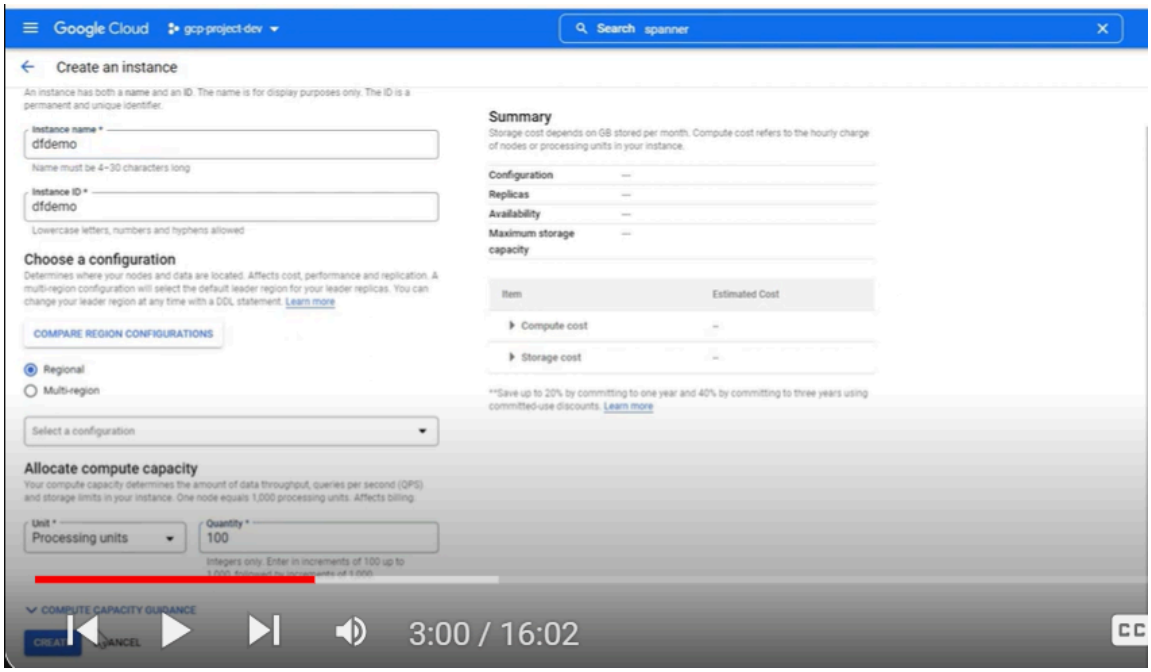
3. Now lets create a job template …and select dataflow template



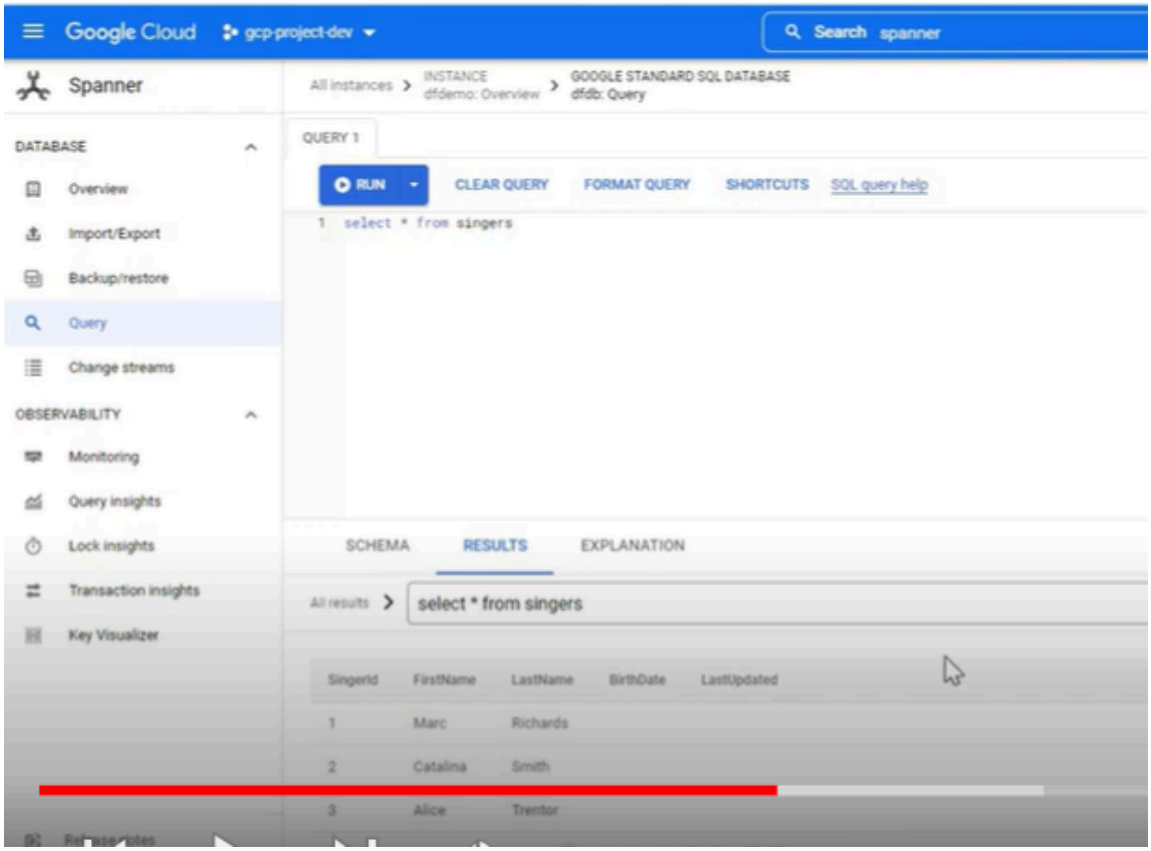
4. Next we need this parameters from cloud spanner



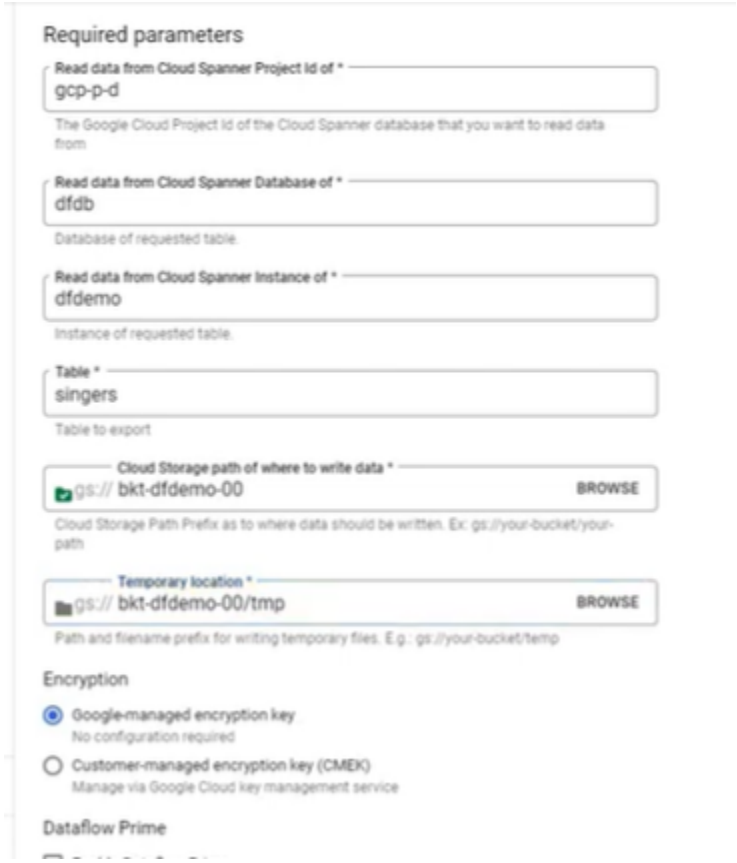
5. Next we will create cloud spanner instance to provide the required parametrs



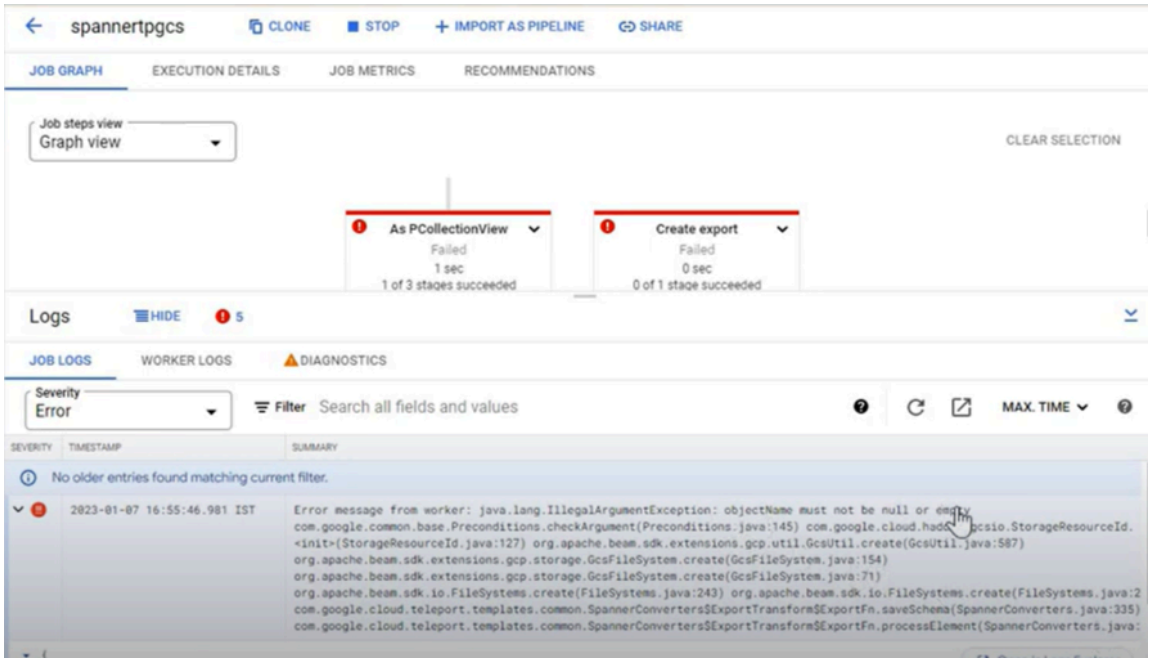
6. Next we create a db and a table and insert sample data
7. Now we will export these 3 rows into csv file and load in GCS



8. Here we gave the required parameters

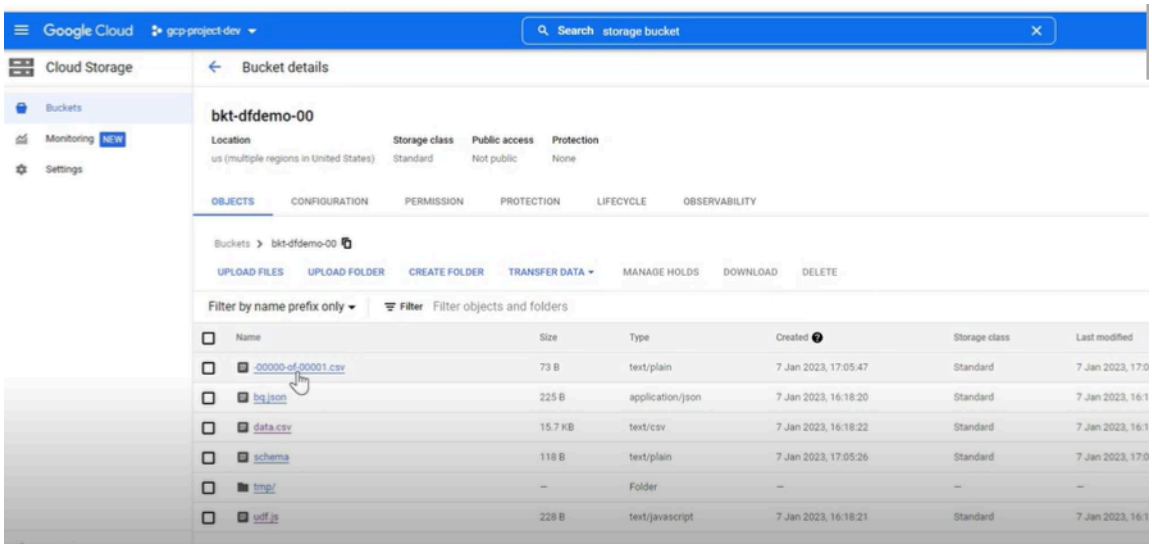


9. Here we got an error…which can be identified using job logs



10. The issue here was…we gave table name as singers..it has to be ‘Singers’

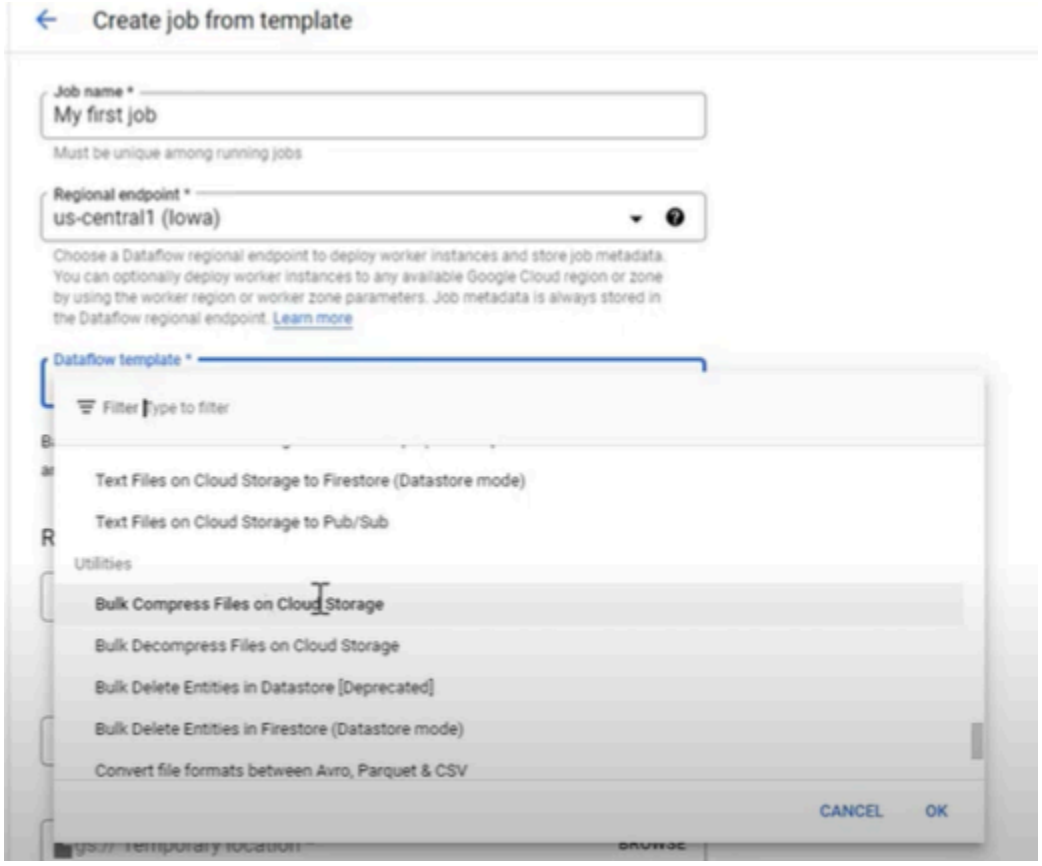
11. After the job completion..we can see our data in cloud storage



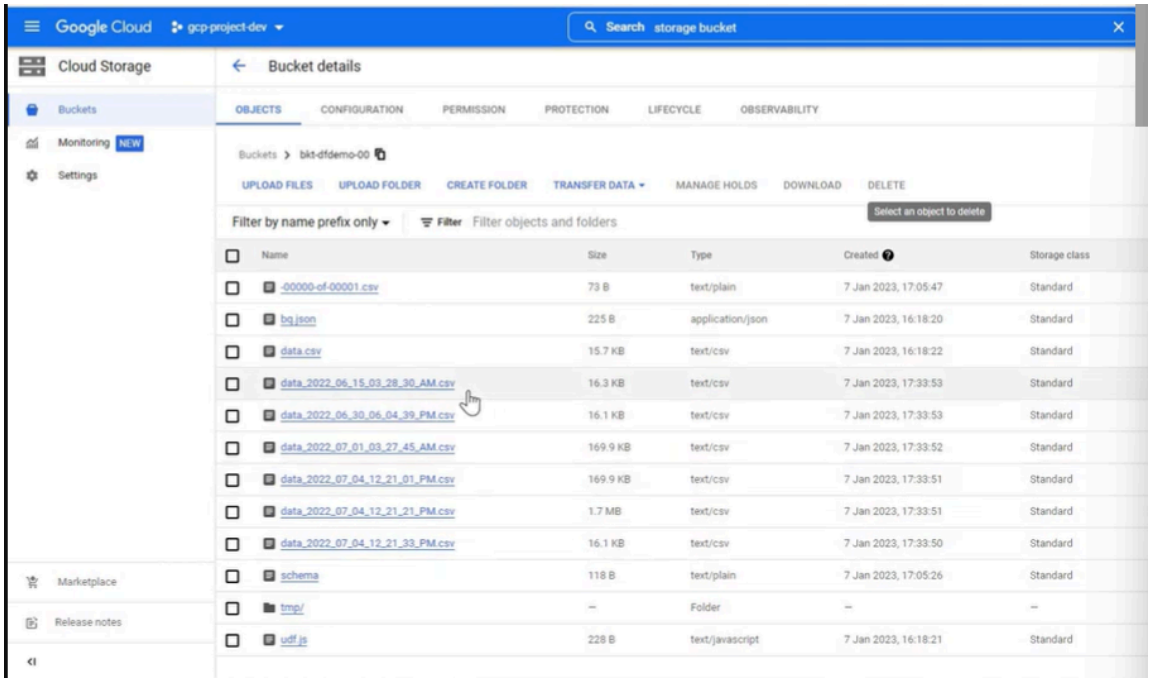
Compress large files in bulk using dataflow

1. We create a job …which compress the files from GCS

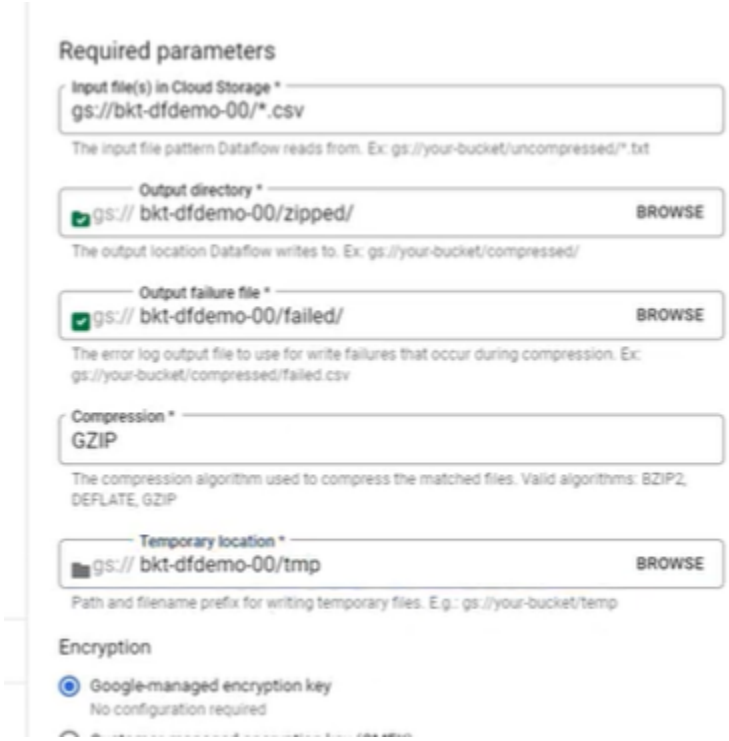
2. Here we have to choose bulk compress files in cloud storage template



3. Here we have multiple CSV fles…and we want to compress the csv files



4. Next we give the required parameters



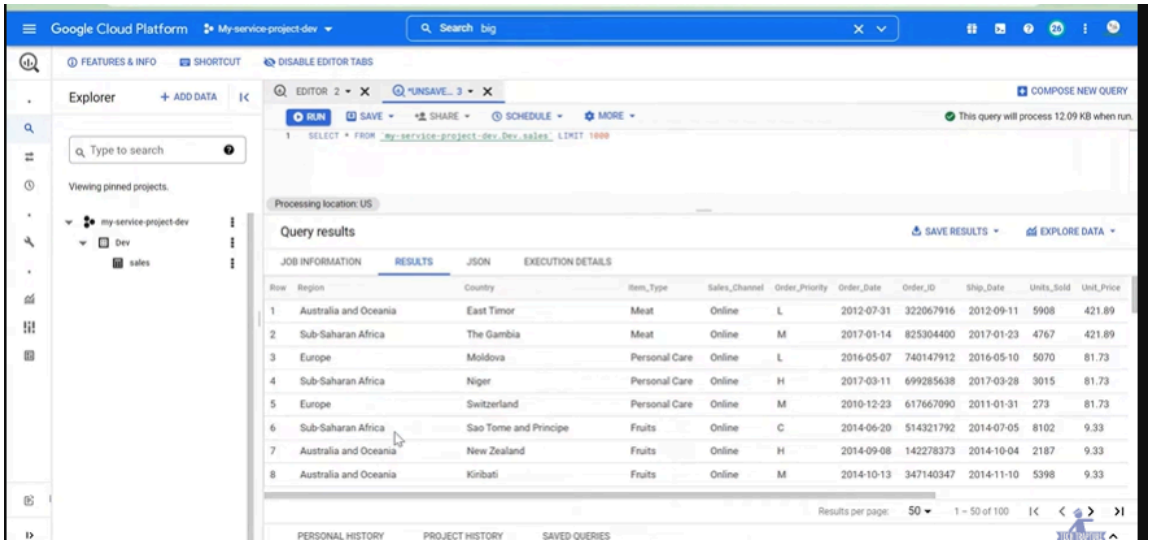
5. Here this will only compresses the csv files

6. When we run the job…it will automatically create the resources that are required for the jobs
7. Here we have zip file..which contains all the compressed csv files

Create DataFlow Job using Dataflow SQL

1. We will writing queries on bigquery to create dataflow job

2. Inside the bigquery…we have project->dataset->table

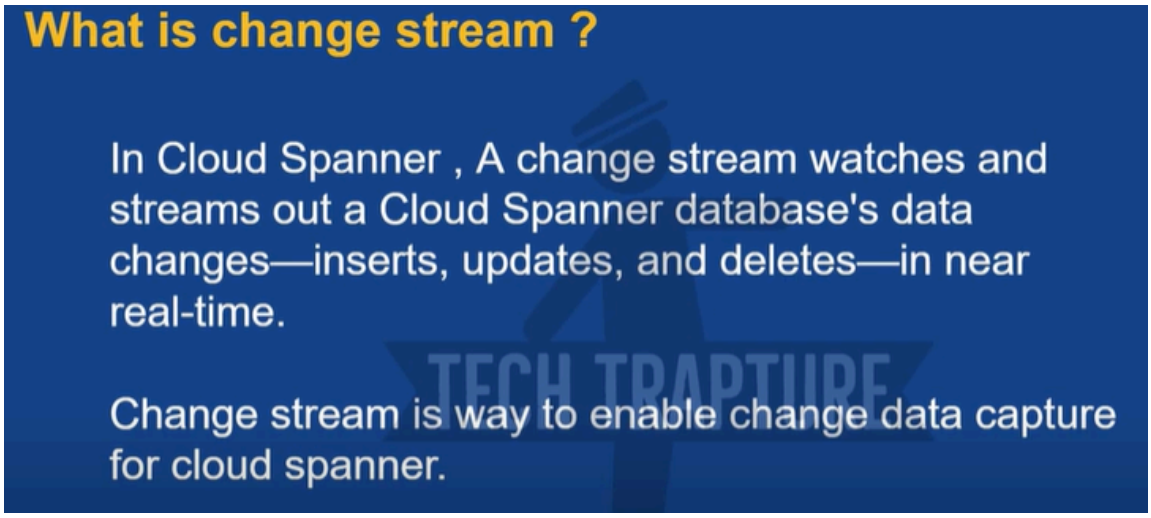


here we can see the table data

3. Now if want to see data where region is ‘Europe’

Create Change Stream in Spanner and stream into BigQuery

1. Change Stream helps us in CDC

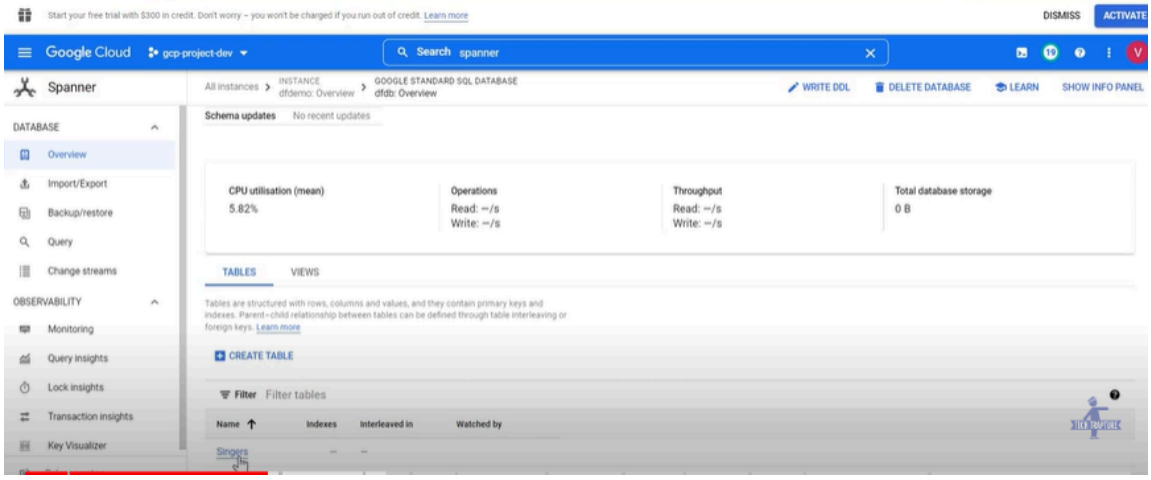


2. What is change data capture?

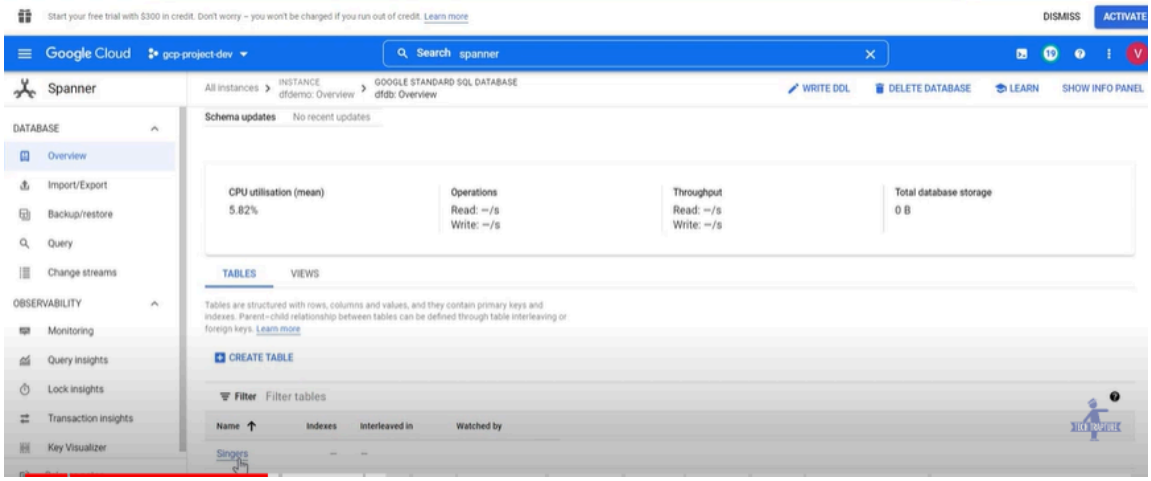
Change data capture (CDC) is a technique for identifying and tracking changes made to data in a database. It essentially captures the modifications (inserts, updates, deletes) happening to specific tables over time. This captured data can then be used for various purposes, such as:

- **Real-time analytics:** CDC allows you to analyze changes as they occur, enabling applications to react to updates promptly.
- **Data warehousing and synchronization:** Captured changes can be efficiently transferred to data warehouses or other systems for further analysis or to keep them synchronized.
- **Audit trails and compliance:** CDC provides a record of data modifications, which can be helpful for auditing purposes and ensuring regulatory compliance.

3. Practical
4. Here we have a singers table in spanner instance

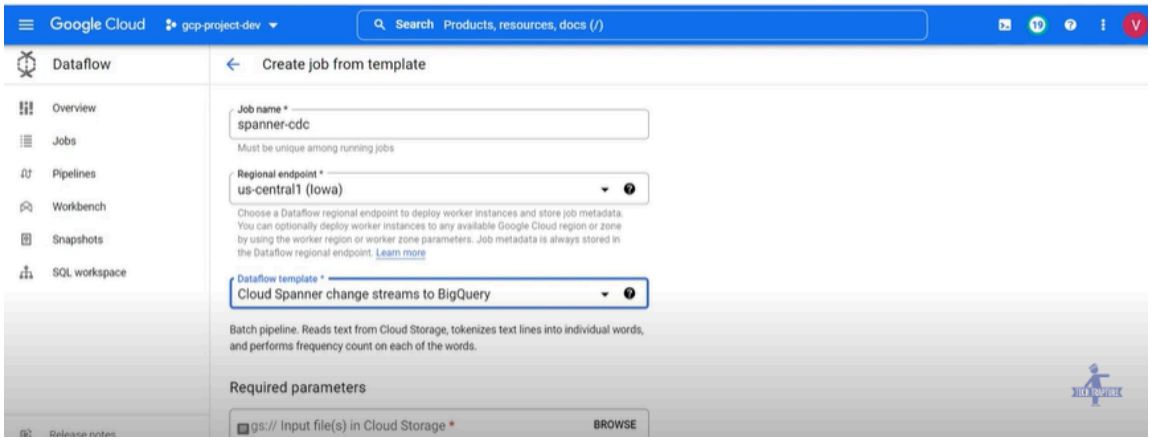


5. On left side we have change stream click on it…and create change stream for All

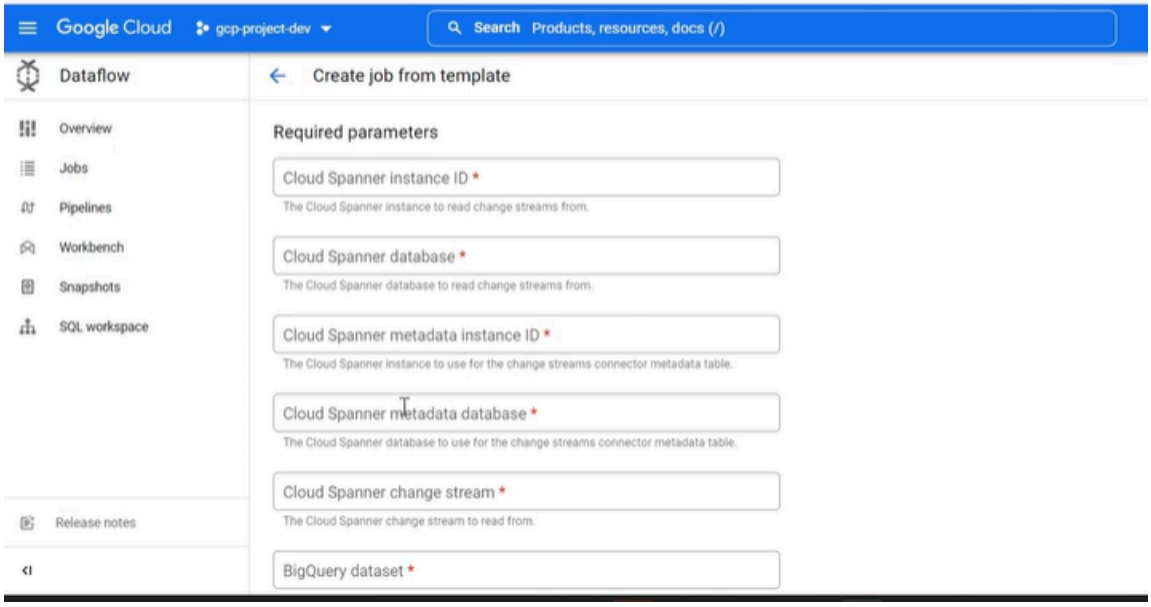


6. Next we need to create a dataflow job

7. And we give a new job name and dataflow template



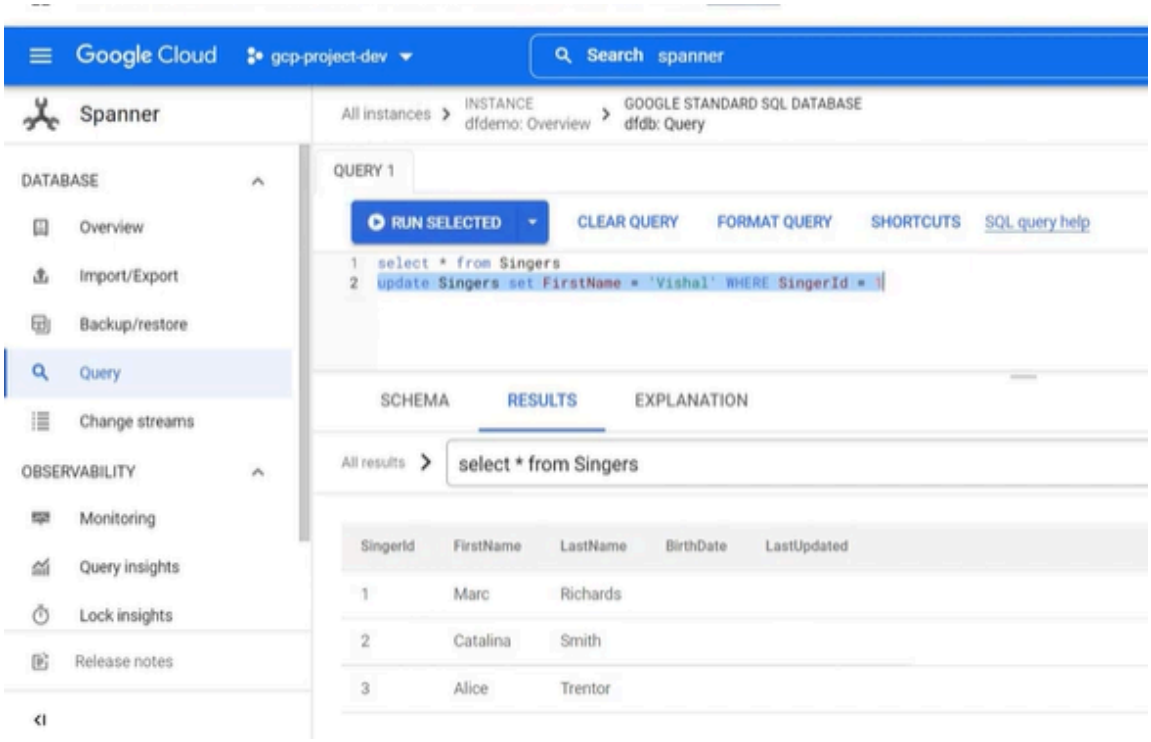
8. Here we need theses parameters



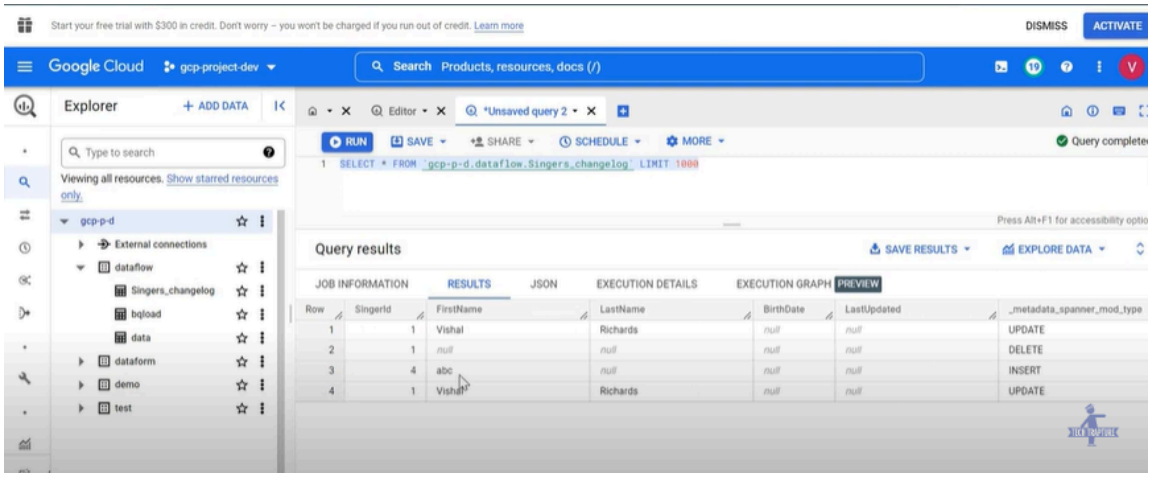
9. We create new spanner instance and mt DB..and give them as parameters

10. next we give CDC that we created and a bigquery dataset
11. And run this job

12. Now lets change any row in the singers table

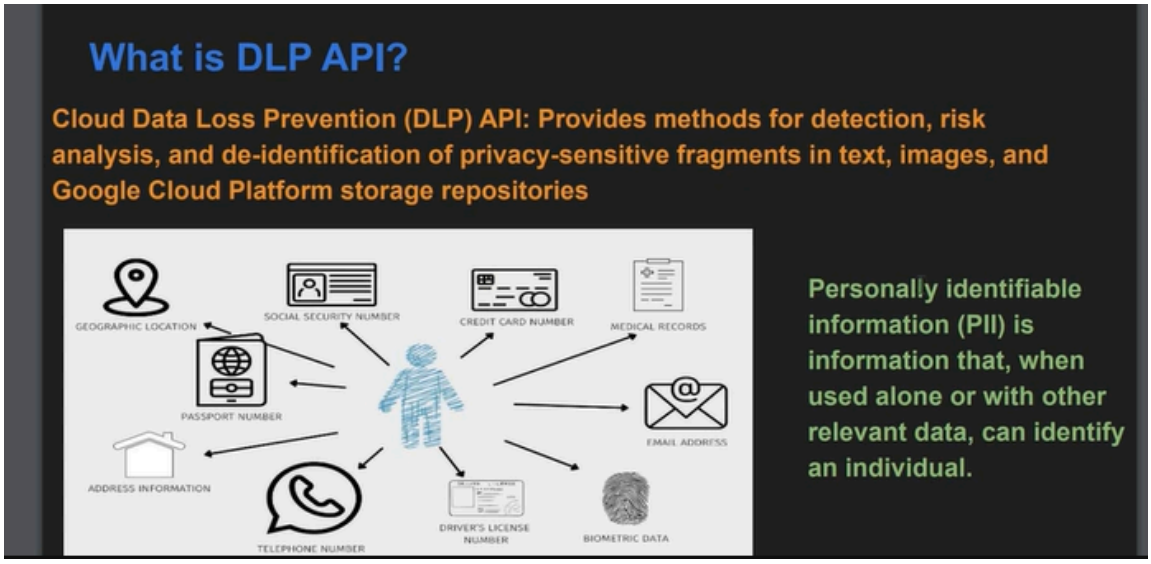


13. Now here in the bigquery…we can all the changes made in our singers table

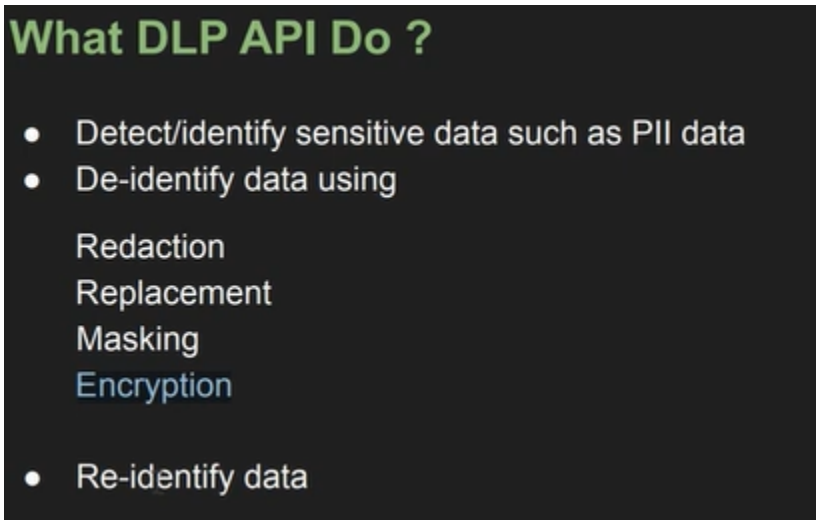


Data Loss Prevention (DLP) API in GCP

1. What is DLP API?

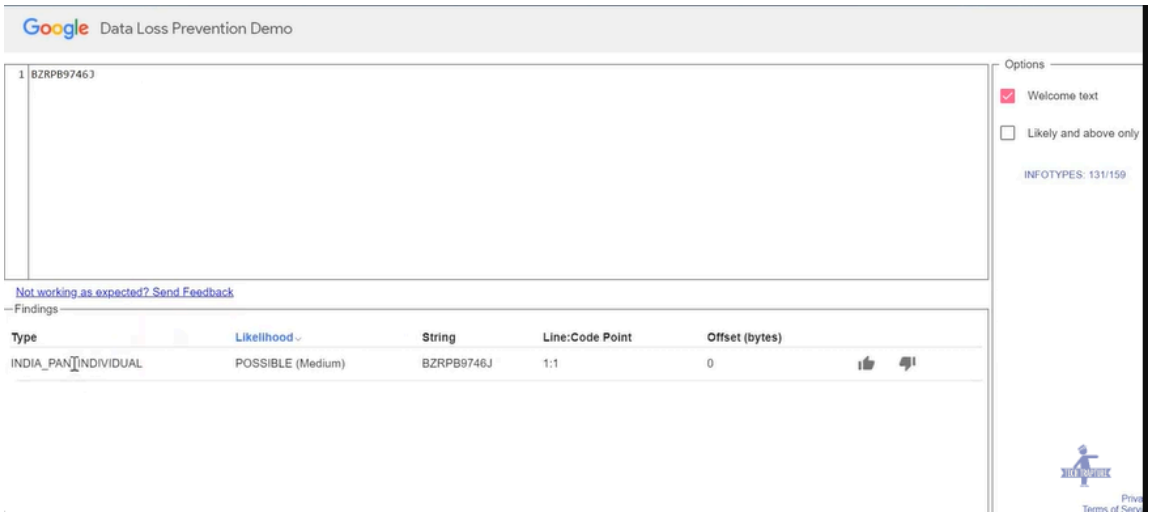


2. It identifies the PII data ..such as SSN Number, Phone number, Aadhar card number etc…and de-identifies them using

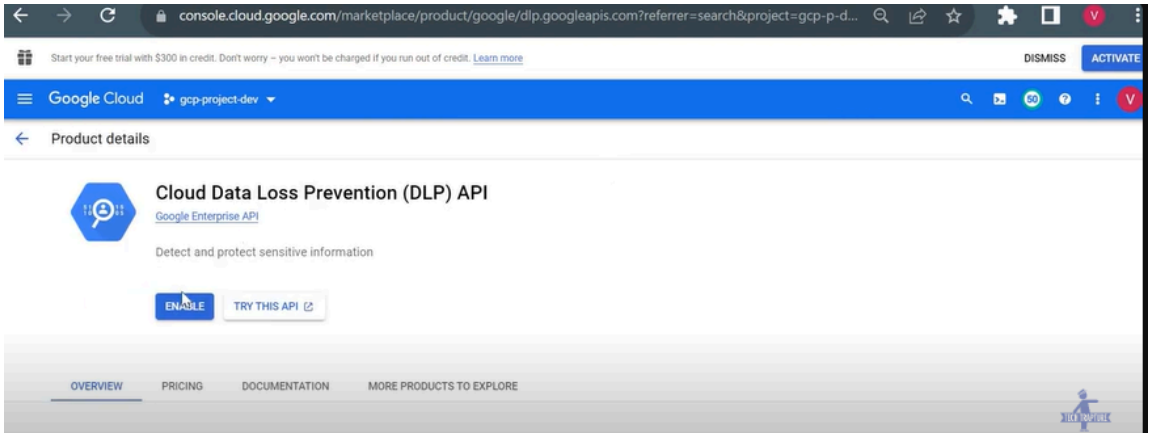


3. Demo

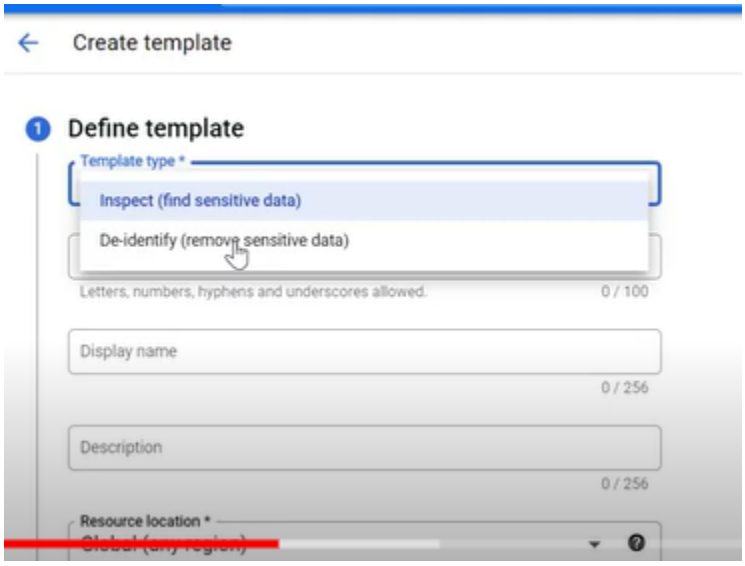
4. GCP has provided a demo..where can type something and i identifies whether it is PII data or not



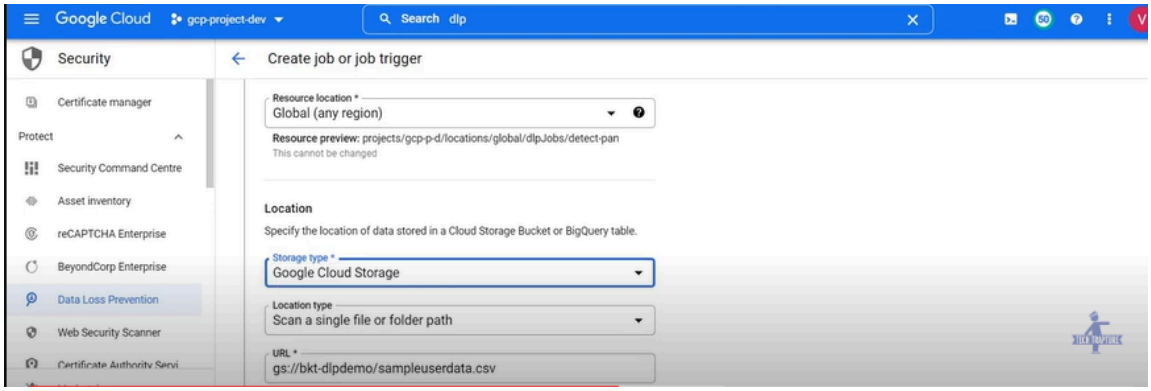
5. Practical
6. First we have to enable this API



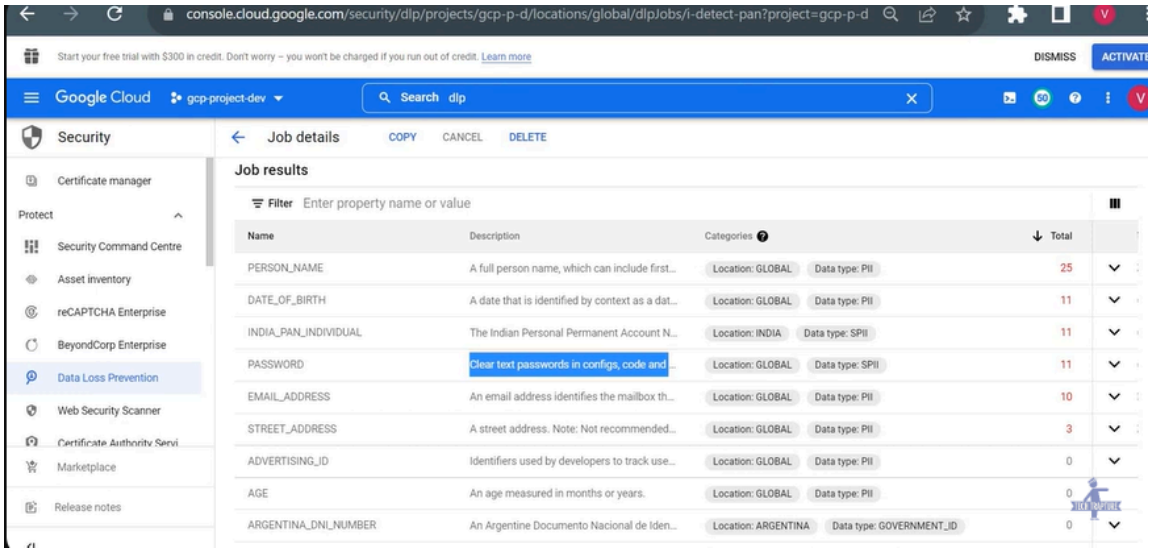
7. First we'll identify the sensitive data in table or file..so we create a template for inspect



8. Then we can configure detection of sensitive data(what to consider as sensitive data)
9. We can also add exclusion rules as well..(to exclude particular names or entity as sensitive data)
10. And we create this template
11. Later we create new bucket in GCS and add a csv file…after adding the file…we retrieve the full path of file
12. Next we go to DLP and we create inspect job…inside that we give location of our file(path)

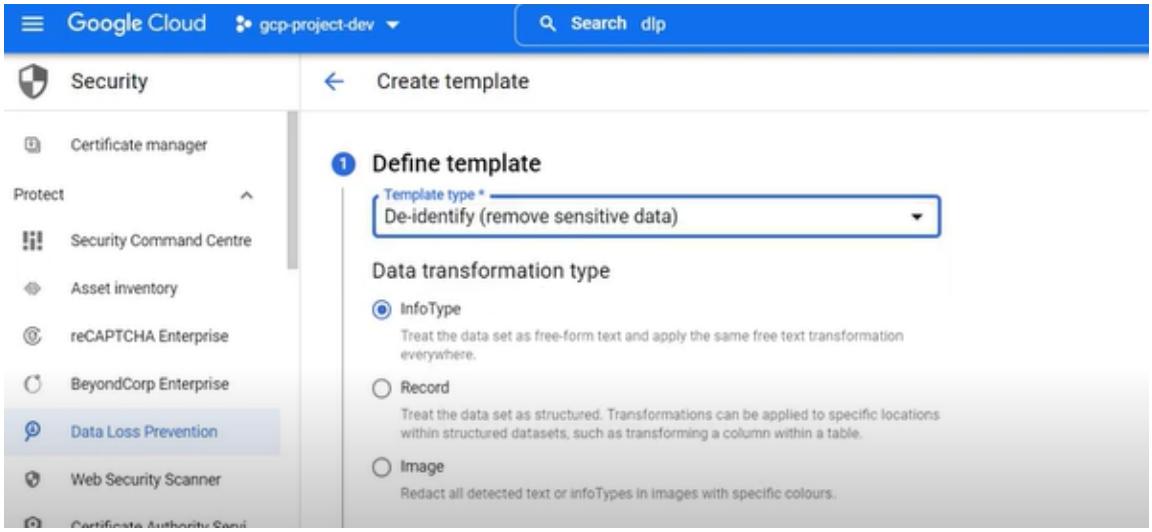


13. Then we can also give sampling
14. Now in the add actions..we'll save this results to bigquery by creating a dataset in bigquery
15. Here are the inspection results

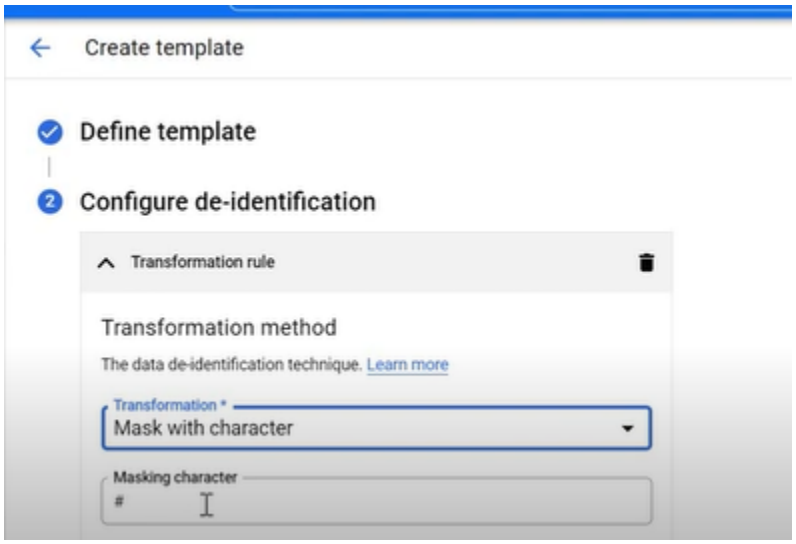


16. And we can see this results inside a table in Bigquery

17. Next we'll create a template to deidentify



18. Here for PII data it masks with '#'



Tokenize/Redact sensitive data using DLP API and DataFlow

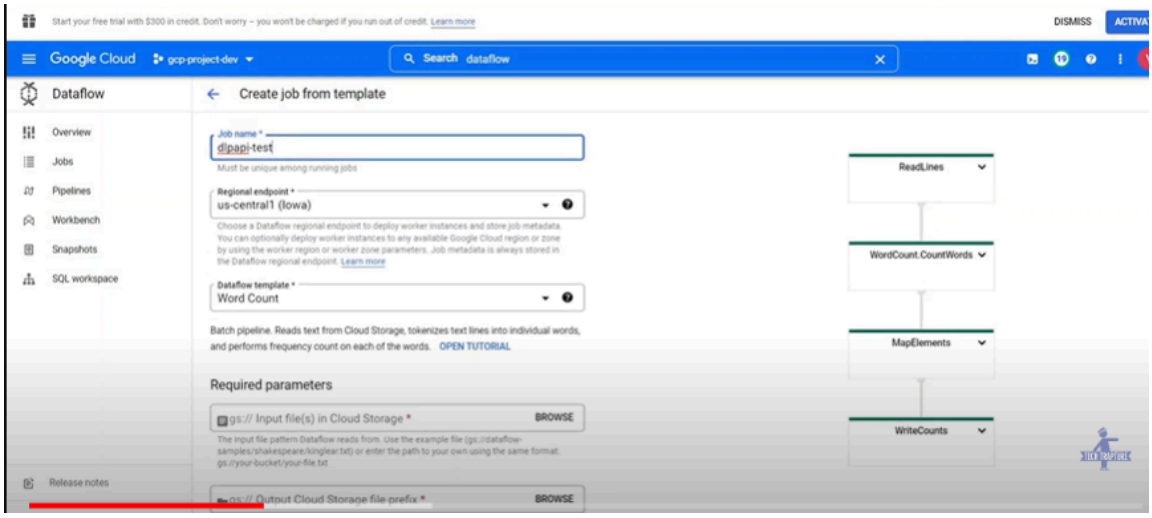
1. Here first step would be to create Inspect template

2. 

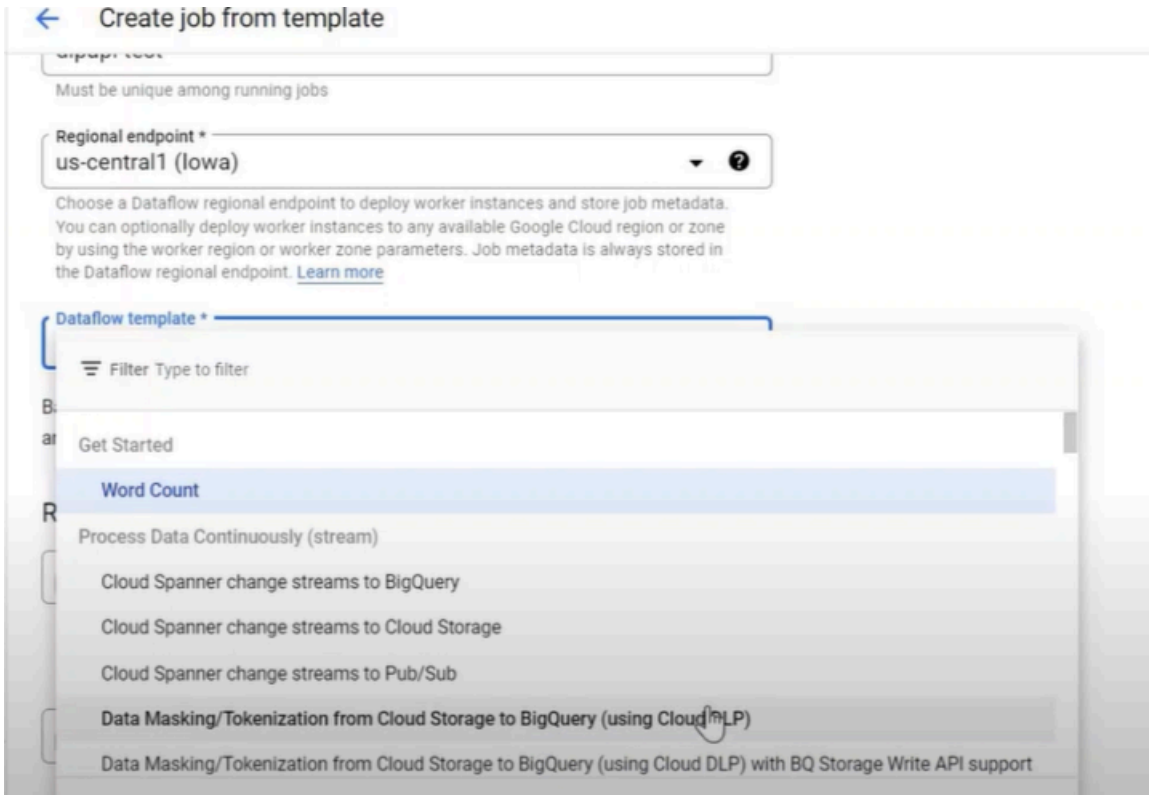
3. This is our architecture


4. Practical

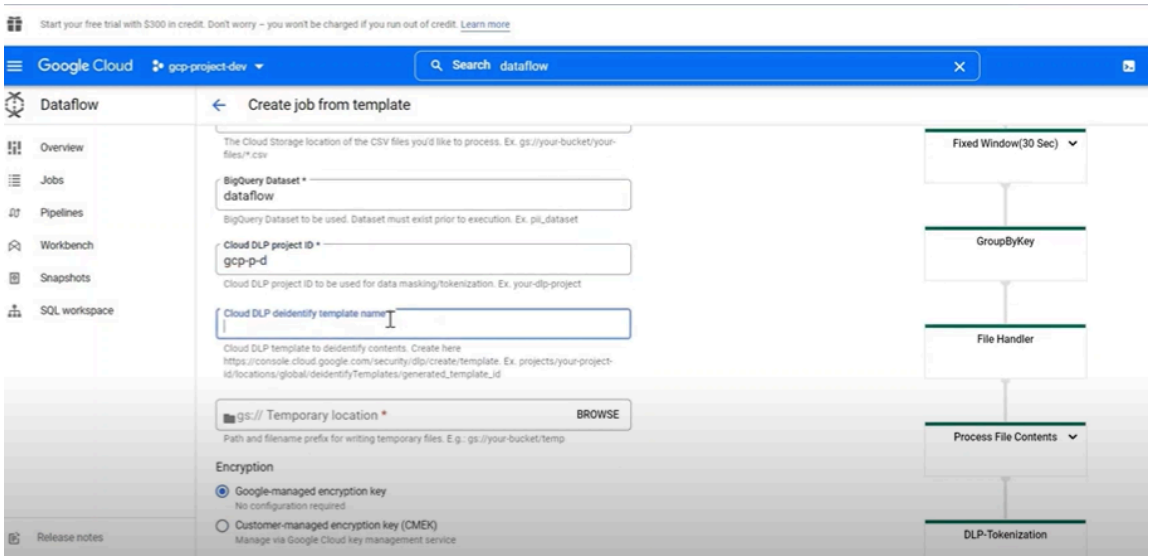
5. Now inside the dataflow we create a job



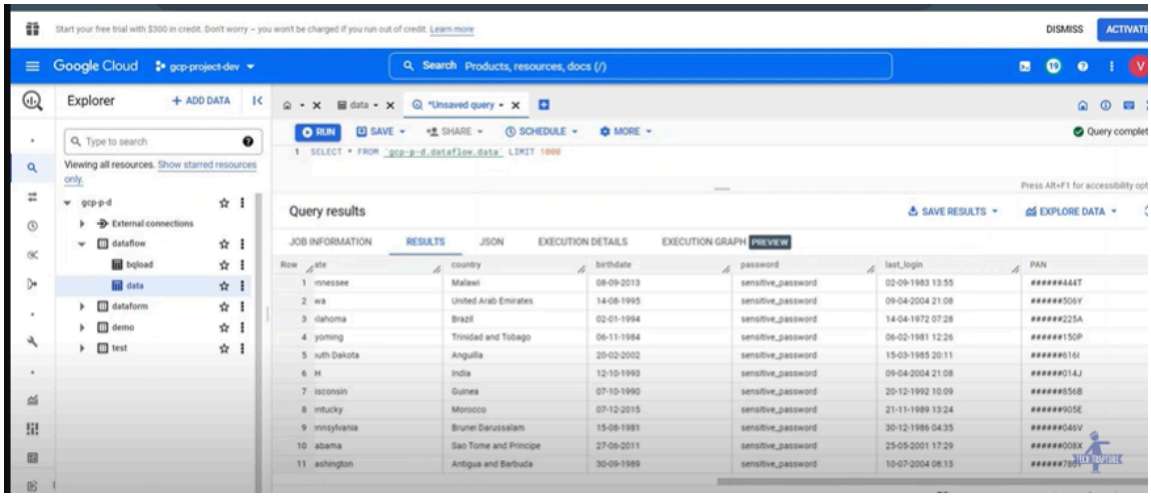
6. Here we have to choose this template



7. Next we have to gove de identification template too



8. So in our DLP we have inspect and de-identify templates..so we copy url of deidentify template and paste it above
9. After that we will give our destination as bigquery
10. Later we upload a csv file to GCS and run our job
11. Here in our big query…we can see our data is masked



12.