Using Flows to Transform Data

A Dataprep flow is an object created to organize and manage the datasets and operations that are involved in data cleaning and transformation process:

1. We begin by creating a flow by clicking the 'Create Flow' button in the top-right corner of the Dataprep dashboard (see Figure 39-4). Enter the user-defined flow name and click 'Create' as shown in Figure 39-5. The Flow page is shown in Figure 39-6.

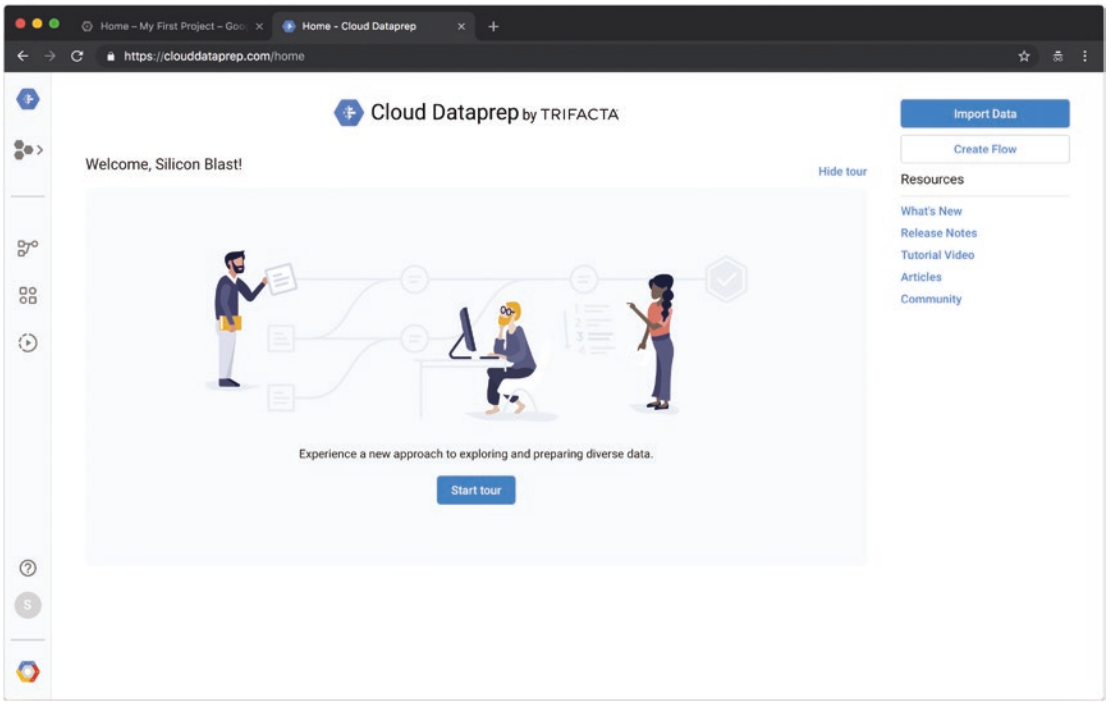

Figure 39-4. *Dataprep dashboard*

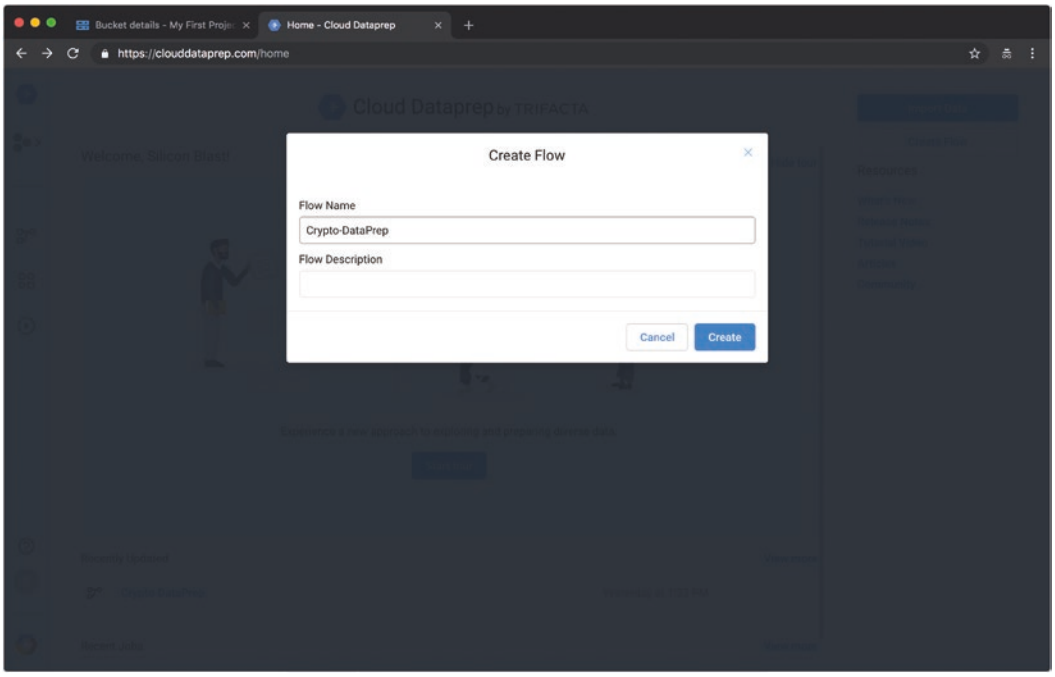

Figure 39-5. *Create Flow*

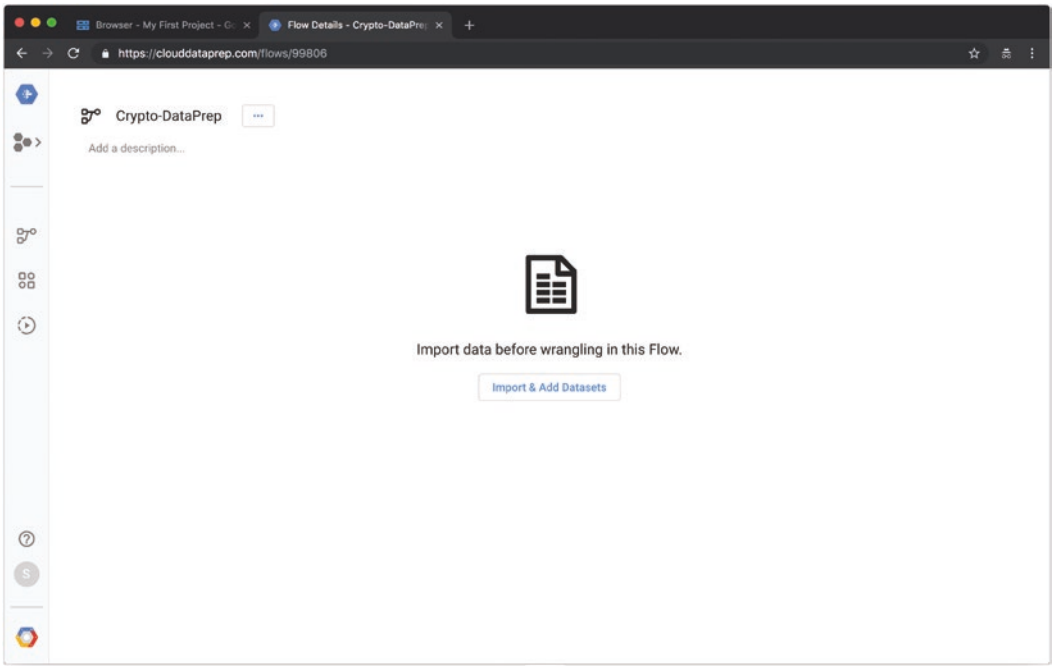

Figure 39-6. *Flow page*

2. Let's start by placing our dataset in a GCS bucket. We'll do so by running the following commands on the terminal.

   Create a new bucket.

   ```
   gsutil mb gs://my-dataprep-data
   ```

3. Transfer data from GitHub to the bucket.

   ```
   gsutil cp crypto-markets.csv gs://my-dataprep-data
   ```

4. Next, we'll transfer our 'crypto-market' dataset from the 'my-dataprep-data' bucket to the Dataprep staging bucket. We can quickly do this by executing the following code on the terminal.

   ```
   gsutil cp -r gs://my-dataprep-data gs://dataprep-staging-7fc4d500-
   8b76-48a1-9562-83675643ca4b

   Copying gs://my-dataprep-data/crypto-markets.csv [Content-
   Type=application/octet-stream]...
   / [1 files][ 47.0 MiB/ 47.0 MiB]
   Operation completed over 1 objects/47.0 MiB.
   ```

5. Next, we'll import and add Datasets to the Flow. Datasets can be uploaded directly to Dataprep which will then be stored to the bucket Dataprep generated on start-up. Also, Dataprep can import datasets already stored in BigQuery or GCS. In this case, we will import the 'crypto-market' dataset that we earlier transferred to the Dataprep staging bucket which is in the folder 'my-dataprep-data' (see Figure 39-7). Figure 39-8 shows the dataset loading into Dataprep.

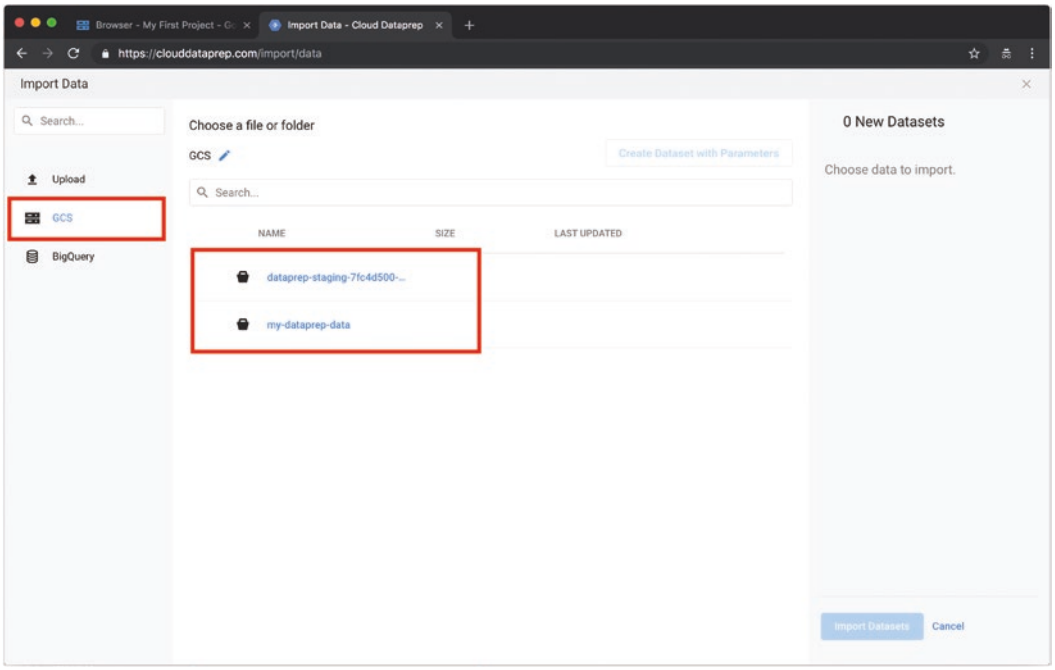

Figure 39-7. *Import Dataset from GCS to Dataprep*

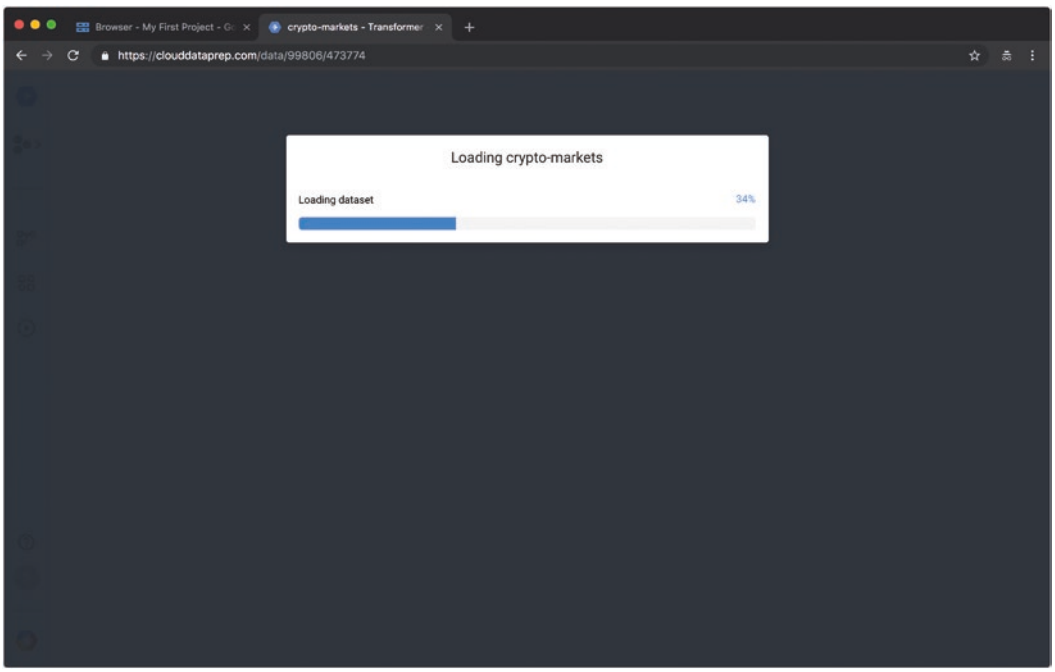

Figure 39-8. *Loading Dataset to Dataprep*

6. Next, we'll create a recipe. A Dataprep recipe contains the transformation steps taken to clean and process a Dataset. This recipe is later executed as a Dataflow job to operate on the Dataset and come up with results. Click the 'Add New Recipe' button to create a recipe. The recipe is in the bounded red box in Figure 39-9.

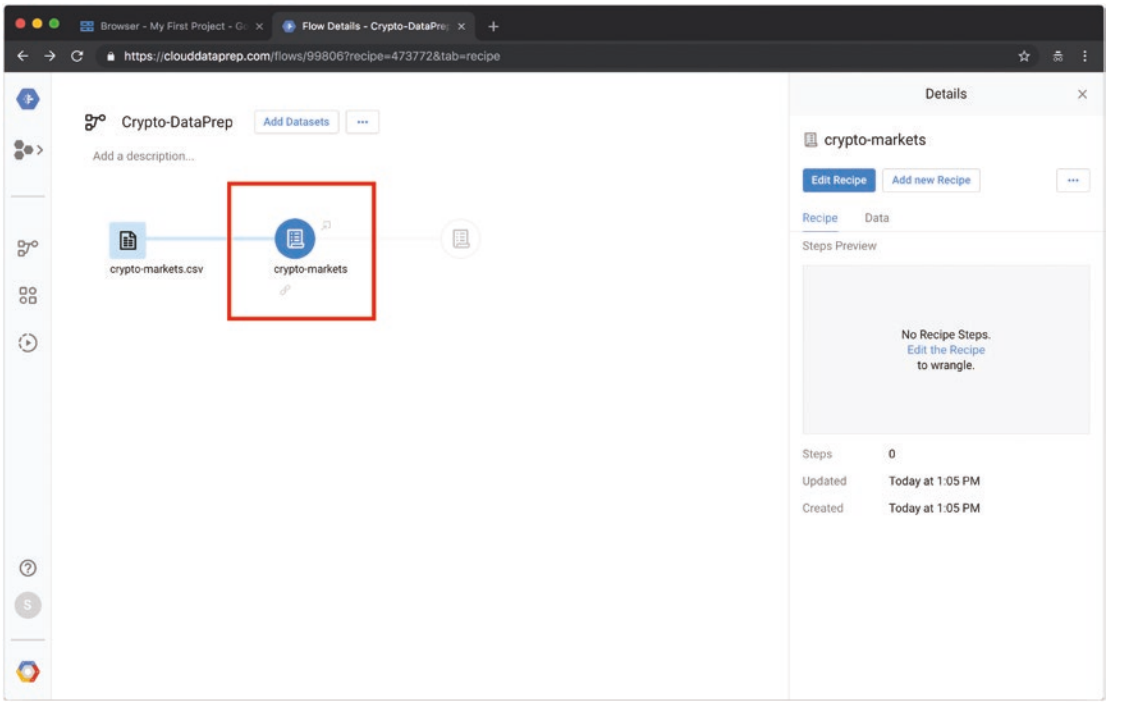

Figure 39-9. *Dataset recipe*

7. Then click the 'Edit Recipe' button to open the 'Transformation Grid' where we carry out various cleaning and processing steps on the Dataset.

8. For the example in this section, we'll carry out a simple transformation process by dropping some unused columns and then removing all rows in the dataset except those for Bitcoin crypto-currency:

a. Remove the ‘slug’ column. Click ‘Add’ within the red box to drop the column (see Figure 39-10).

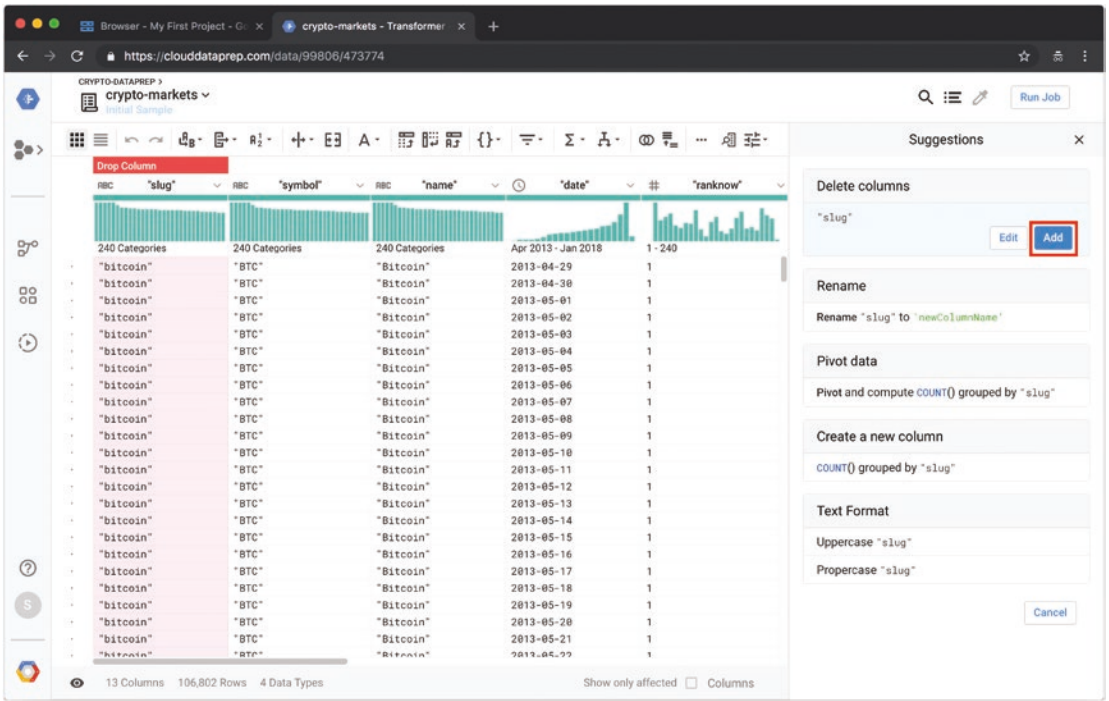

Figure 39-10. *Remove ‘slug’ column*

b. Remove the ‘name’ column (see Figure 39-11).

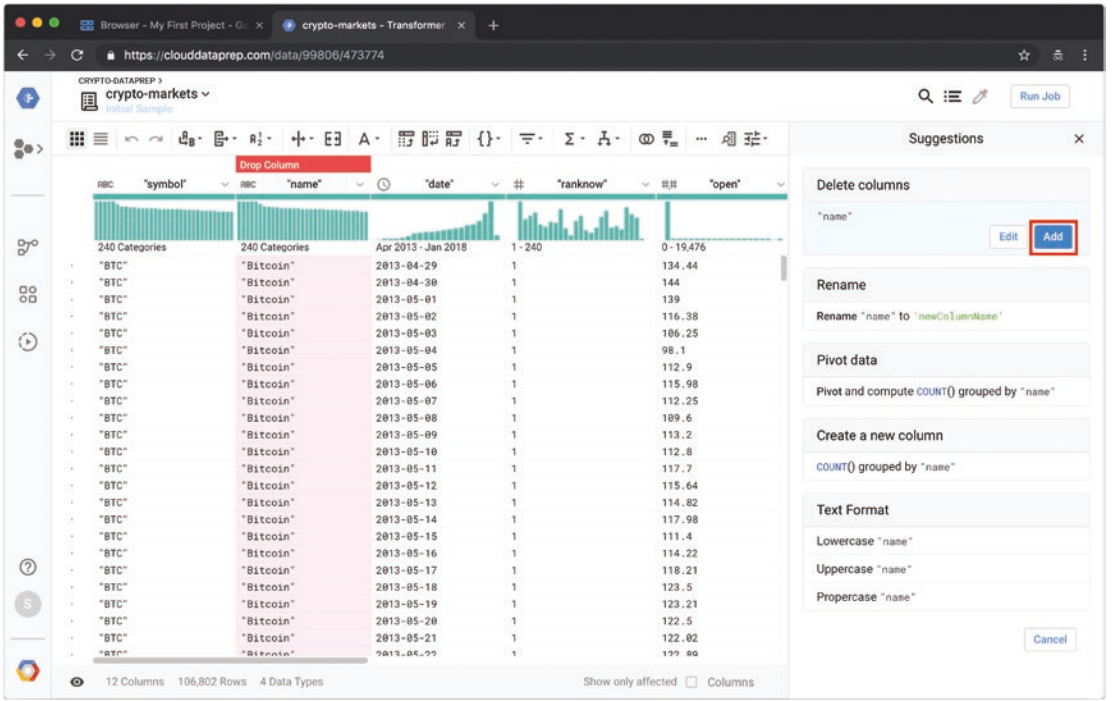

Figure 39-11. *Remove 'name' column*

c. Next, we'll filter the rows in the dataset to retain only the Bitcoin records (see Figures 39-12 and 39-13).

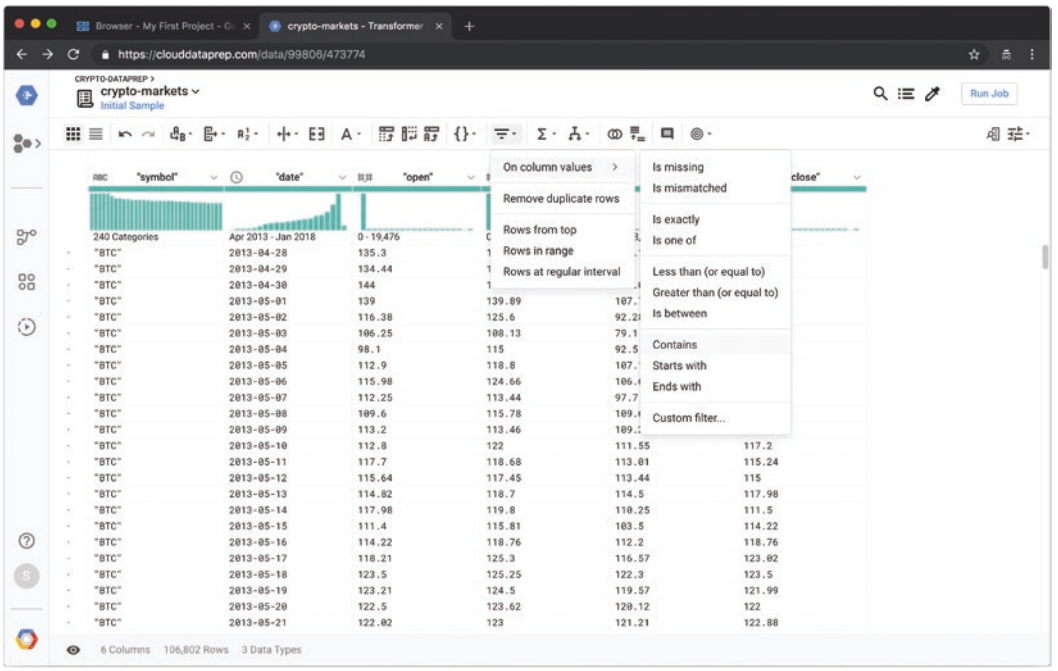

Figure 39-12. *Filter rows using Dataprep*

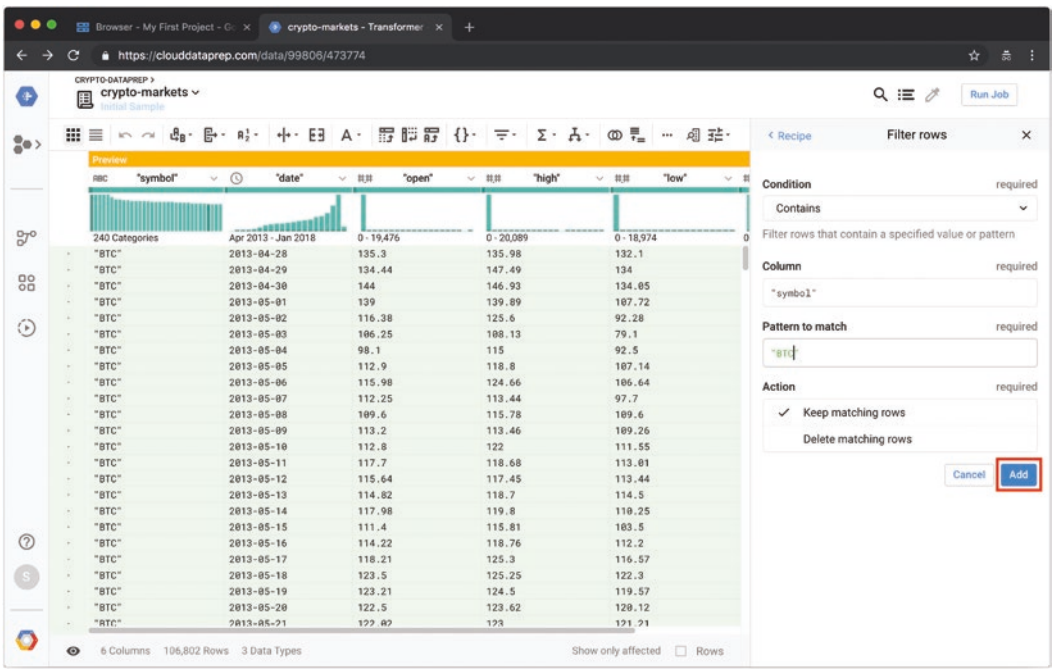

Figure 39-13. *Remove all rows except the Bitcoin records*

9. Figure 39-14 shows the dataset transformation recipes. Click 'Run Job' in Figure 39-14 and also in Figure 39-15 to run the job on Cloud Dataflow.

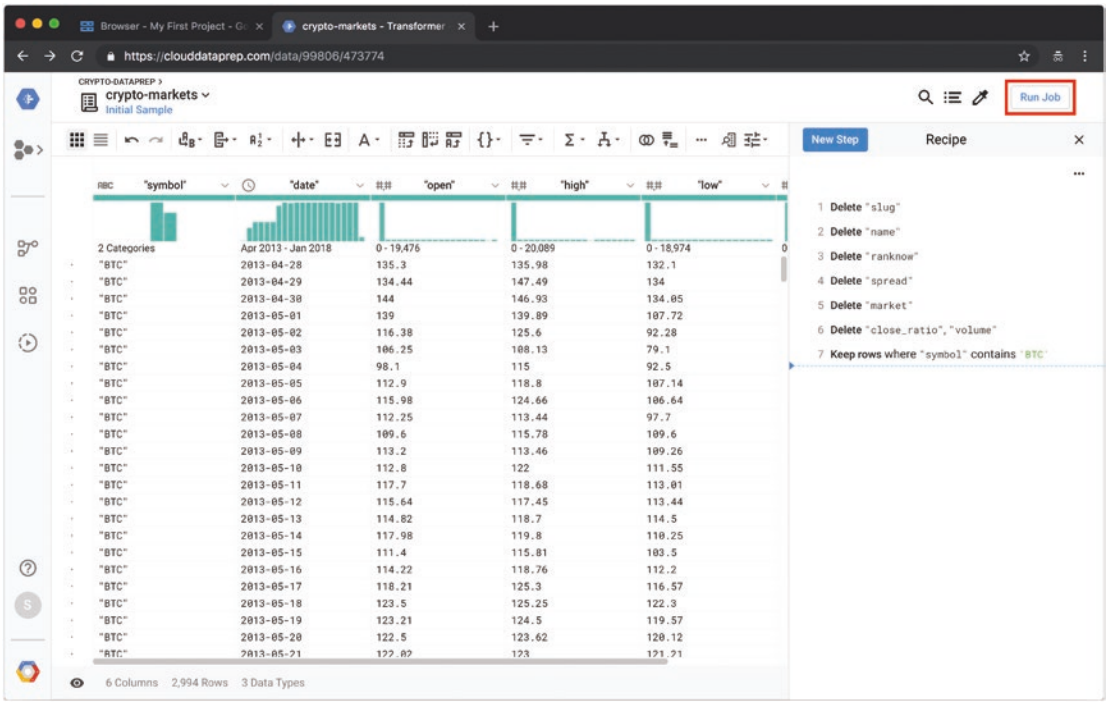

Figure 39-14. *View transformation recipes*

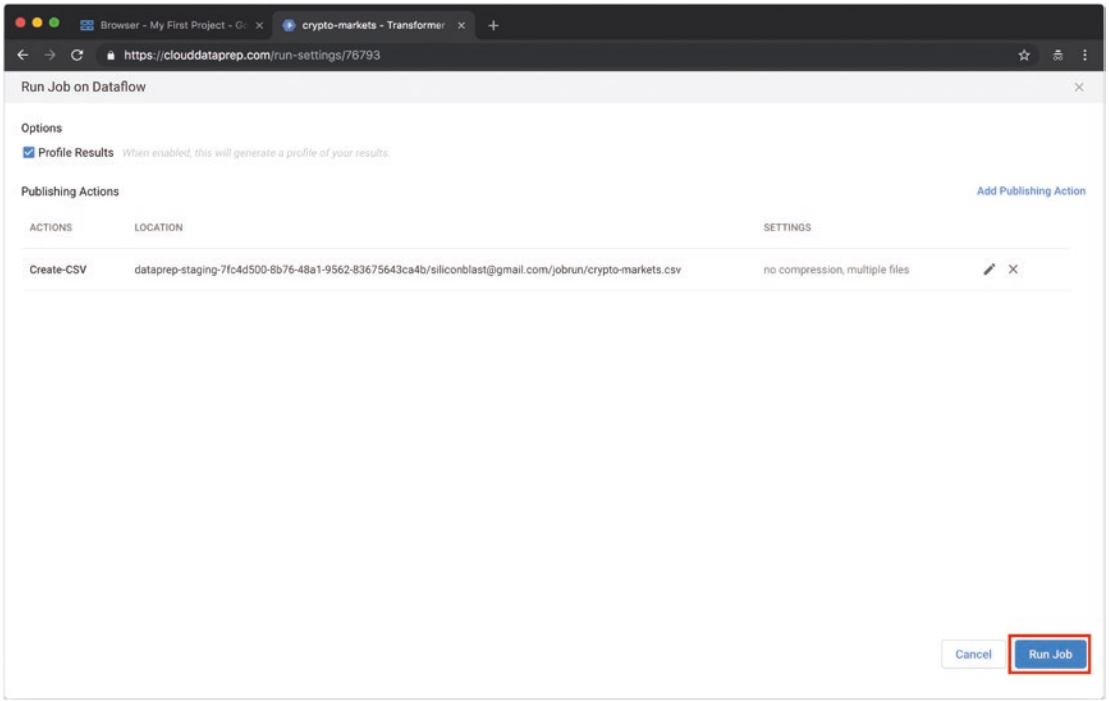

Figure 39-15. *Run Job on Dataflow*

10. Figure 39-16 shows the running job, and Figure 39-17 shows the completed job after some minutes.

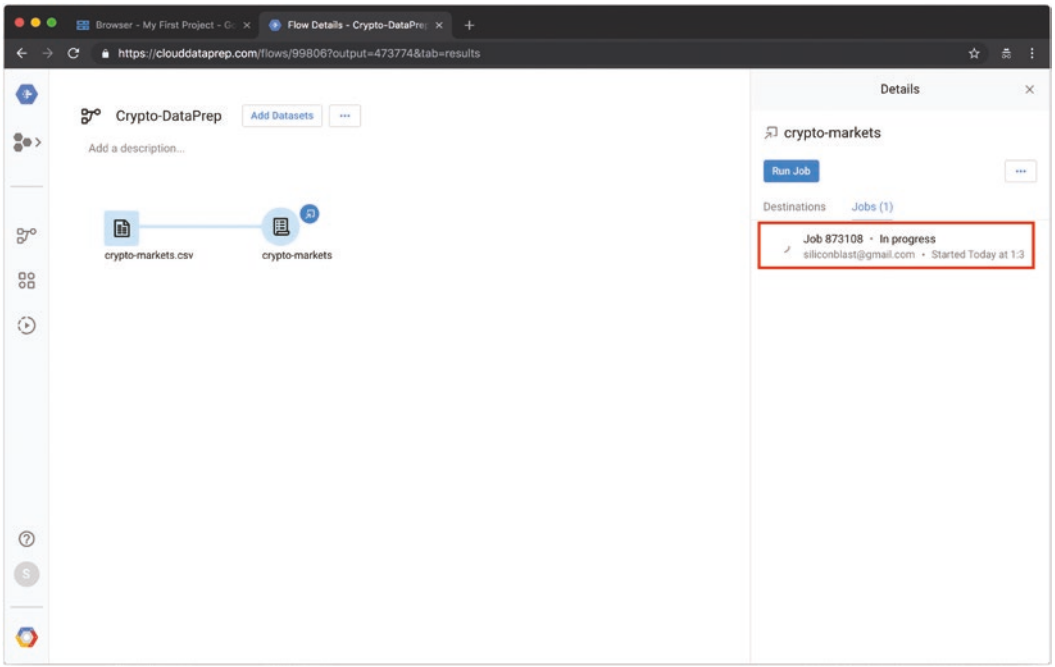

Figure 39-16. *Job running on Dataflow*

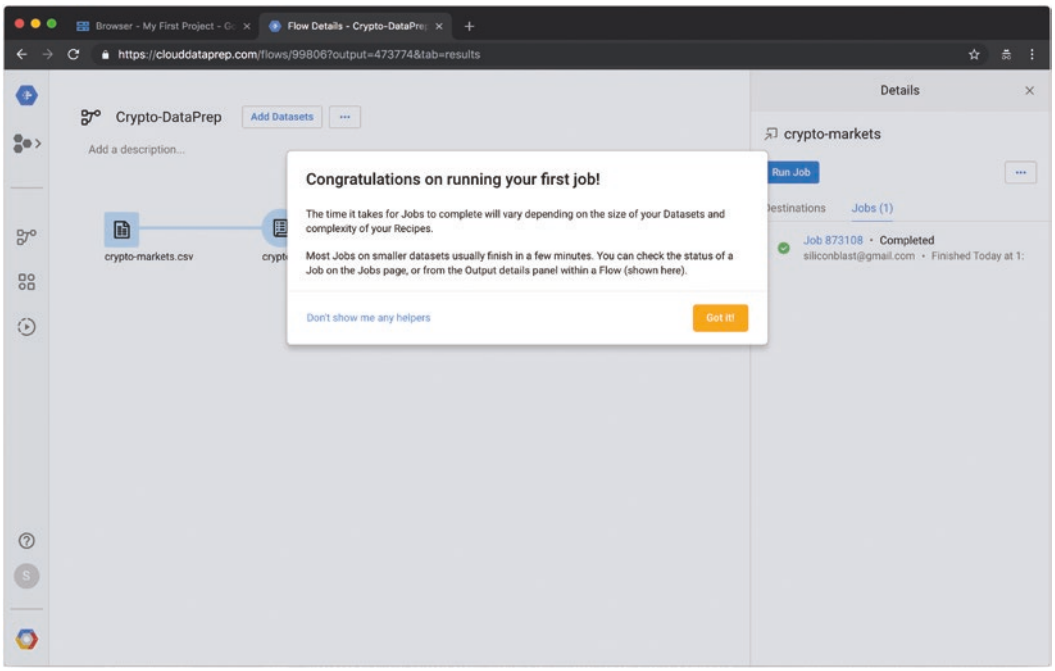

Figure 39-17. *Completed job*

11. View the results of the job (see Figure 39-18).

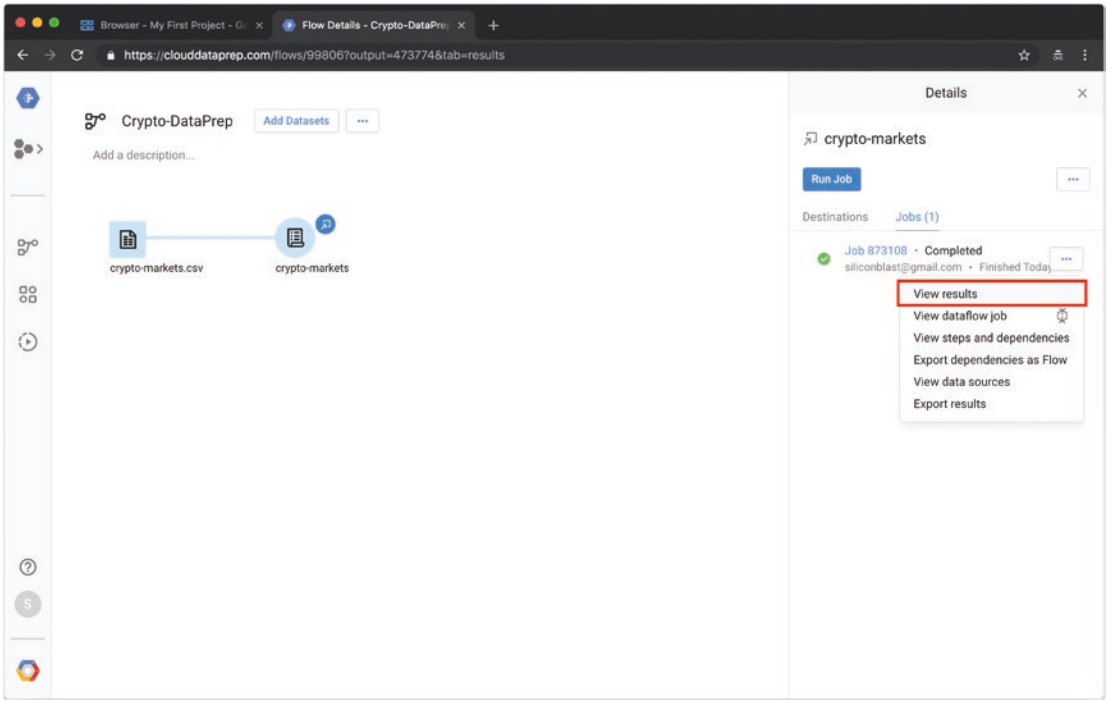

Figure 39-18. *View job result*

12. From the Results page shown in Figure 39-19, we can export the results back to GCS (see Figure 39-20).

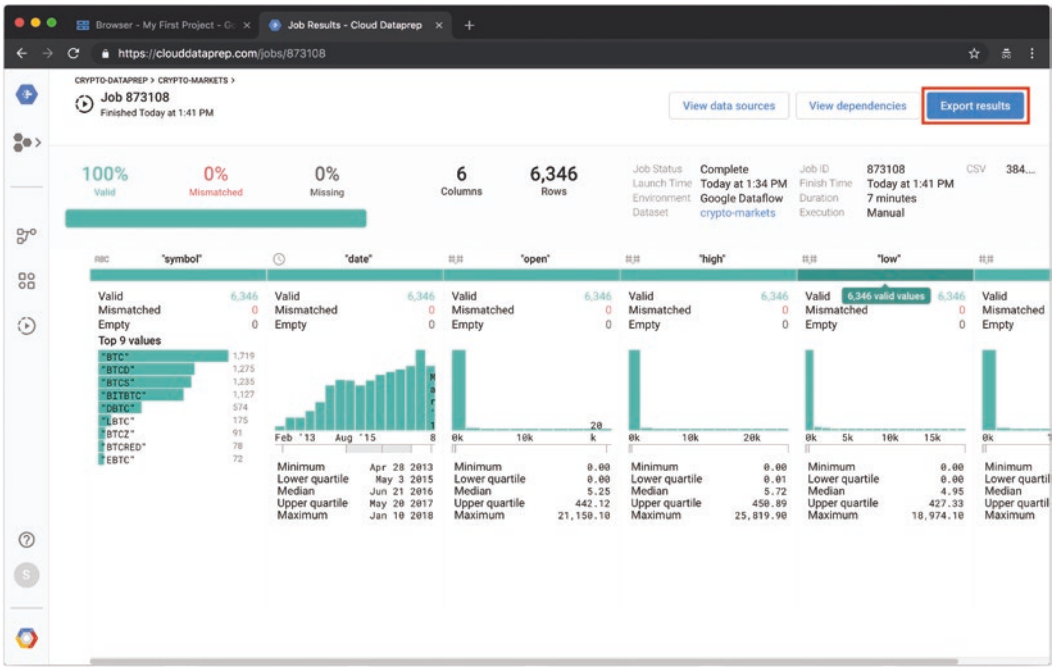

Figure 39-19. *Job Results page*

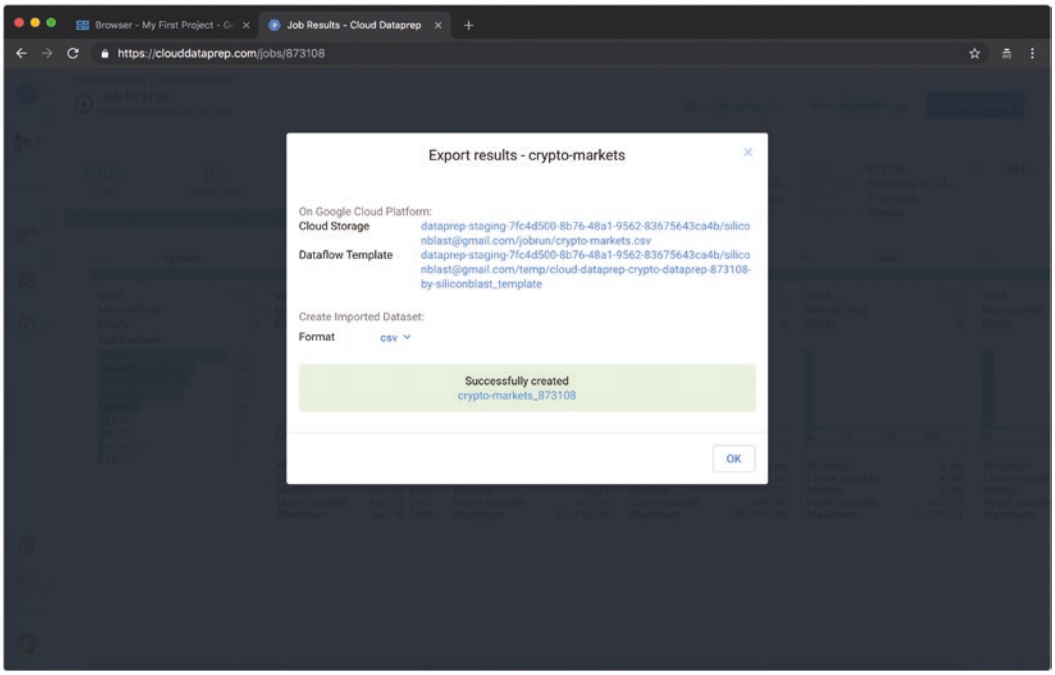

Figure 39-20. *Export completed jobs*

This chapter provides an example overview of working with Dataprep to visually explore and transform large datasets on GCP by using the Google Cloud Dataflow infrastructure for distributed processing. In the next chapter, we will introduce working with Cloud Dataflow for building custom data transformation pipelines.

CHAPTER 40

Google Cloud Dataflow

Google Cloud Dataflow provides a serverless, parallel, and distributed infrastructure for running jobs for batch and stream data processing. One of the core strengths of Dataflow is its ability to almost seamlessly handle the switch from processing of batch historical data to streaming datasets while elegantly taking into consideration the perks of streaming processing such as windowing. Dataflow is a major component of the data/ML pipeline on GCP. Typically, Dataflow is used to transform humongous datasets from a variety of sources such as Cloud Pub/Sub or Apache Kafka to a sink such as BigQuery or Google Cloud Storage.

Critical to Dataflow is the use of the Apache Beam programming model for building the parallel data processing pipelines for batch and stream operations. The data processing pipelines built with the Beam SDKs can be executed on various processing backends such as Apache Apex, Apache Spark, Apache Flink, and of course Google Cloud Dataflow. In this section, we will build data transformation pipelines using the Beam Python SDK. As of this time of writing, Beam also supports building data pipelines using Java, Go, and Scala languages.

Beam Programming

Apache Beam provides a set of broad concepts to simplify the process of building a transformation pipeline for distributed batch and stream jobs. We'll go through these concepts providing simple code samples:

- A Pipeline: A Pipeline object wraps the entire operation and prescribes the transformation process by defining the input data source to the pipeline, how that data will be transformed, and where the data will be written. Also, the Pipeline object indicates the distributed processing backend to execute on. Indeed, a Pipeline

E. Bisong, *Building Machine Learning and Deep Learning Models on Google Cloud Platform*,
https://doi.org/10.1007/978-1-4842-4470-8_40