MCDA 5560 BUSSINESS INTELLIGENCE AND DATA VISUALIZATION

Assignment 1 - Data Preparation

Team Members

Names	A#
Raghav Sharma	A00445168
Timur	A00439340
Kuzhippallil, Maria A.	A00442283

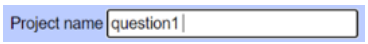
Contents

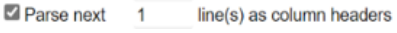
Data Preparation using OpenRefine

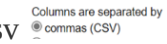
1) Install OpenRefine as shown in the tutorial

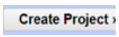
2) Create Project question1 in OpenRefine

Steps –

- Creating project by loading the question1.csv file *(question 3)*.
- Provide name for project by 

- Check parse headers option to consider first row as header 

- Columns are separated by csv 

- Create project by clicking on 

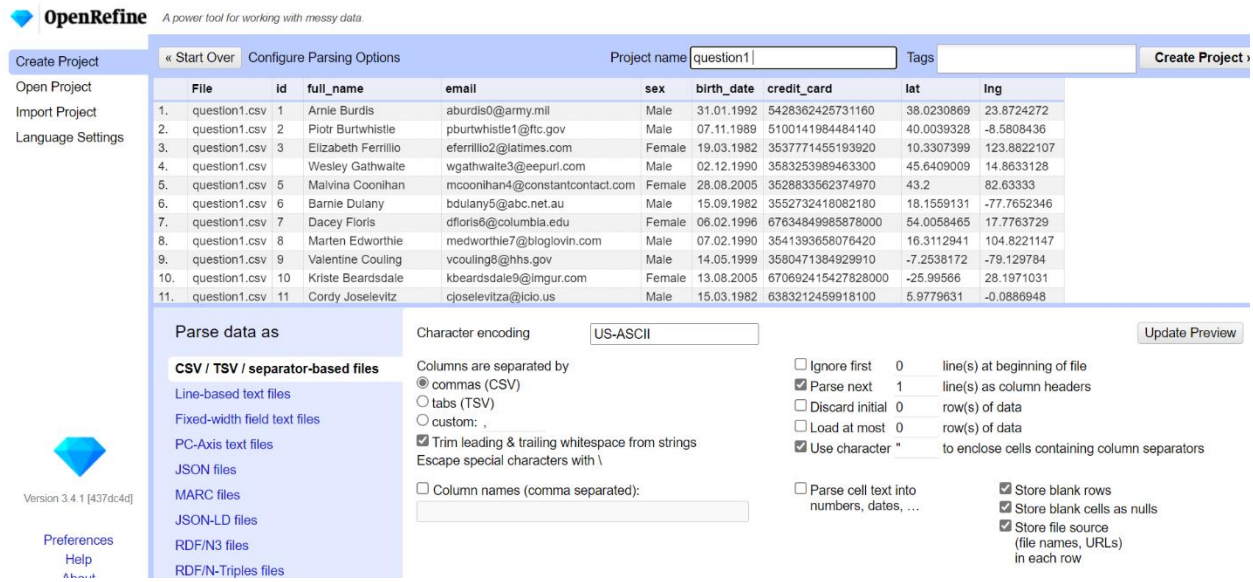
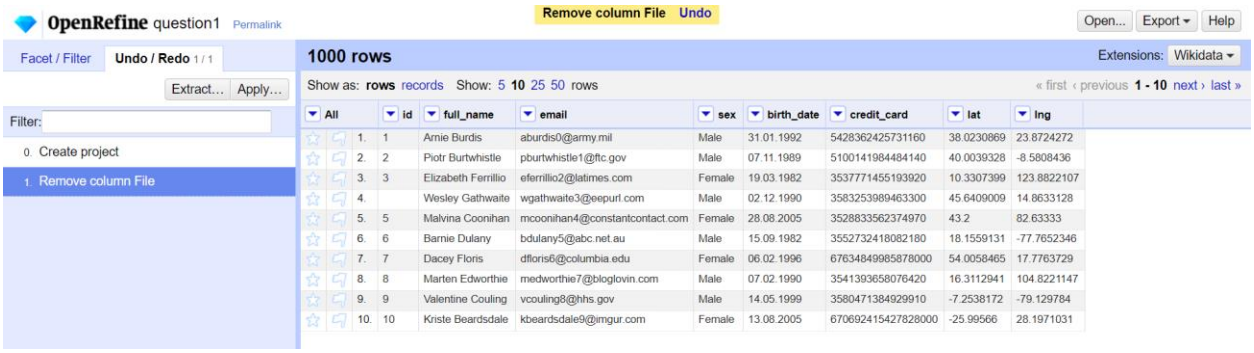


Fig – Create question1 project

Fig – Create question1 project



All	id	full_name	email	sex	birth_date	credit_card	lat	lng
1.	1	Arnie Burdis	aburdis0@army.mil	Male	31.01.1992	5428362425731160	38.0230869	23.8724272
2.	2	Piotr Burtwhistle	pburtwhistle1@ftc.gov	Male	07.11.1989	5100141984484140	40.0039328	-8.5808436
3.	3	Elizabeth Ferrillio	eferrillio2@latimes.com	Female	19.03.1982	3537771455193920	10.3307399	123.8822107
4.		Wesley Gathwaite	wgathwaite3@eepurl.com	Male	02.12.1990	3583253989463300	45.6409009	14.8633128
5.	5	Malvina Coonihan	mcoonihan4@constantcontact.com	Female	28.08.2005	3528833562374970	43.2	82.63333
6.	6	Barnie Dulany	bdulany5@abc.net.au	Male	15.09.1982	3552732418082180	18.1559131	-77.7652346
7.	7	Dacey Floris	dfloris6@columbia.edu	Female	06.02.1996	67634849985878000	54.0058465	17.7763729
8.	8	Marten Edworthie	medworthie7@bloglovin.com	Male	07.02.1990	3541393658076420	16.3112941	104.8221147
9.	9	Valentine Couling	vcouling8@hhs.gov	Male	14.05.1999	3580471384929910	-7.2538172	-79.129784
10.	10	Kriste Beardsdale	kbeardsdale9@imgur.com	Female	13.08.2005	670692415427828000	-25.99566	28.1971031

Fig – Deleting File column

Row count after loading data→ **1000**

3) Load question1.csv

Steps –

- Click on choose files  and load question1.csv.
- Then go ahead by clicking next

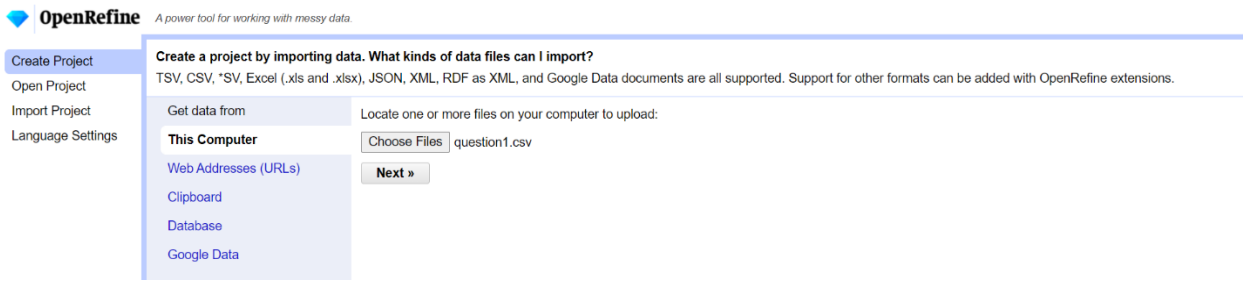
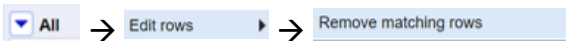


Fig – Load question1.csv

4) Remove all rows in which id is empty

Steps –

- Use drop down option next to id  .
- Choose facet, followed by customized facet then facet by blank to find empty records for id.

- Select true  which satisfies the criteria of finding the blank records.
- Remove matching records by selecting dropdown of all followed by edit rows and remove matching rows. 

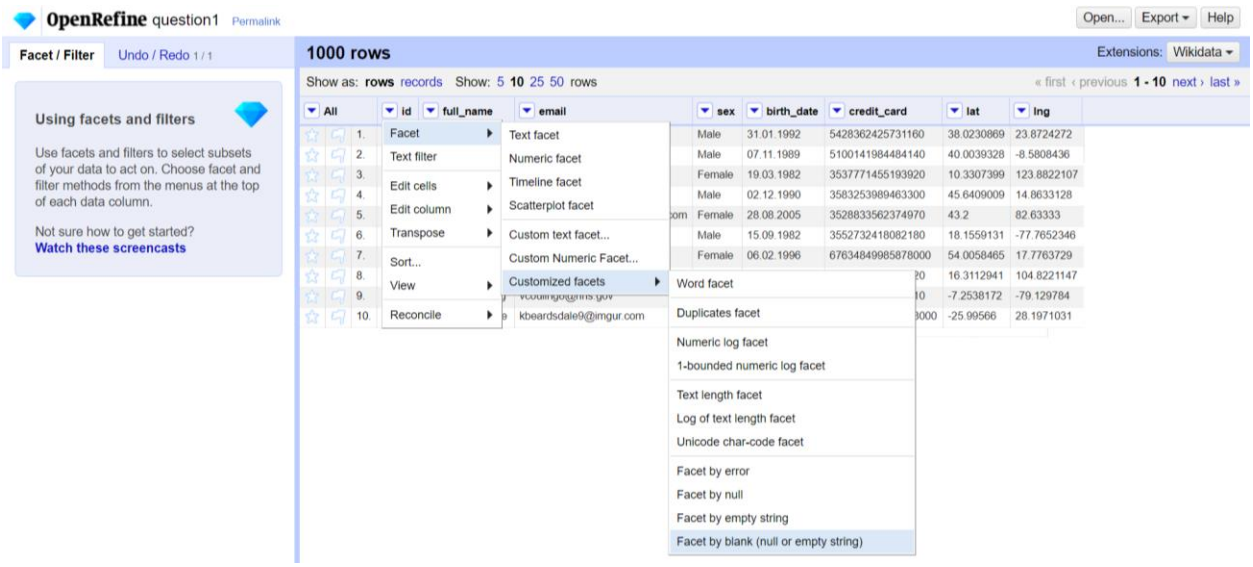


Fig – Facet by blank

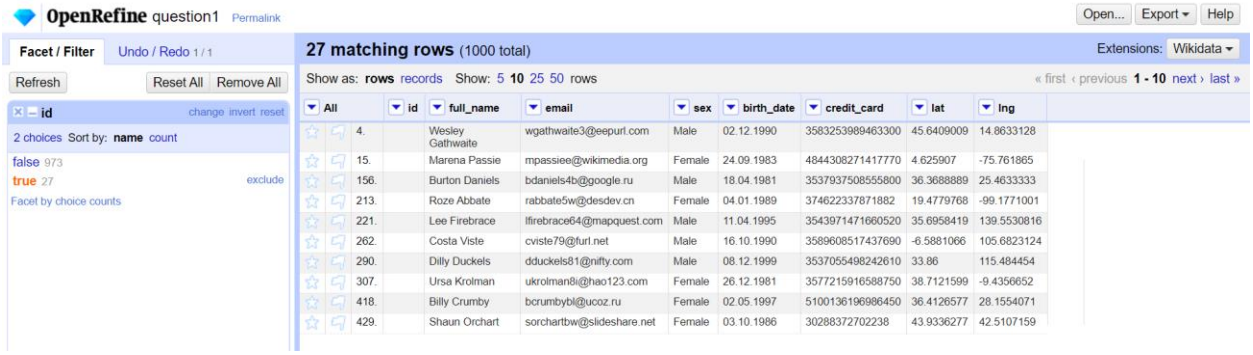


Fig – 27 records with null value for id

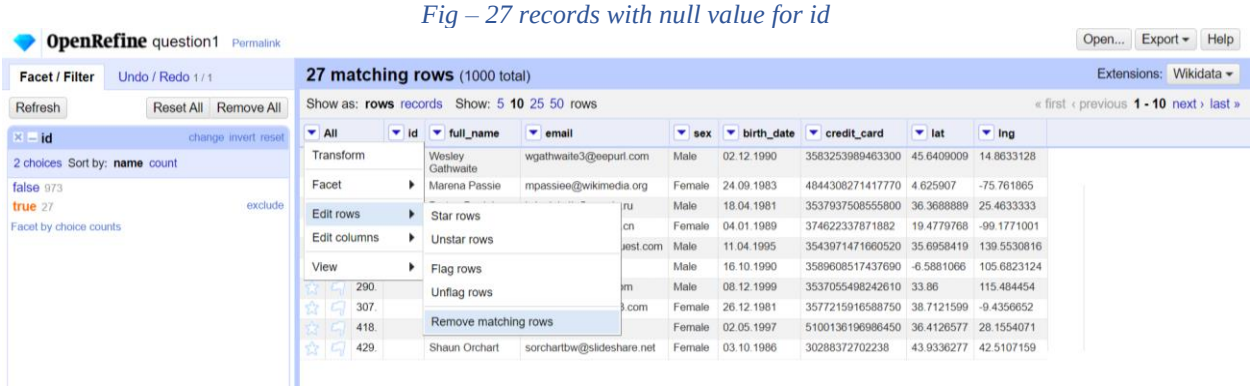


Fig – Removing id with null value

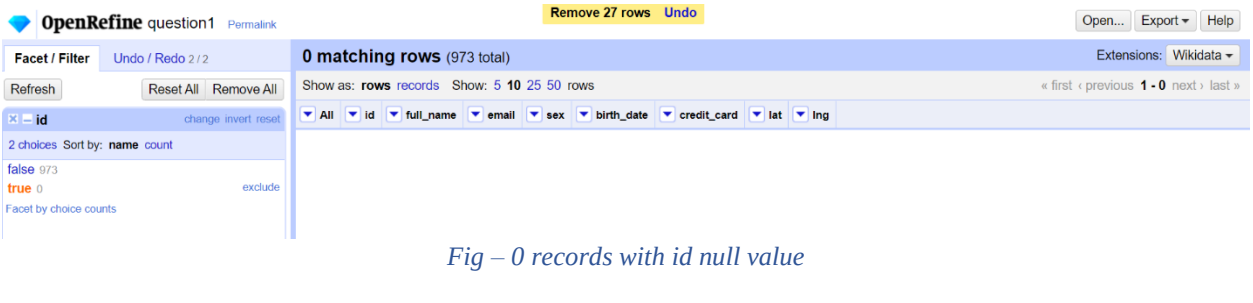


Fig – 0 records with id null value

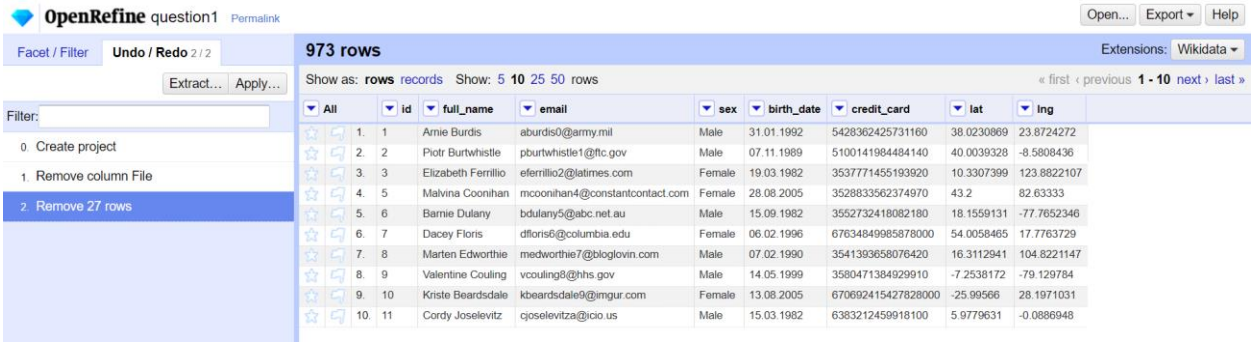
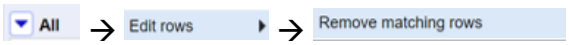


Fig – Removing id with null value

Row count after deleting id with null/blank values → **973**

5) Remove all rows in which email is empty

Steps –

- Use drop down option next to email  .
- Choose facet, followed by customized facet then facet by blank to find empty records for email. 
- Select true which satisfies the criteria of finding the blank records.
- Remove matching records by selecting dropdown of all followed by edit rows and remove matching rows. 

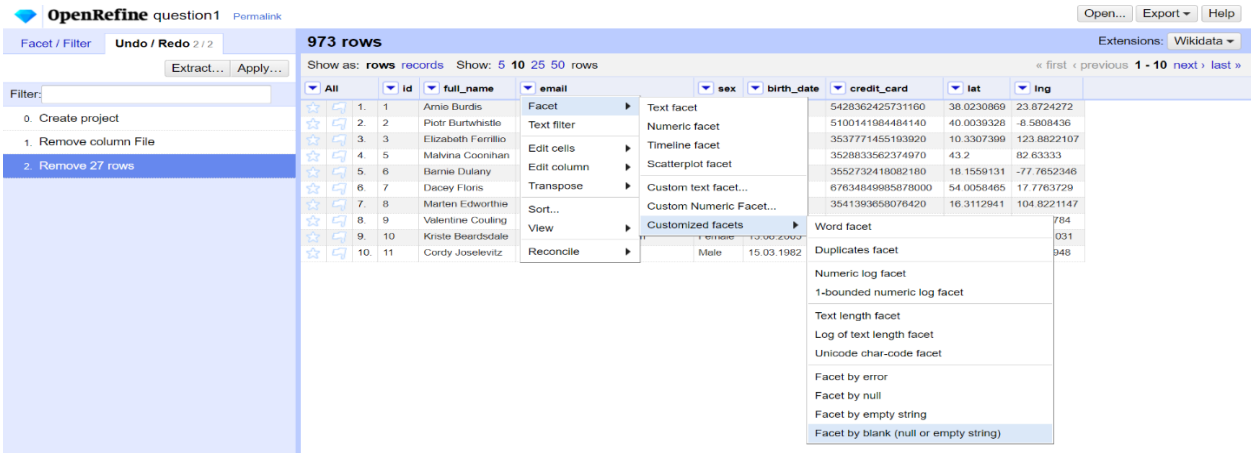


Fig – Facet by blank

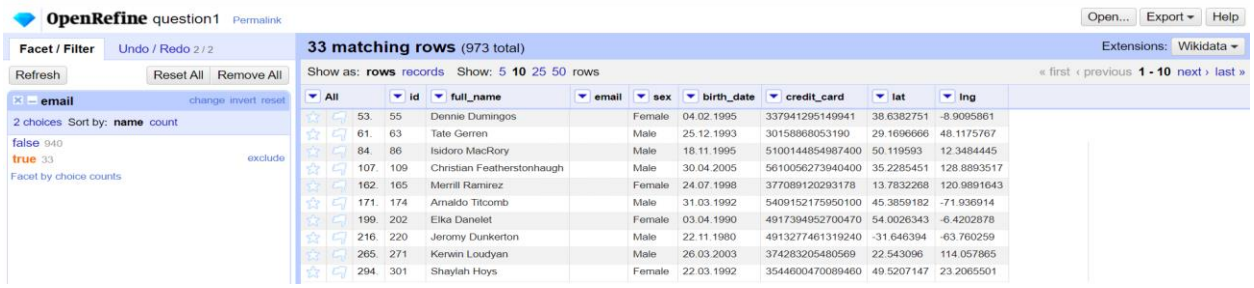


Fig – 33 blank records for email id field

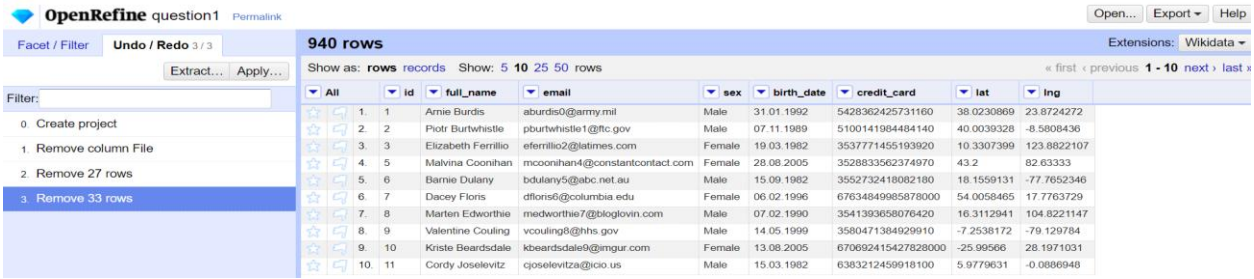


Fig – Removing email id with null values

Row count after deleting id with null/blank values → **940**

6) Split Full Name in to First name and Last Name. Store first name into f_name and last name into l_name (you will create these two columns).

Steps –

- Select split column option **Split column full_name into several columns**.
- Provide separator i.e " " and provide the number of columns to split into.
- Check remove this column ☑ Remove this column
- Rename the columns to f_name and l_name.

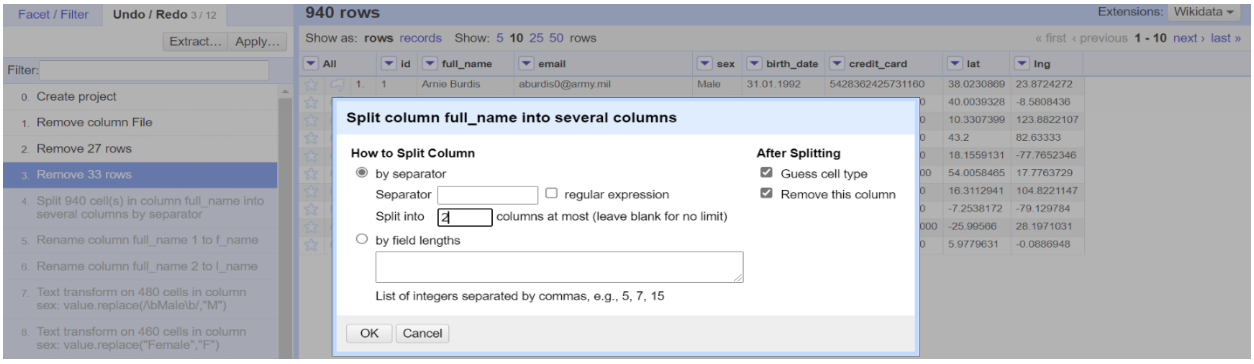


Fig – Splitting full name column with space separator

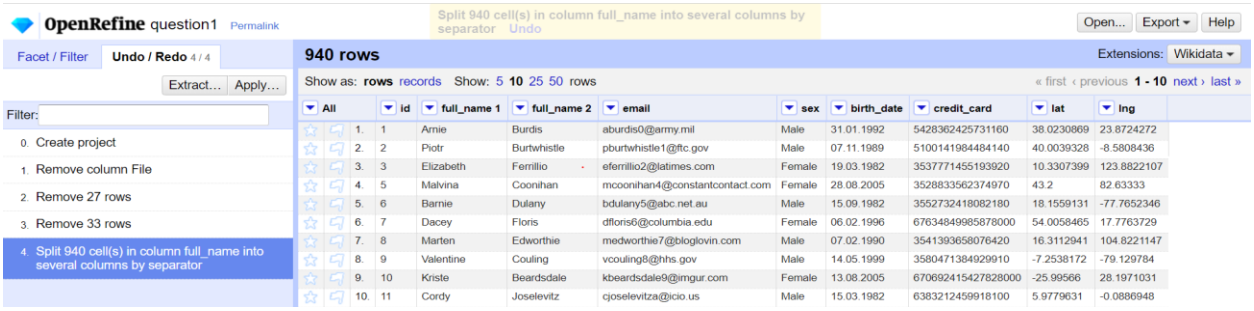


Fig – Full name column split to full_name2 and full_name2

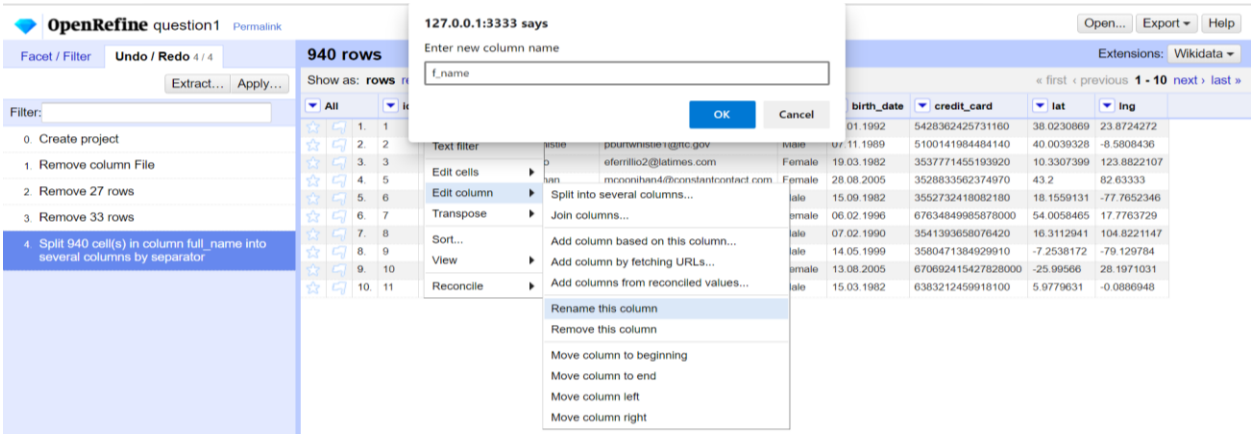


Fig – Renaming to f_name

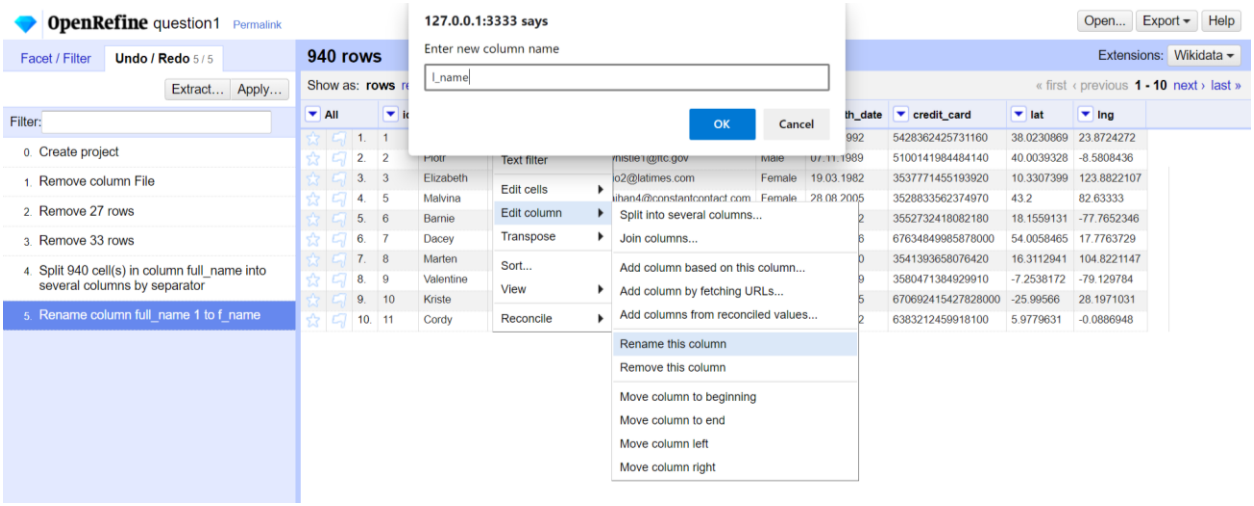


Fig – Renaming to l_name

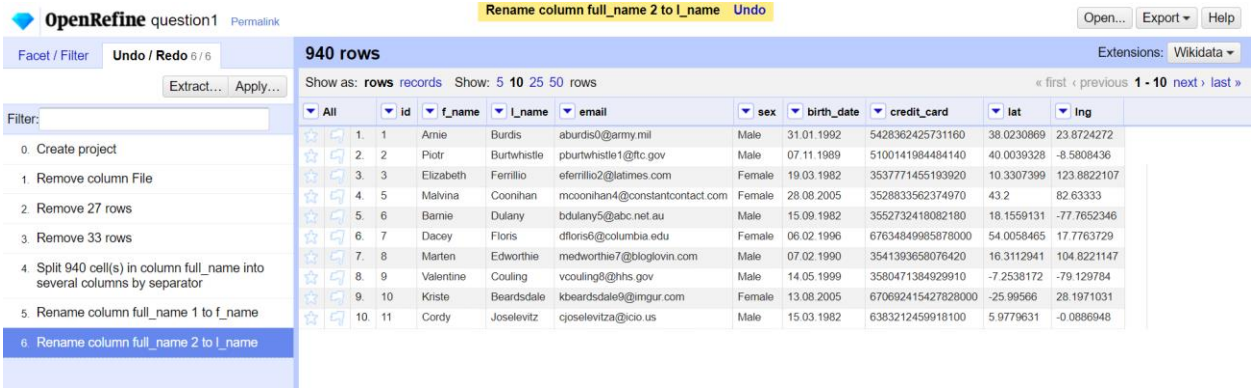


Fig – Removed full_name column and split to f_name and l_name

Row count after creating f_name and l_name columns from full_name → **940**

7) Code Gender (sex column) into M and F instead of Male and Female (Do not create a new column).

Steps –

- Choose replace Replace for sex column.
- Find the value to be replaced Find: Male and provide new value to replace with Replace with: M for both Male and Female with M and F.
- Then click ok OK to apply changes.

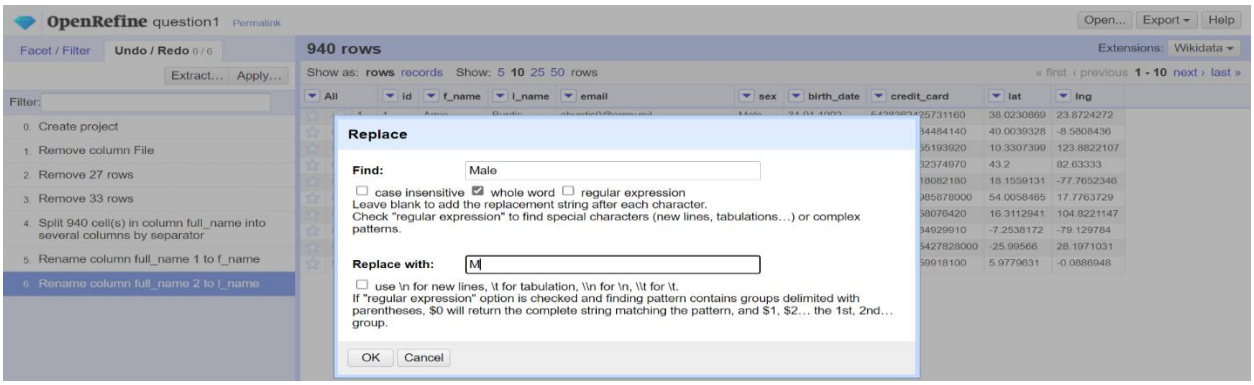


Fig – Replacing Male to M

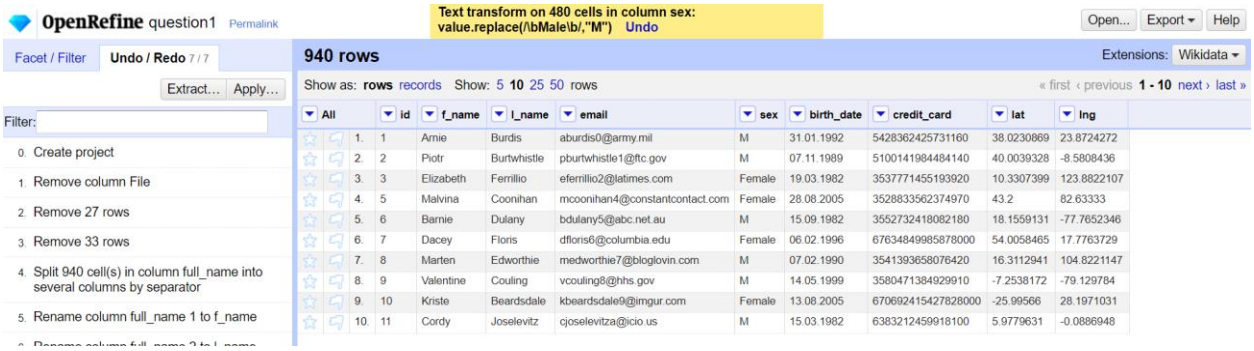


Fig – Transformed Male to M

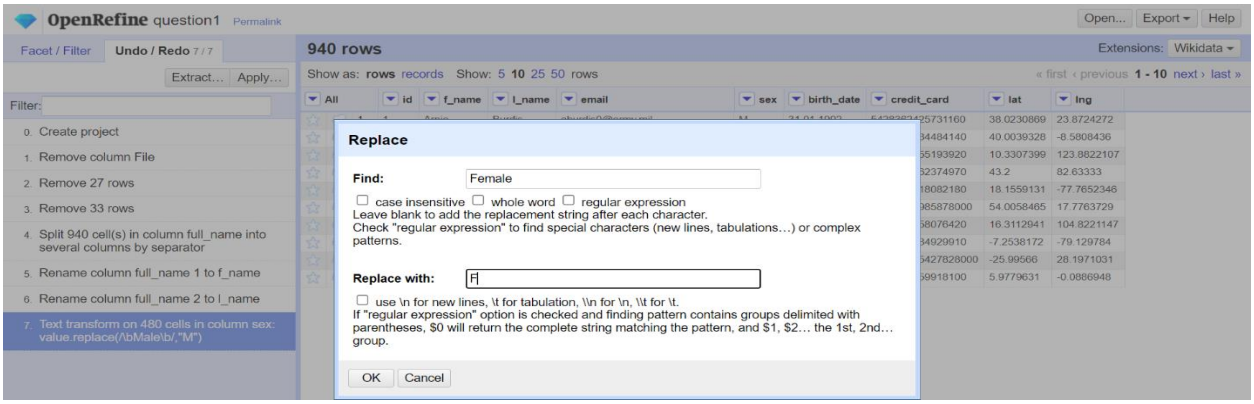


Fig – Replacing Female to F

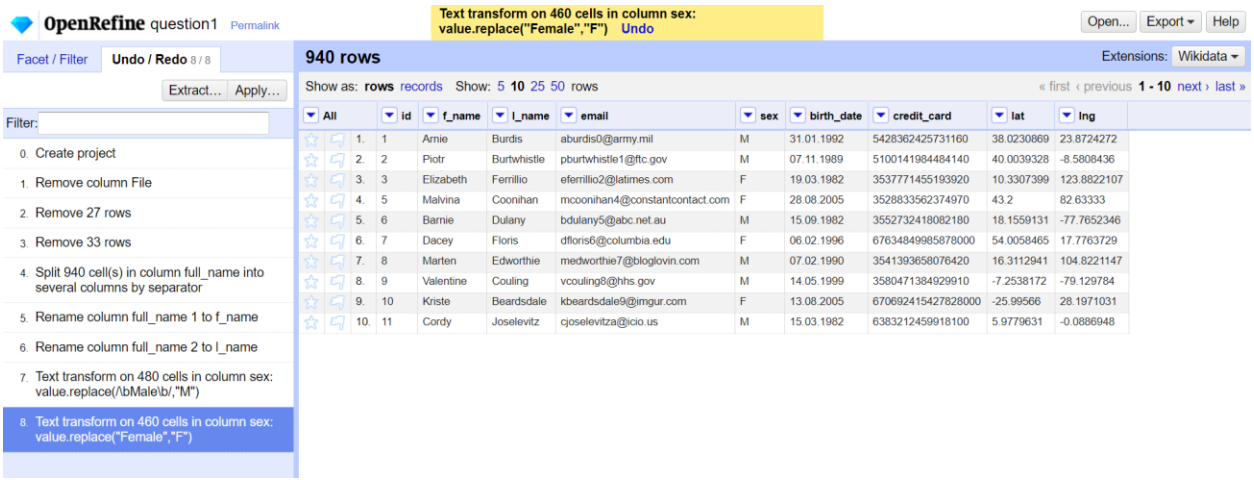


Fig – Transformed Female to F

Row count after transforming Male and Female to “M” and “F” respectively → **940**

8) Calculate age of the person on 24th May 2019 from the birth_date column. Store calculated age in age column (You will create this column).

Steps –

- Choose Add column based on this column.
- Provide new column name "age" New column name age .
- Use GREP to determine the age.

Expression Language General Refine Expression Language (GREL)

```
diff(toDate("24/05/2019","dd/MM/yyyy"),toDate(value.split(".").join(
"/"),"dd/MM/yyyy"), "days")/365
```

No syntax error.

- Click on ok to create the new age column.

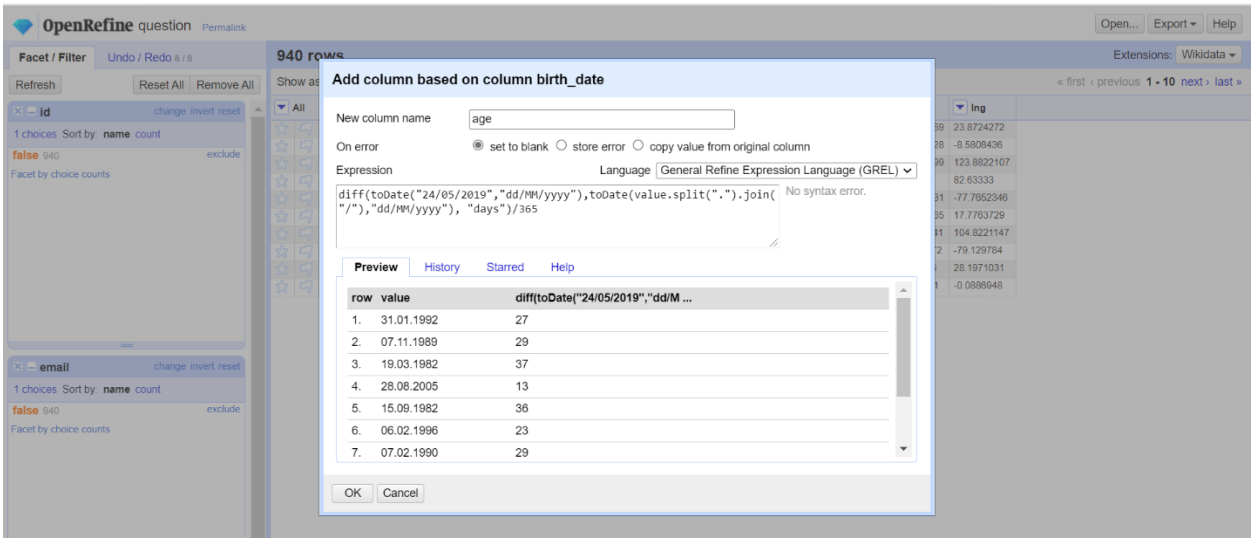


Fig – Adding age column based on birth_date

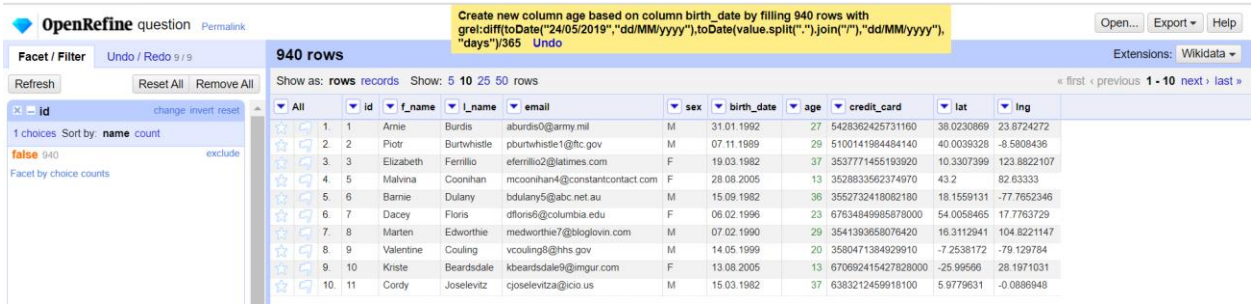


Fig – Age column added

Row count after adding age column → **940**

9) Only keep last 4 digits of credit card in the credit_card column (Do not create a new column).

Steps –

- Choose custom text transform.
- Provide the expression `value.slice(-4)`. (Slice -4 provides the last four digits)
- Then click ok OK to apply changes.

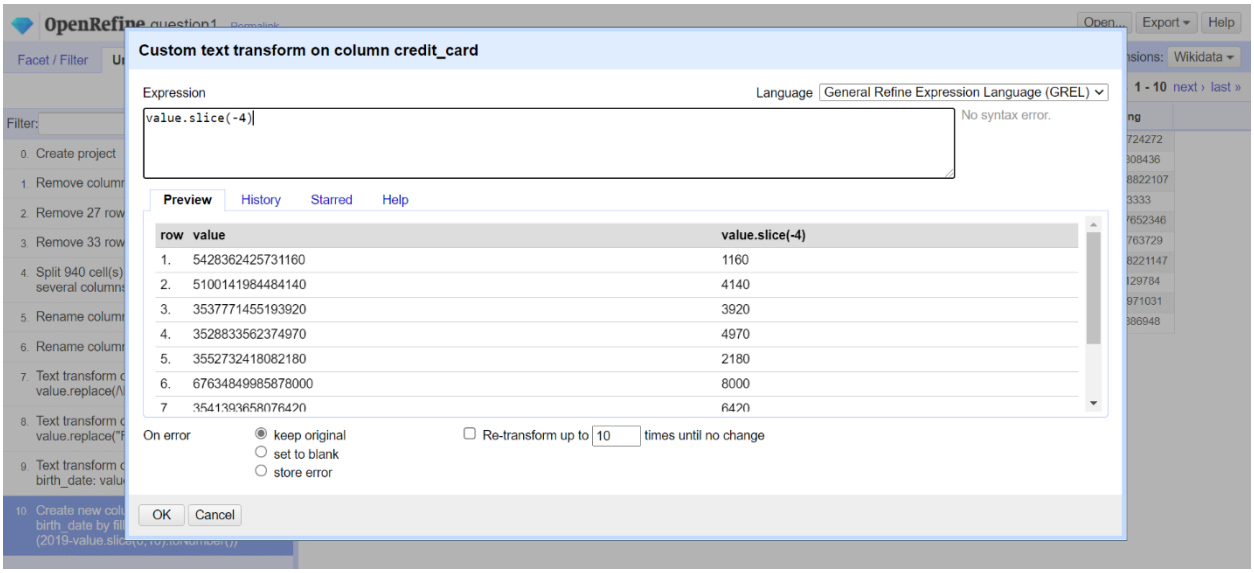


Fig – Slicing last 4 digits of credit card

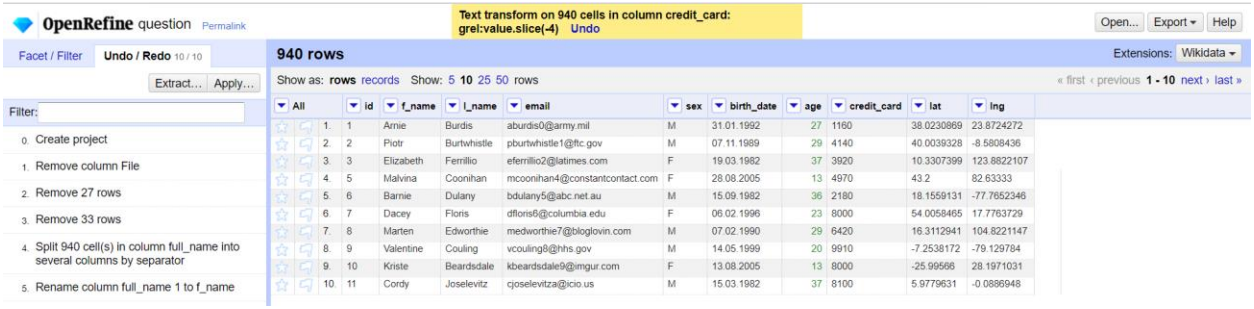
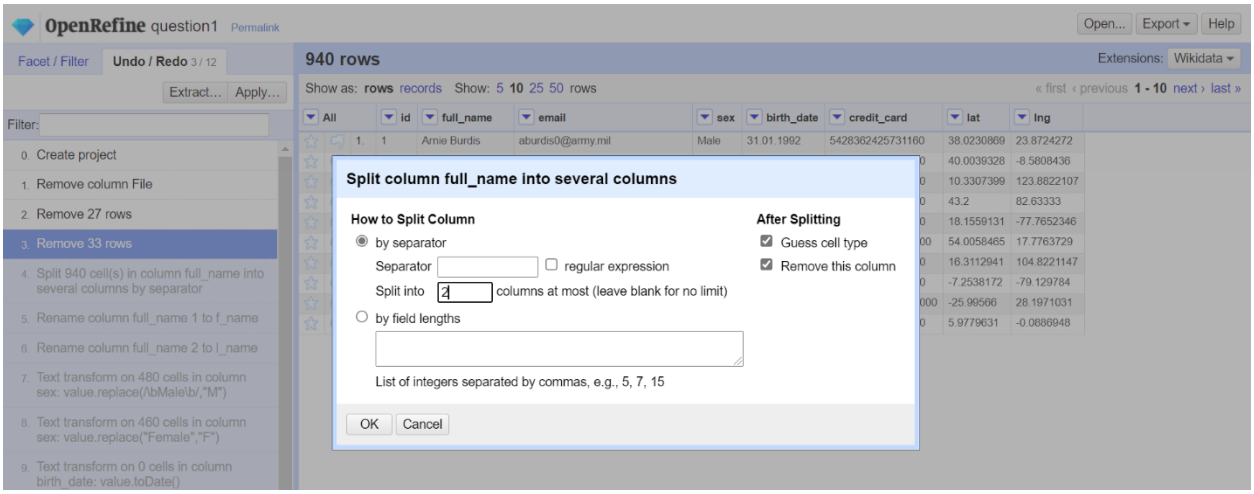


Fig – Transformed credit card containing last 4 values

Row count after transforming credit card values → **940**

10) Delete full_name column.

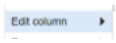
Note - For solving question 6 while splitting the full name column is removed by checking this option ☑ Remove this column

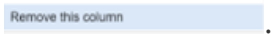


Row count after deleting full_name → **940**

11) Delete birth_date column.

Steps –

- Delete birth_date by selecting the column birth_date, then select 

followed by 

.

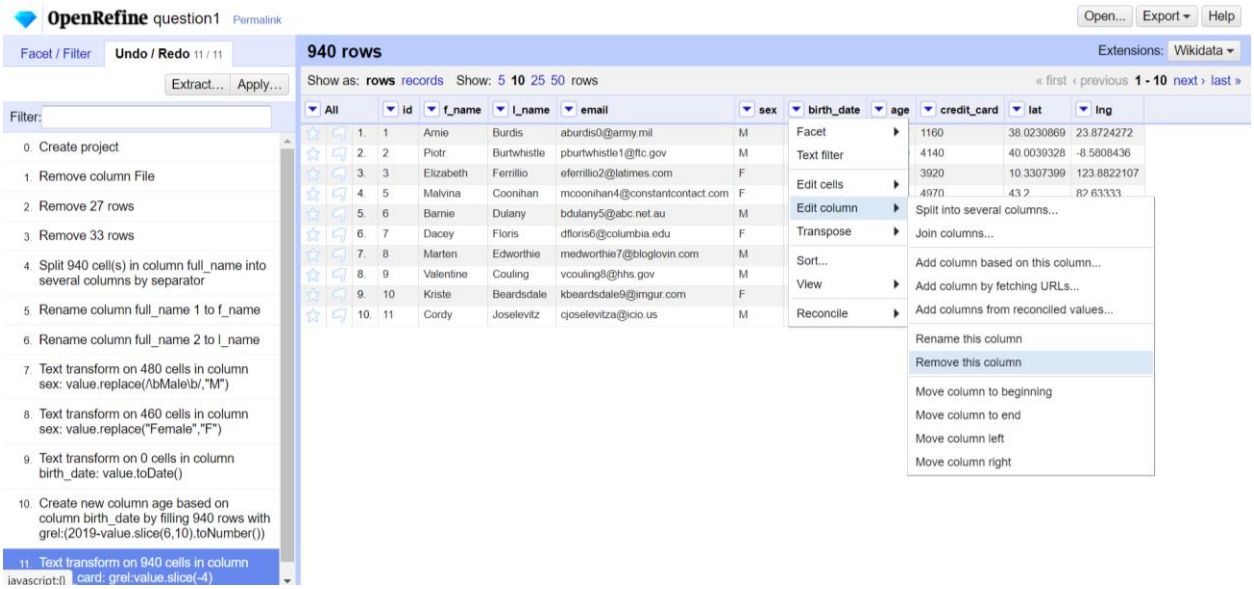


Fig – Removing birth_date column

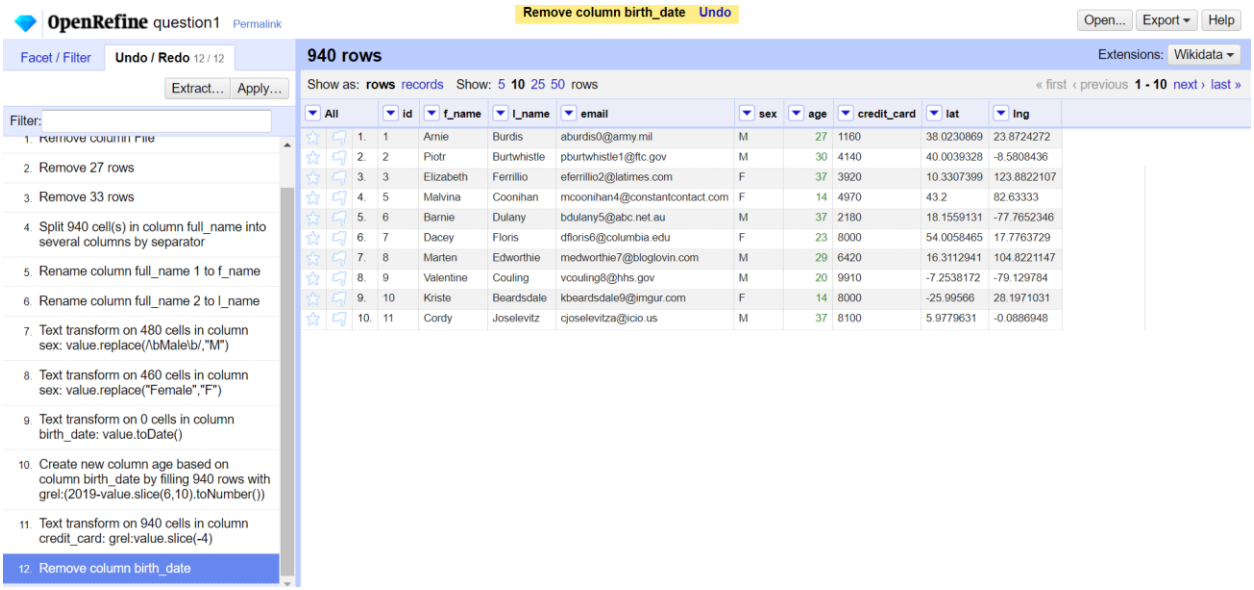


Fig – Data after deleting birth_date

Row count after deleting birth_date → **940**

12) Signup for OpenCage Reverse Geocoding API (https://opencagedata.com/api) and get API Key.

13)URL format to retrieve location (JSON Data): https://api.opencagedata.com/geocode/v1/json?q=LAT+LNG&key=YOUR-API

14)You can replace the text highlighted in red using expression and retrieve location detail (JSON data). You can store this JSON data into raw_data column (You will create this column).

Steps –

- Replacing the latitude and longitude of Halifax in the URL - https://api.opencagedata.com/geocode/v1/json?key=ce3d3605adad476eb6d50777f3265ccc&q=**44.6820**%2C-**63.7443**
- Create a project with the above web addresses URL.
- Parse data as CSV/TSV/separator-based files by selecting the option 

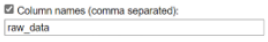
- Change column name to raw data by selecting 

Fig – JSON data

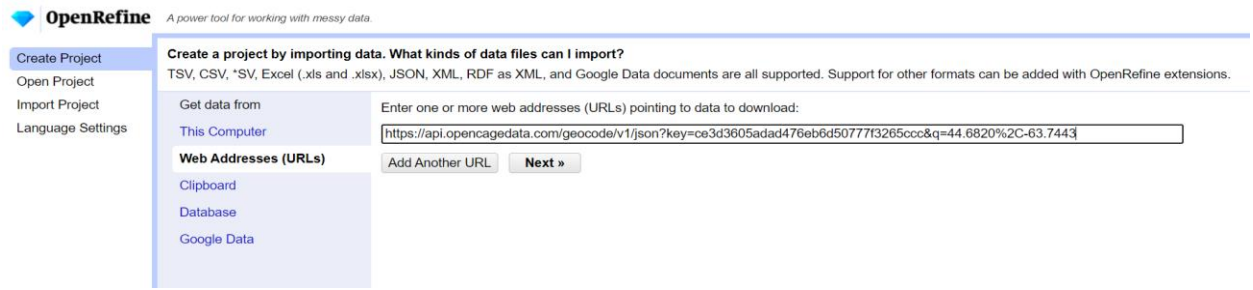


Fig – Creating project from URL

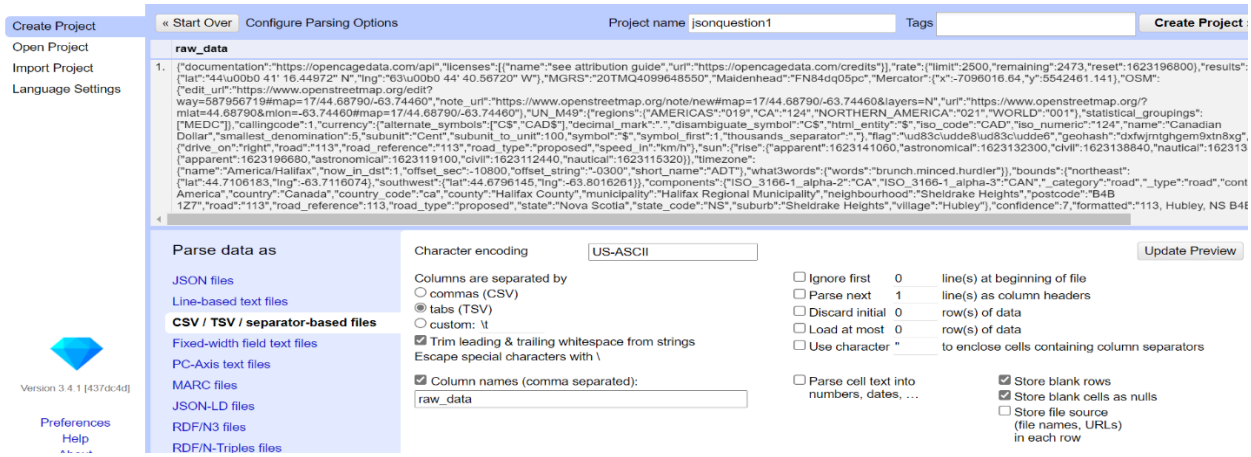


Fig – Stored json data in raw_data column

Fig – Data

15)Filter continent and country from the JSON data and store into continent and country columns (You will create these two columns).

Steps –

- Add column based on column raw_data
- Choose new column name (country and continent) New column name country
- Choose an expression `forEach(value.parseJson().results,v,v.components.country).join(",")` to obtain the county and continent.
- New column country and continent will be added to the data.

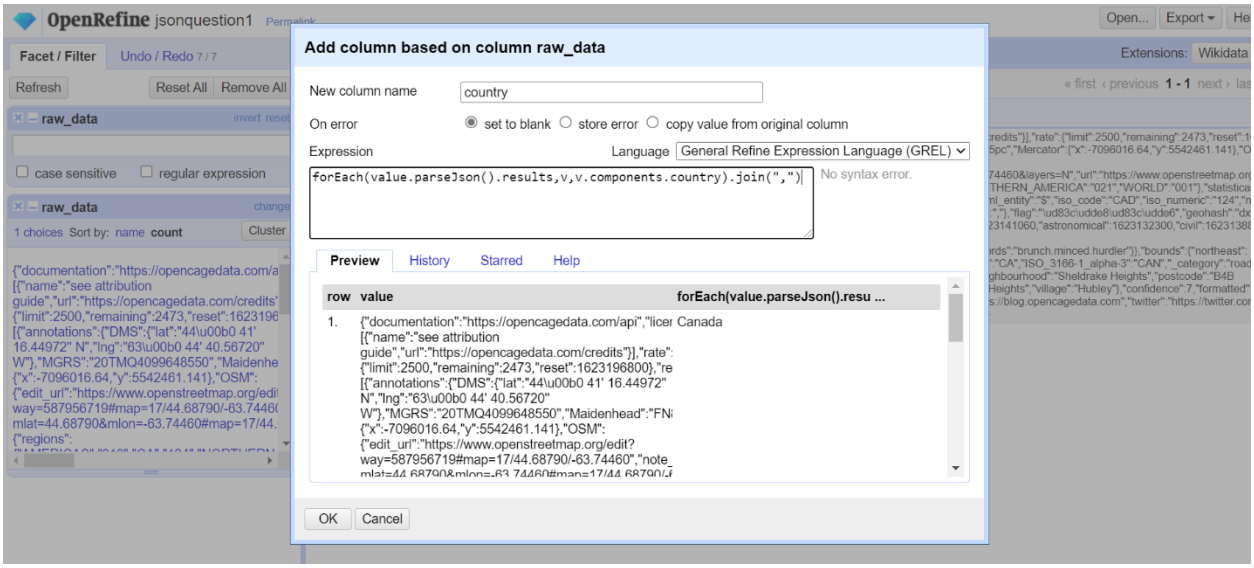


Fig – Adding country column

Fig – Added country column

Fig – Country column

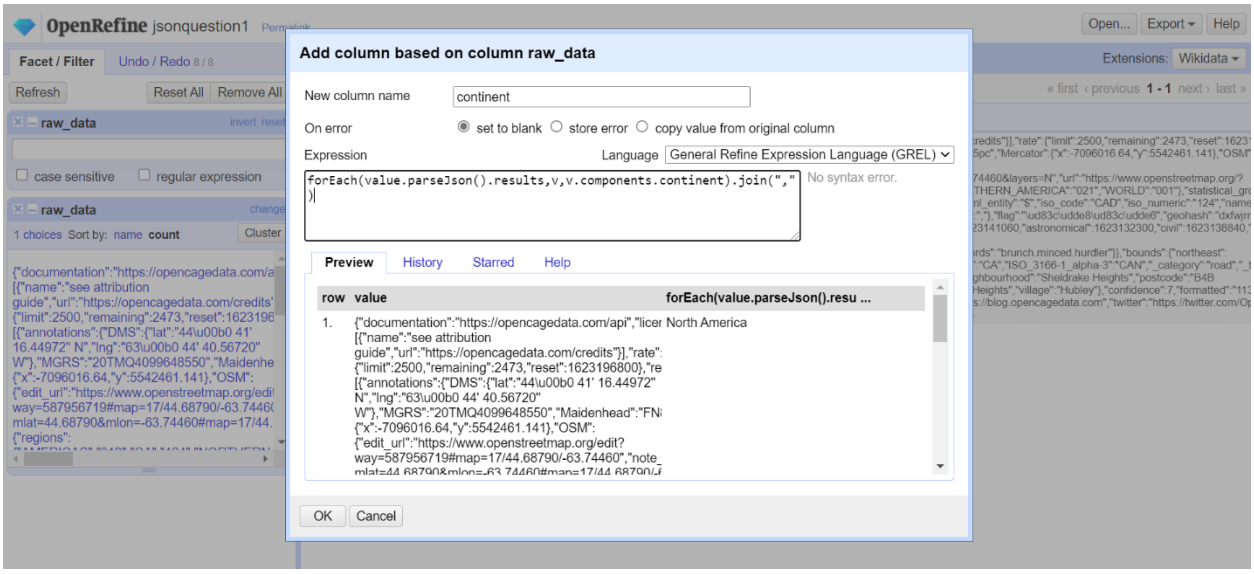


Fig – Adding continent column

Fig – Added continent column

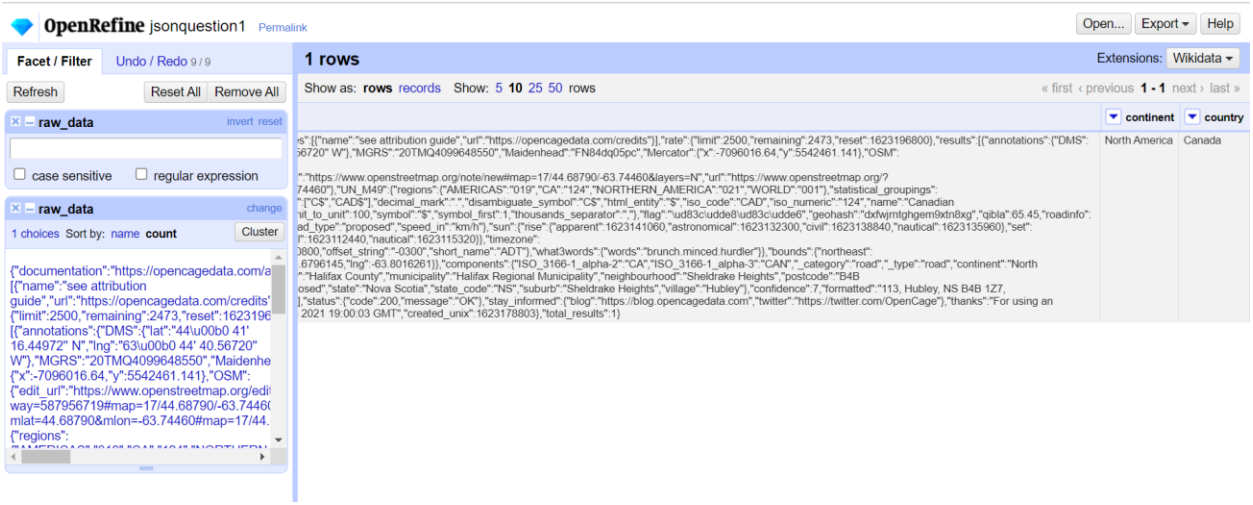
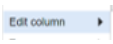
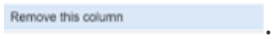


Fig – Continent column

16)Delete raw_data column.

Steps –

- Delete raw_data by selecting the column raw_data, then select 

followed by 

.

Fig – Deleted raw_data column

17)Export project as well as CSV file. Rename project file as solution1.tar.gz and rename CSV file as solution1.csv

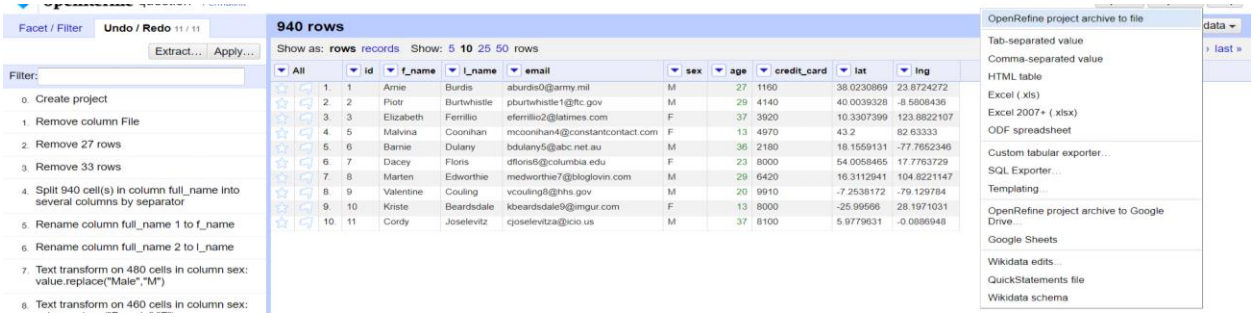


Fig – Export project archive file

Data Preparation using Google Cloud

Task - Loading data

Steps -

- Click on Import data and add to data flow
- Load the data set by choosing file Choose a file .
- Import data set question2_1.csv, question2_2.csv and question2_3.csv by clicking on Import & Add to Flow .

(Note – use Edit settings Edit settings to consider row 1 as header)

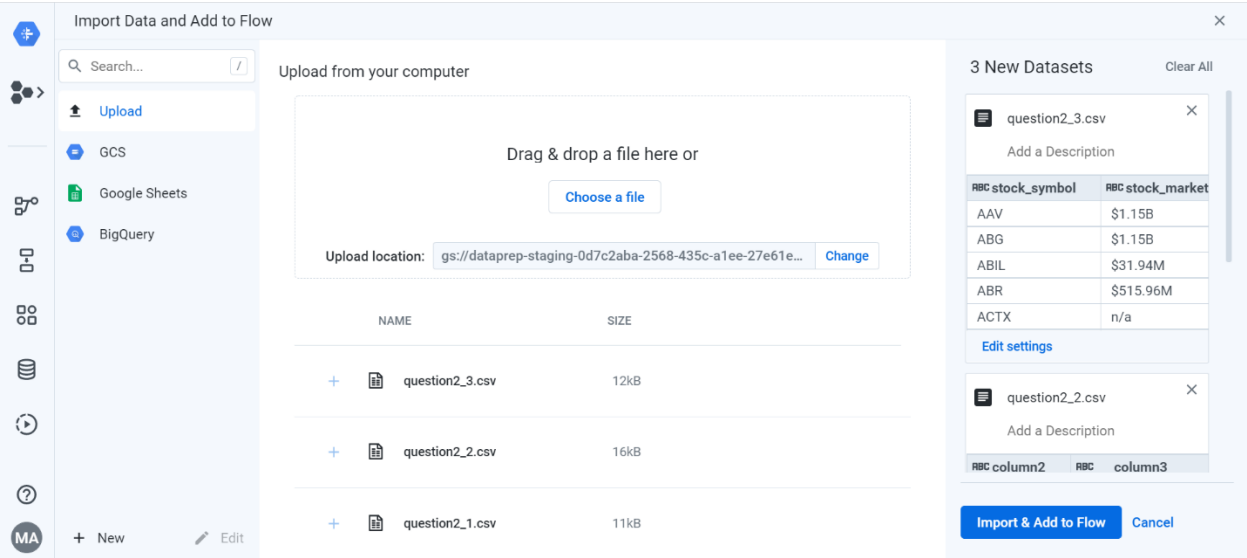
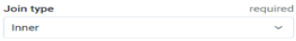


Fig – Adding dataset to data flow

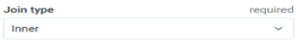
Task – Joining dataset question2_1, question2_2, question2_3

Steps –

- Add Recipe for joining data and select the dataset and click on accept 

.
- Performing inner join 

operation based on key stock_symbol of question2_1.csv and question2_2.csv
- Retaining only the required columns such as stock_symbol, stock_market and stock_sector columns join operation and click on review 

.
- On the same recipe, join question2_3.csv along with already joined question2_1.csv and question2_2.csv.
- Performing inner join 

operation based on key stock_symbol of question2_1.csv, question2_2.csv and question2_3.csv
- Retaining only the required columns such as stock_symbol, stock_market, stock_sector and stock_market_cap columns join operation and click on review 

.

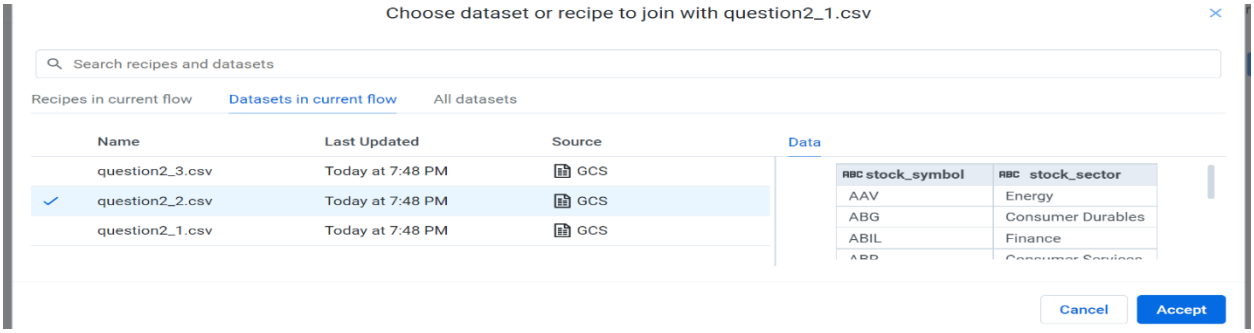


Fig – Joining data with stock_symbol (question2_1.csv and question2_2.csv)

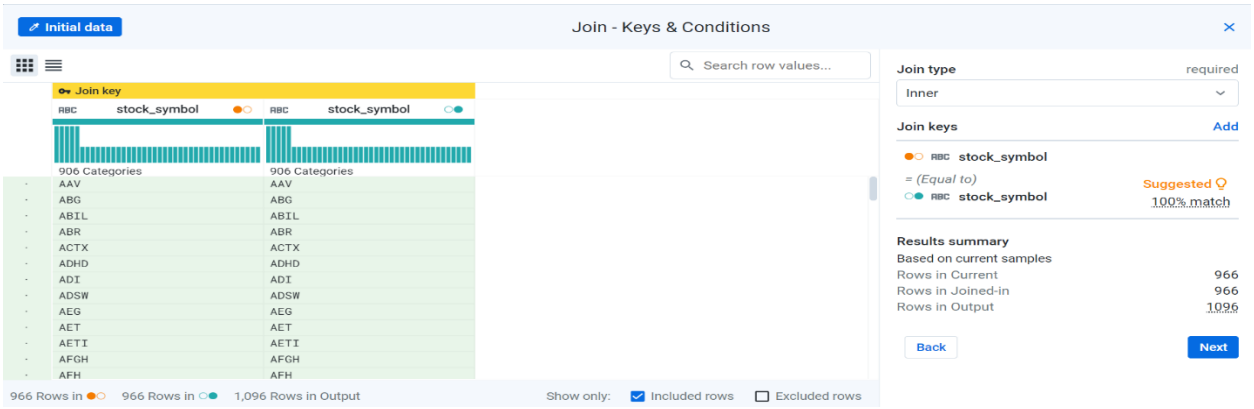


Fig – Performing Inner Join

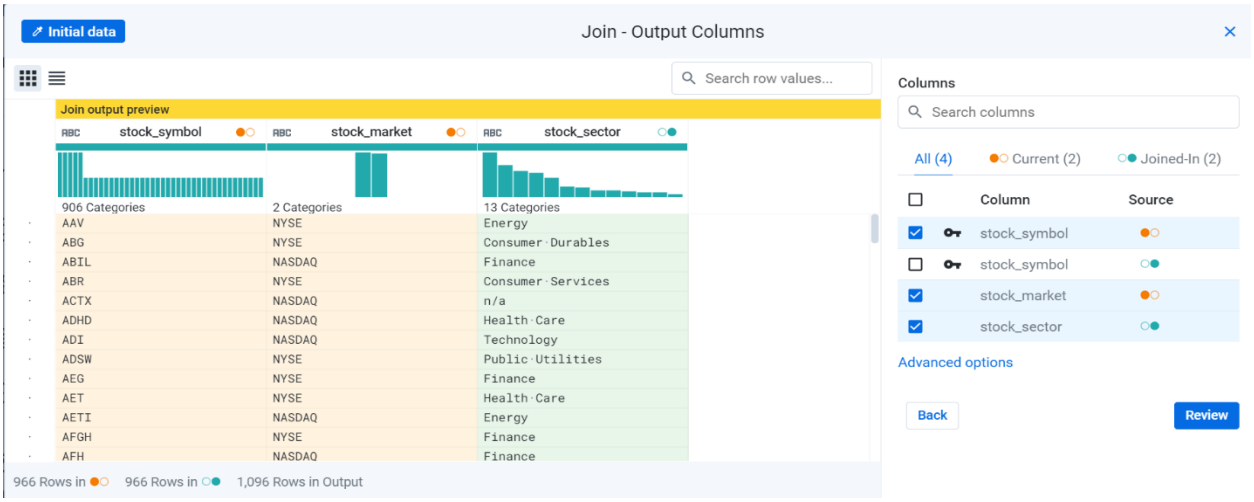


Fig – Selecting the required columns

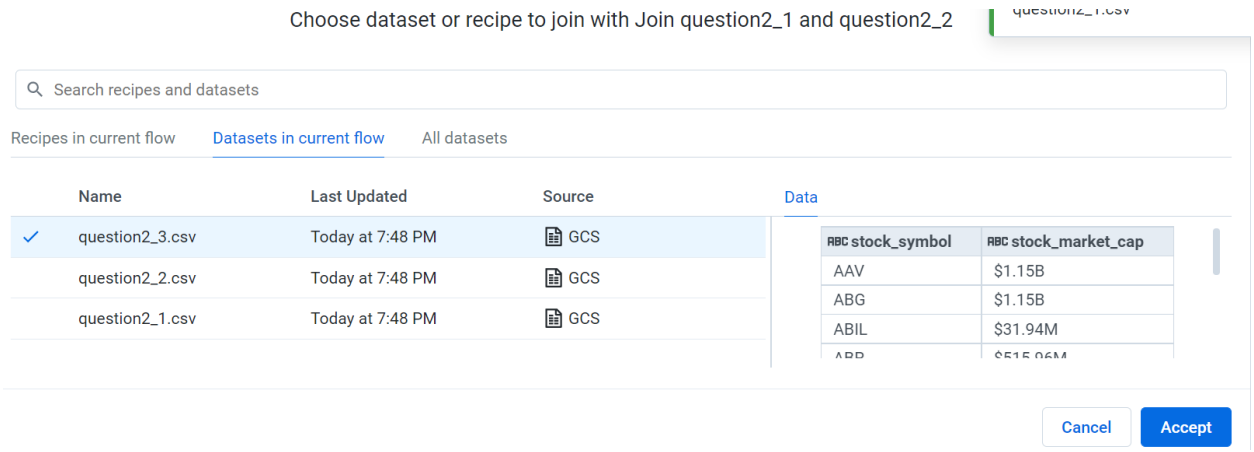


Fig – Merging question2_3.csv with the data with key as stock_symbol

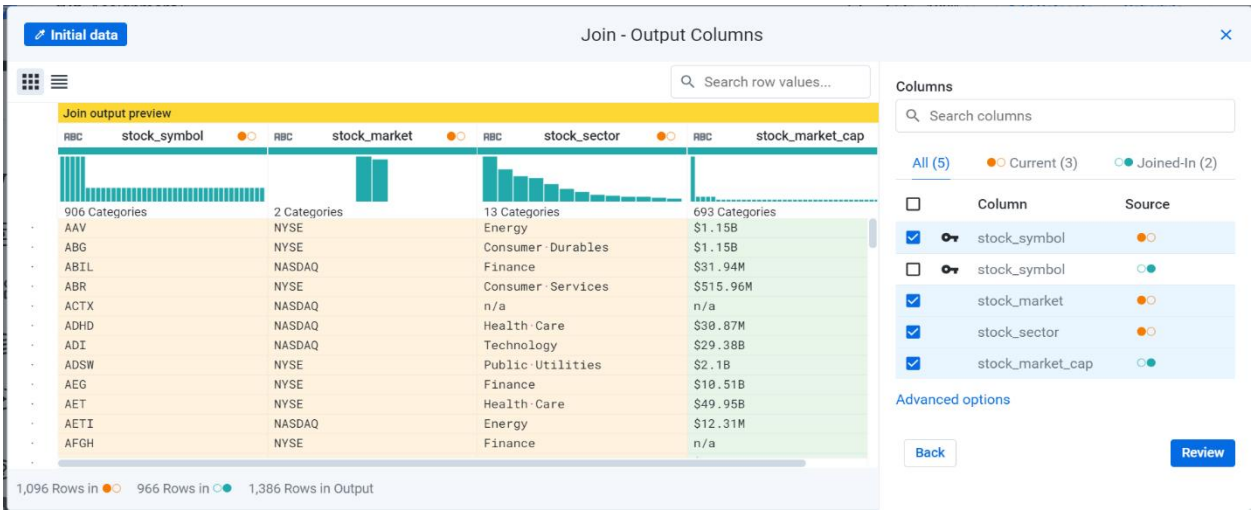


Fig – Joining data with stock_symbol

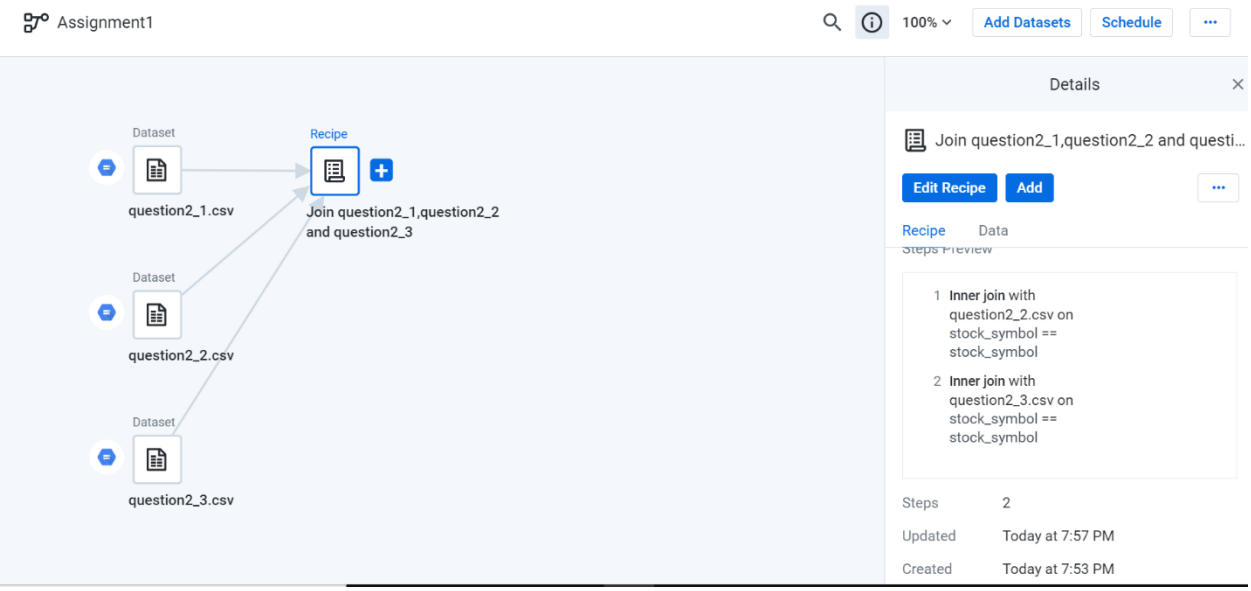


Fig – Joined question2_1, question2_2 and question2_3

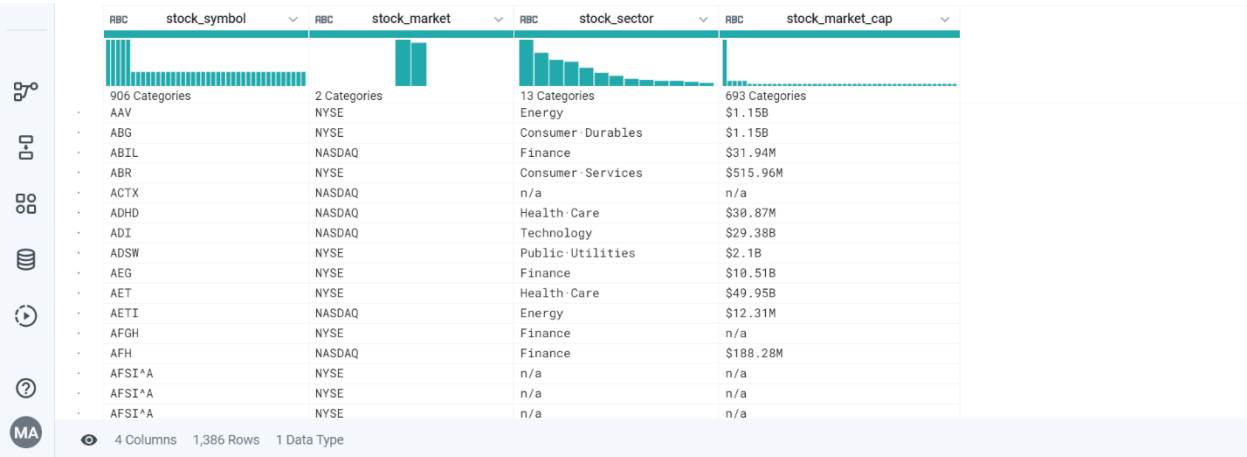


Fig – Joined Data

Row count after join operation → **1386**

Task – Converting Billion to Million

Steps –

- Add a recipe to perform conversion from billion to million.
- Selecting on convert pattern option Convert patterns .
- Provide condition and new value to be replaced the condition.
- Using regular expression to convert stock_market_cap that has data in billion to million (capturing groups).

  /(\d*)\.(\d)B/ (condition) → $1$200M (new value)

  /(\d*)\.(\d)(\d)B/ (condition) → $1$2$30M (new value)

- Then click on add Add to apply new value to stock_market_cap column.

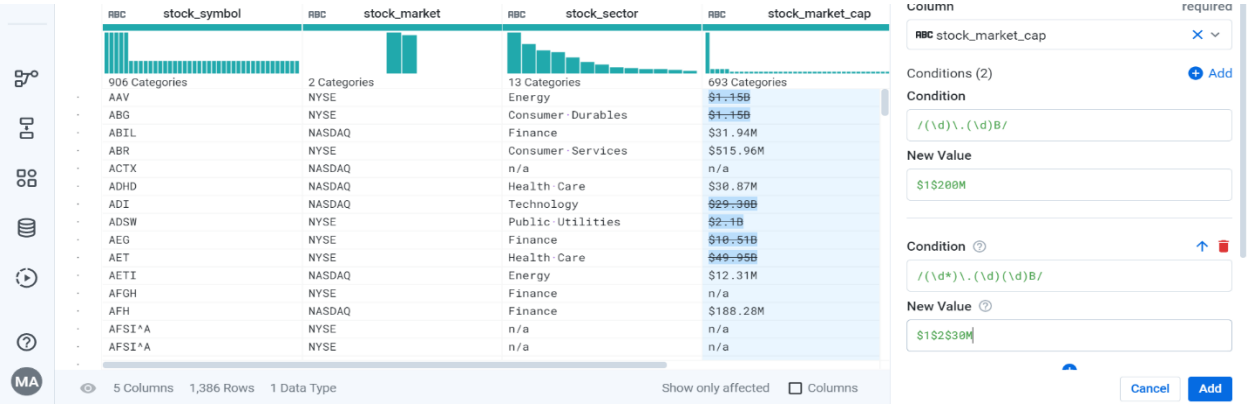


Fig – Regex to convert Billion to Million

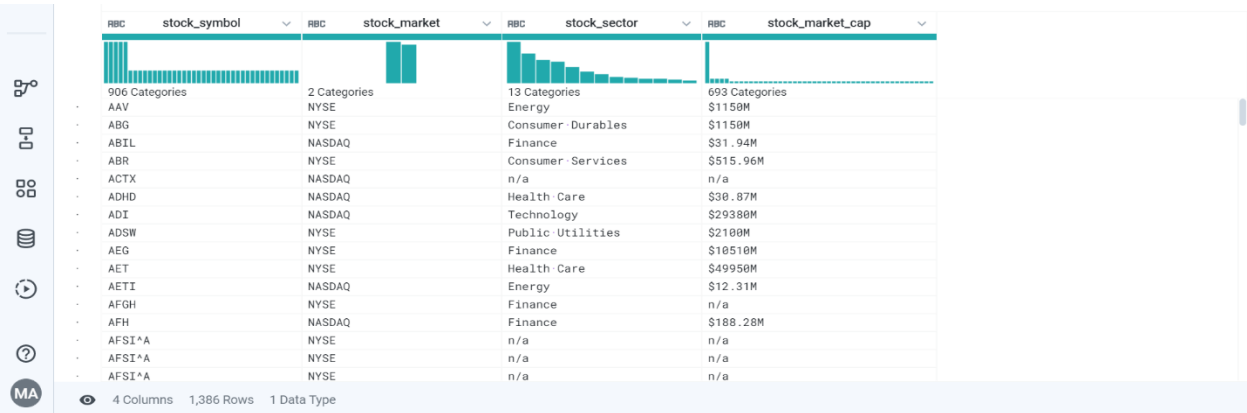


Fig – stock_market_cap billion converted to million

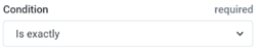
Row count after converting billion to million → **1386**

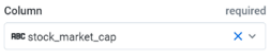
Task – Eliminating "n/a" values

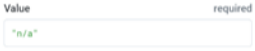
Steps –

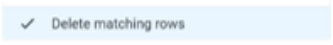
- Add recipe for deleting n/a or blank values.
- On details section, we can see the number of "n/a" 

present in stock_market_cap column (249).
- Select filter rows

, choose condition (is exactly) 

, select the column (stock_market_cap) 

, supply value ("n/a") 

and action (delete the rows that contains "n/a") 

.
- After selections are made then click on add 

to delete.
- Similarly for stock_sector column which has n/a values.

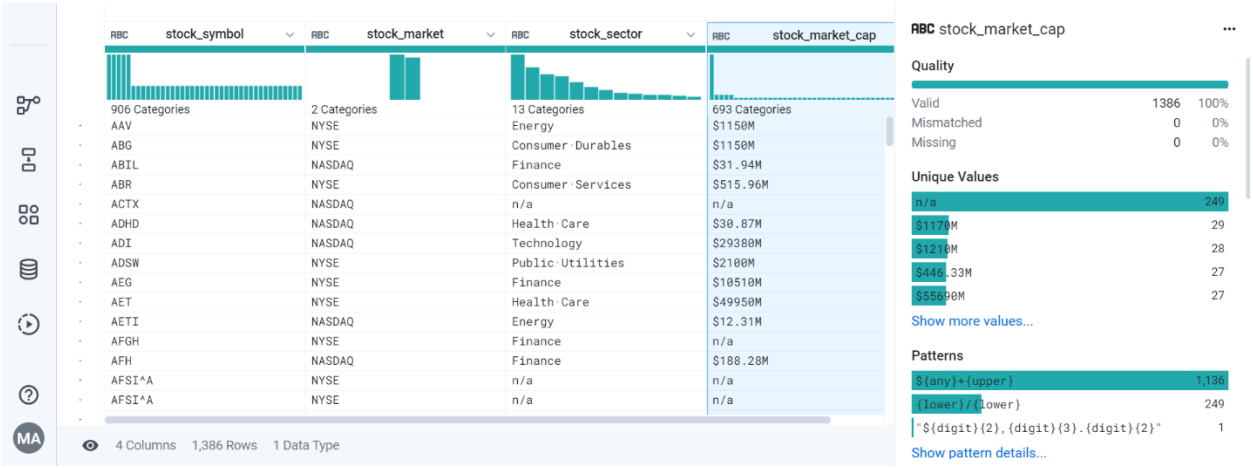


Fig – n/a values present in stock_market_cap

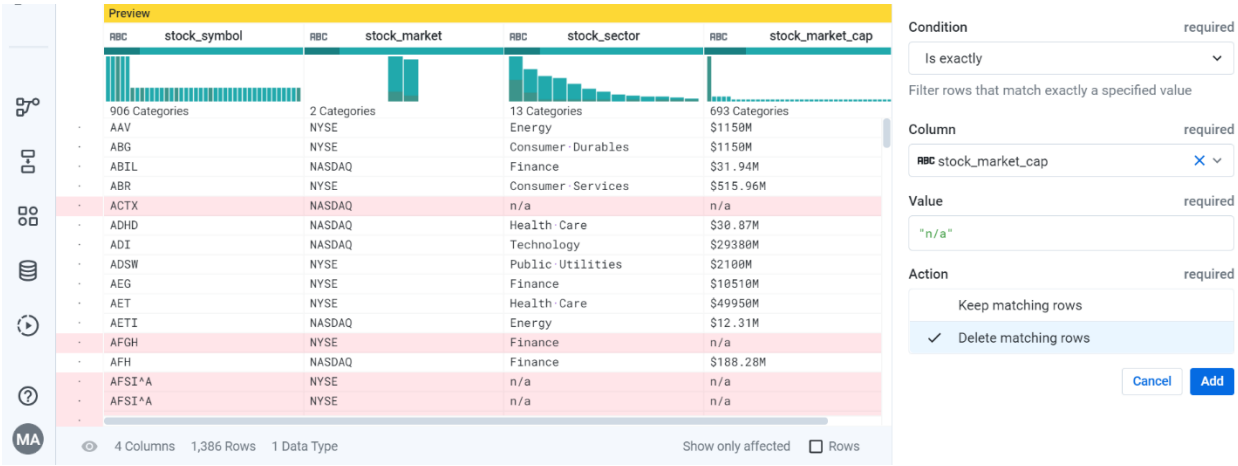


Fig – Deleting rows with stock_market_cap having "n/a"

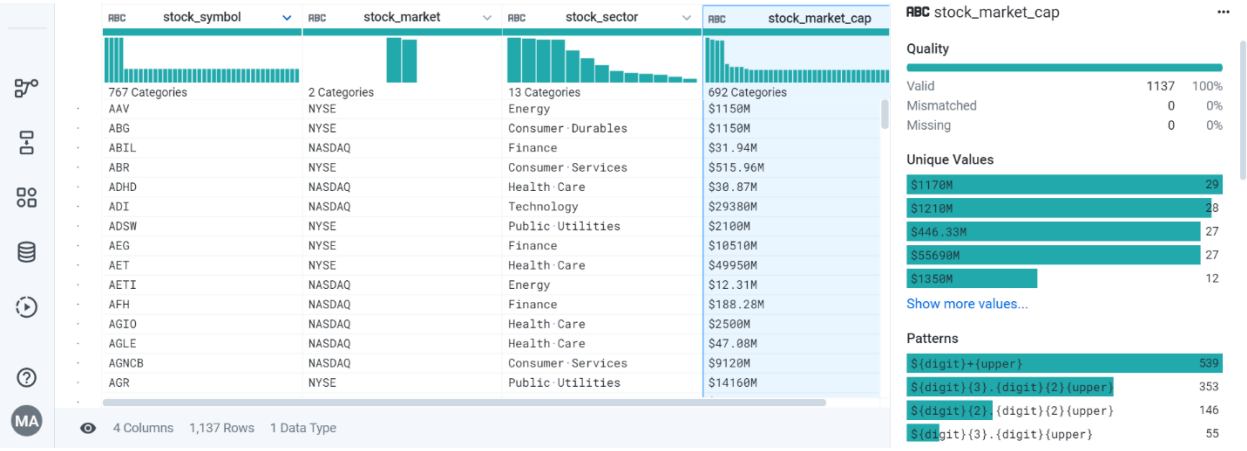


Fig – 0 n/a values present in stock_market_cap

Fig – 169 n/a values present in stock_sector

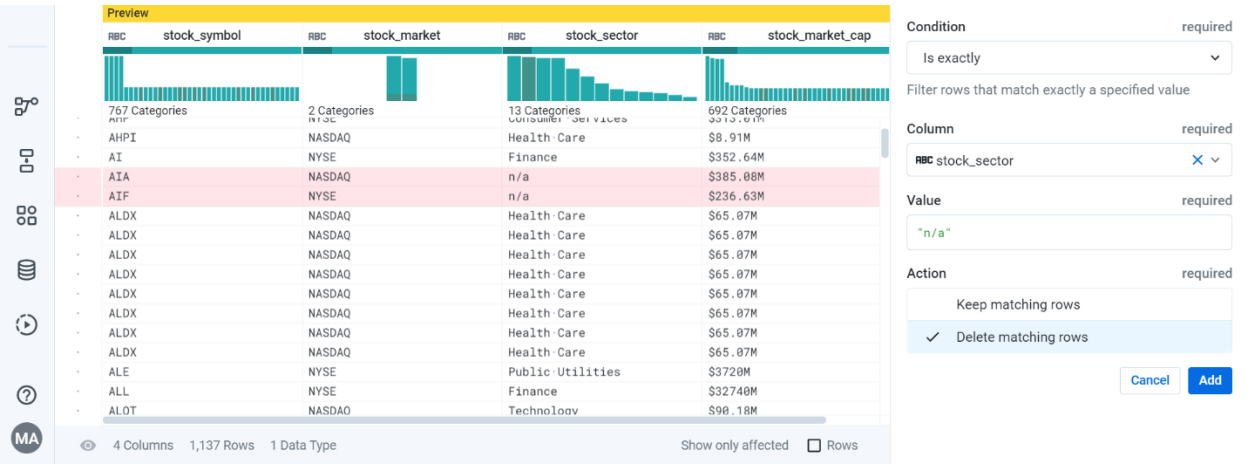


Fig – Eliminating stock_sector with n/a values

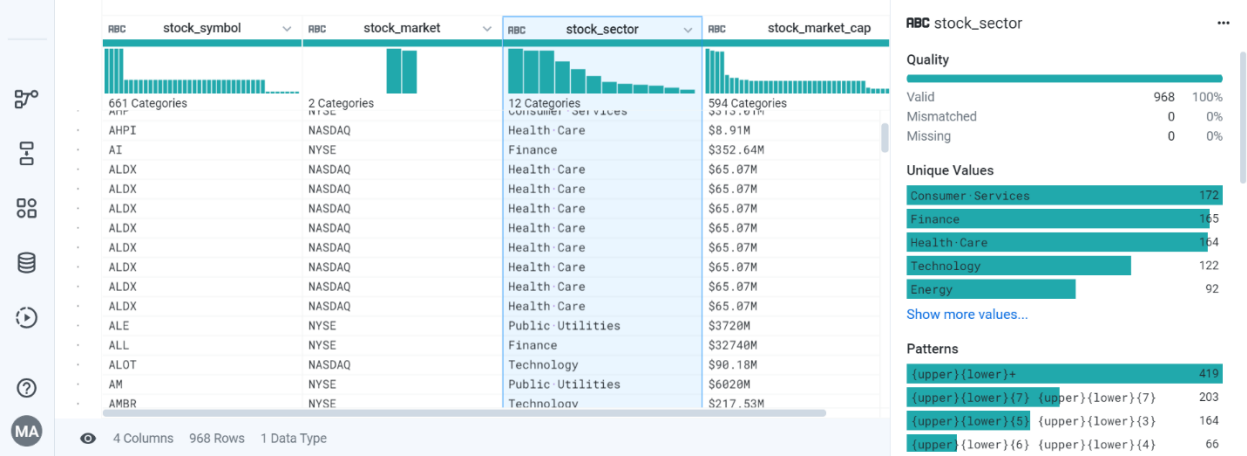


Fig – 0 n/a for stock_sector

Row count after eliminating "n/a" values → **968**

Task – Selecting records greater than 30M

Step –

- Stock_market_cap has text data hence it must be converted to decimal value.
- Filter rows containing mismatched values by selecting condition (is mismatched), select column (stock_market_cap), delete matching rows with mismatched values.
- Again, filtering rows based on stock_market_cap with condition (is greater than), selecting value greater than (30) and keep the rows which match the condition.
- Click add Add to make required changes.

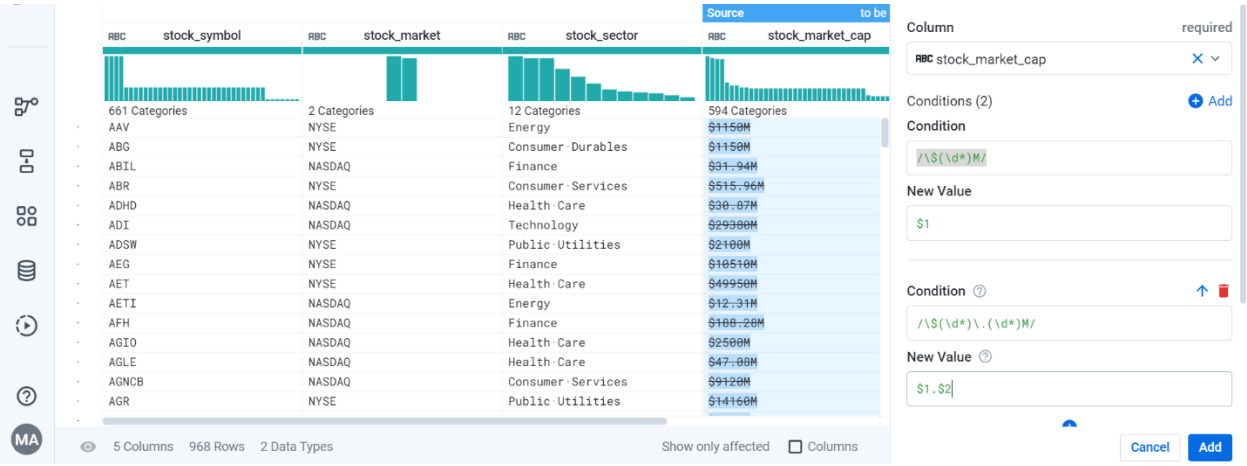


Fig – Converting stock_market_cap to decimal

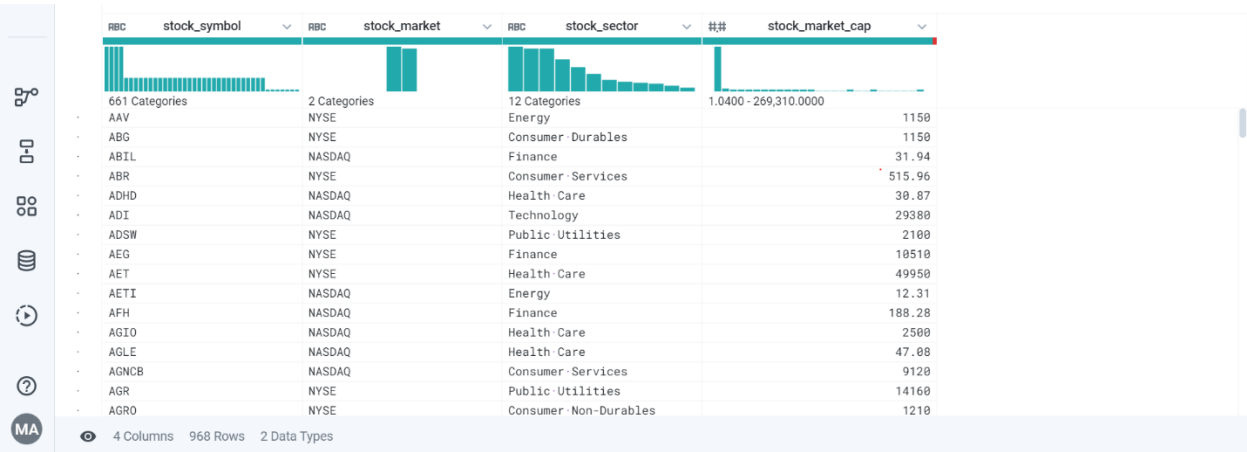


Fig – Decimal value for stock_market_cap

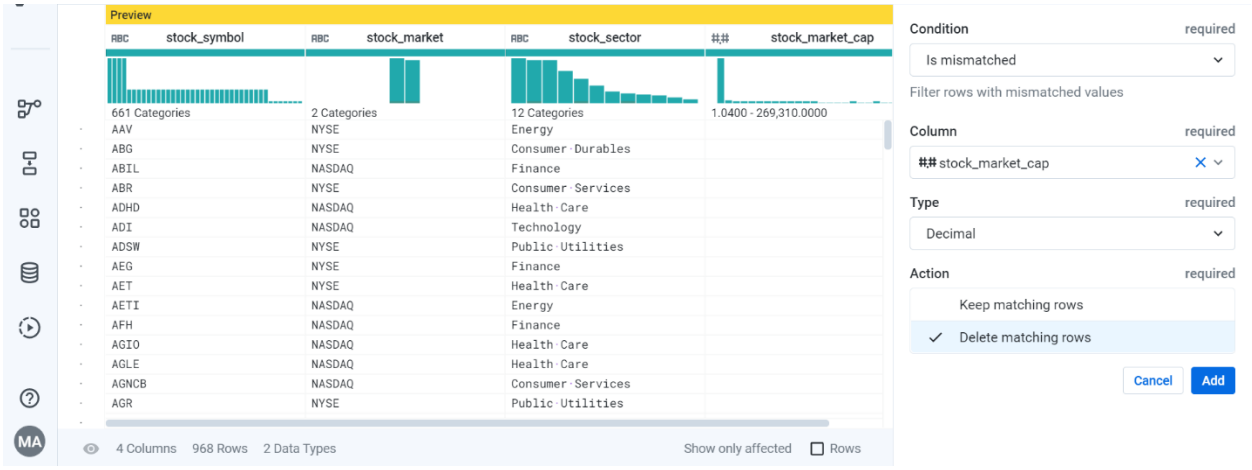


Fig – Deleting mismatched rows

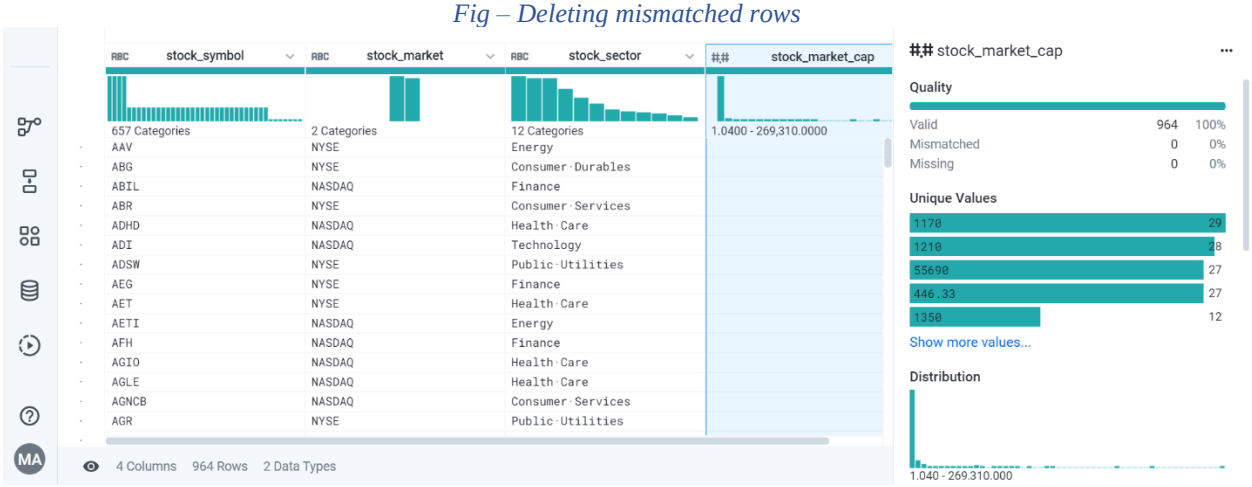


Fig – Deleted mismatched rows

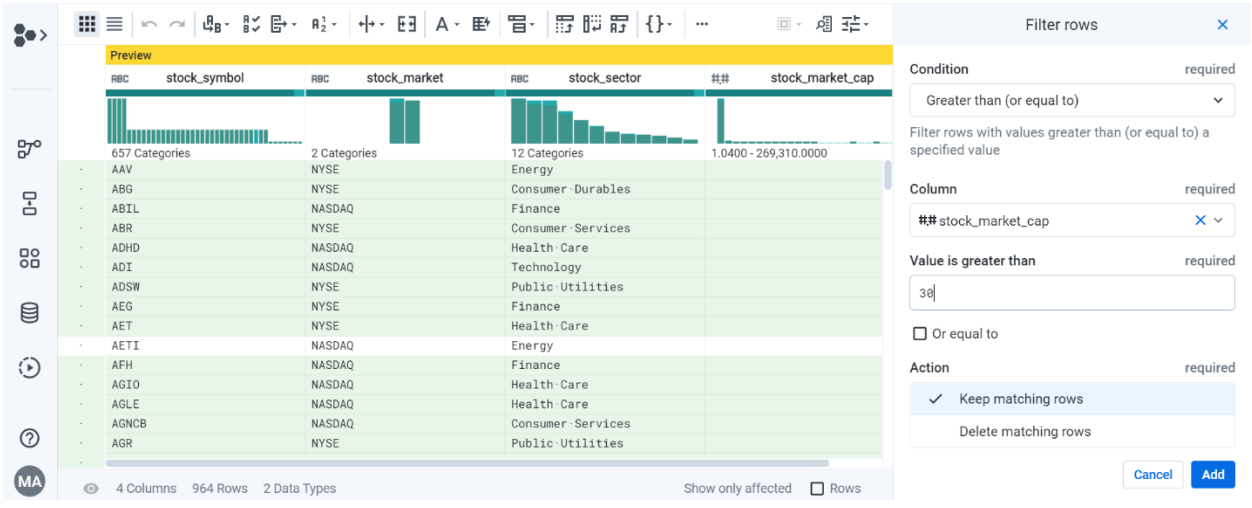


Fig – Selecting stock_market_cap greater than 30 million

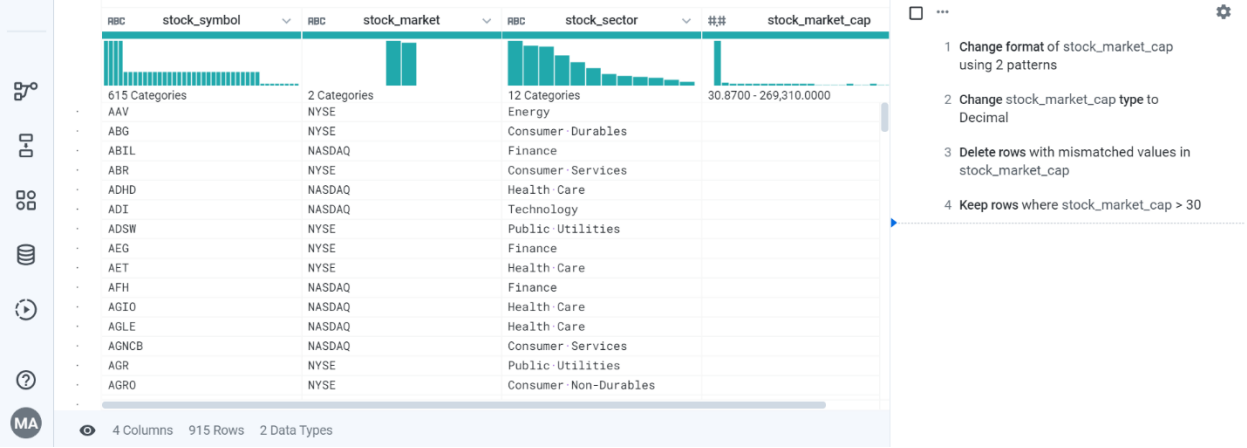
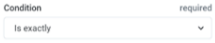
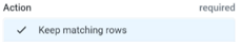


Fig – Stock_market_cap greater than 30

Row count after retaining stock_market_cap greater than 30 → **915**

Task – Splitting the data based on stock_sector for Finance, Technology and Health Care

Steps –

- Add recipe for splitting the data into Finance, Technology and Health Care Sector data.
- Filtering rows for splitting the data based on condition (is exactly)  , value matching (“Finance”)  and action (keep matching rows)  .
- Delete stock_sector column
- Similarly create filter for Technology and Health Care.
- Click add  to make required changes.

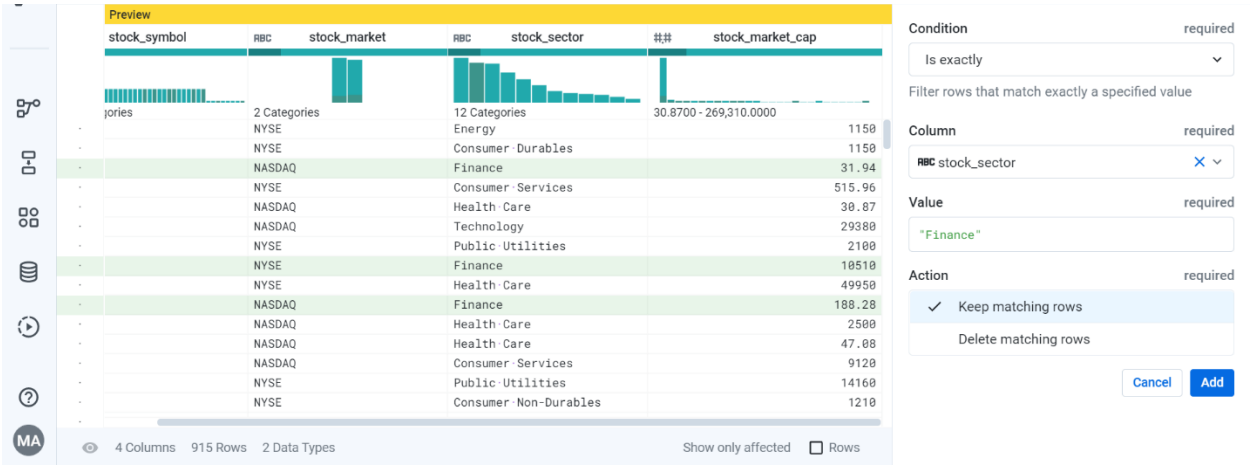


Fig – Finance stock sector

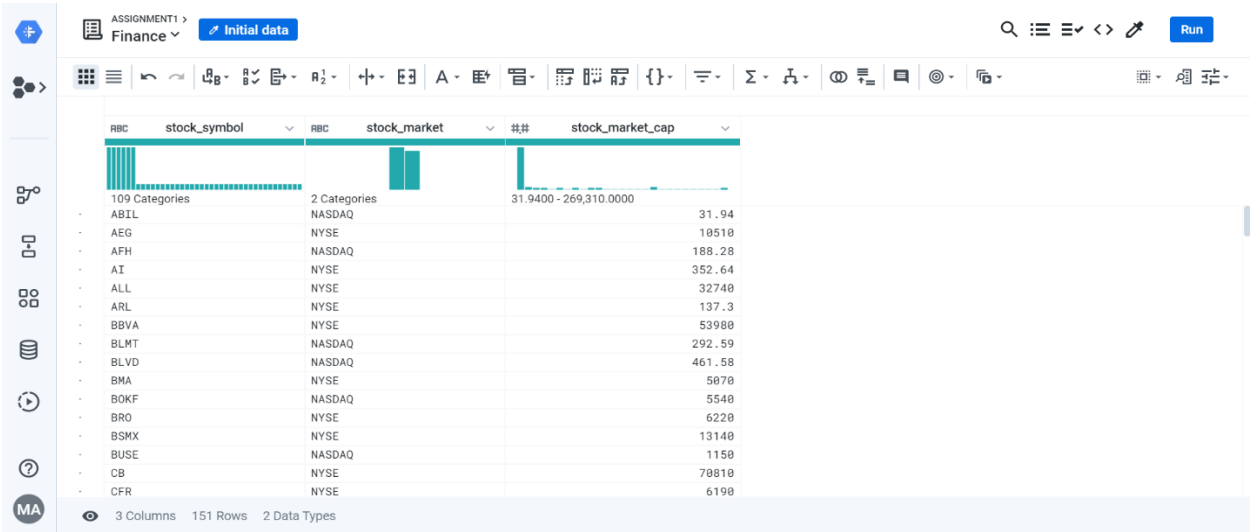


Fig – Deleting stock_sector column

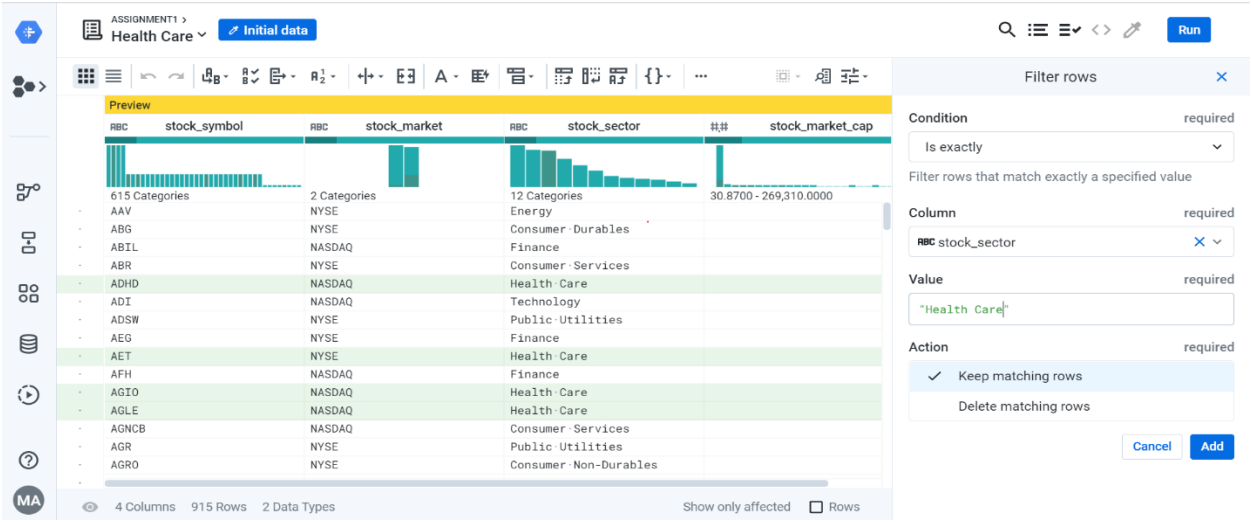


Fig –Health care stock sector

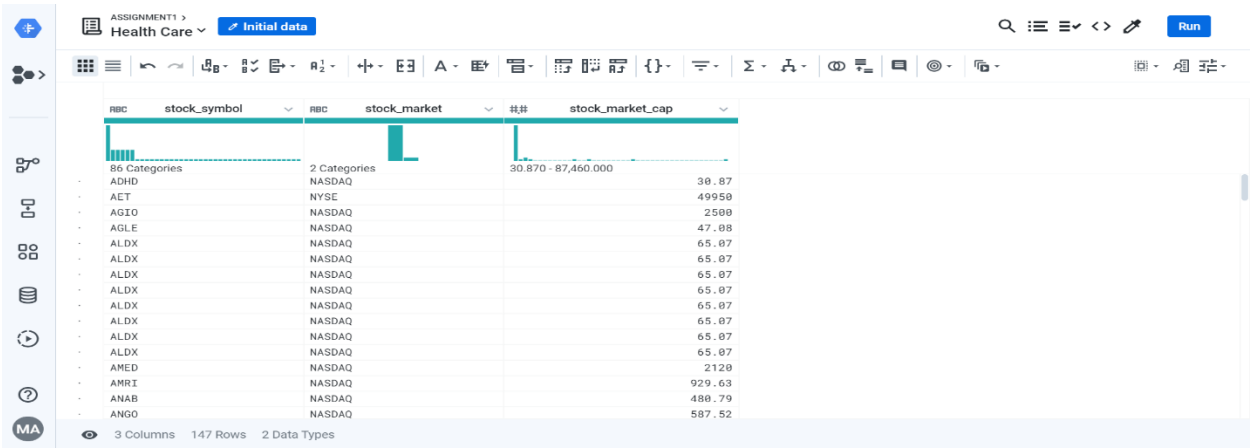


Fig – Deleting stock_sector column

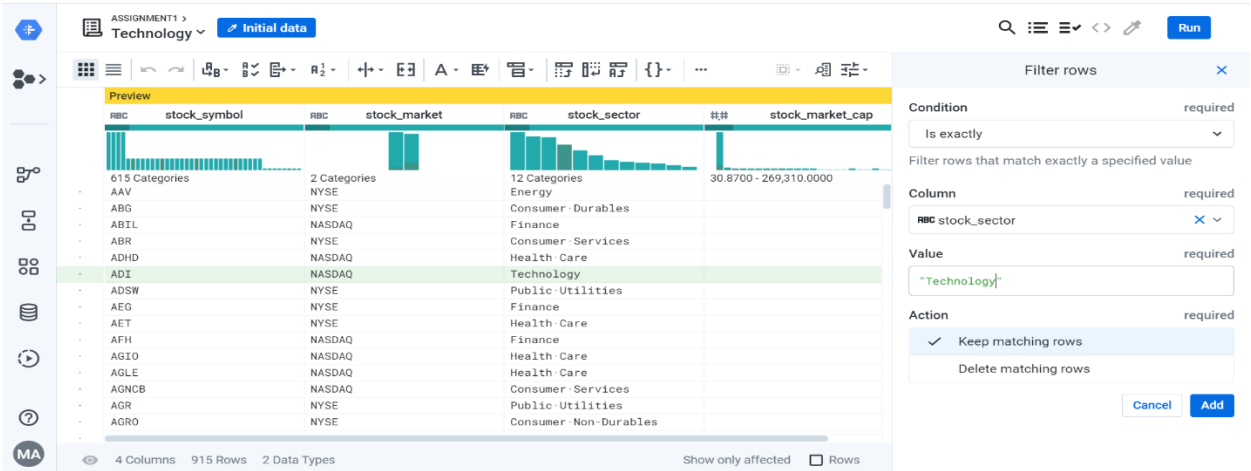


Fig – Technology stock sector

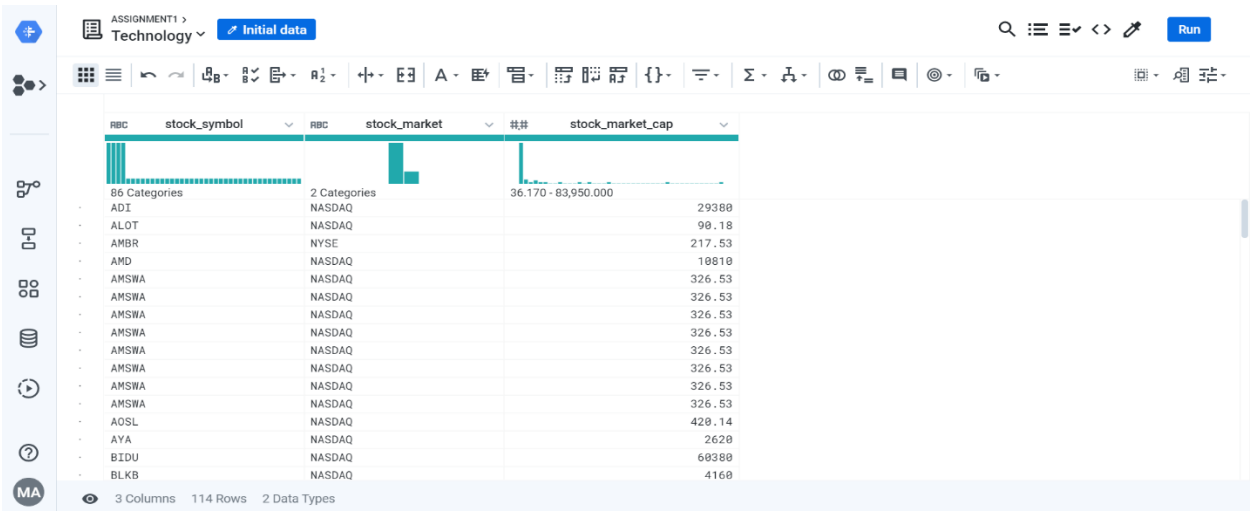


Fig – Deleting stock sector column

Row count for Finance data→ **151**

Row count for Technology data→ **114**

Row count for Health Care data → **147**

Task –Removing duplicate values present in Finance, Technology and Health Care data

Steps –

- Select stock_symbol, stock_market and stock_market_cap and remove duplicate rows `Remove duplicate rows` .
- Rename the stock_market_cap column to stock_market_cap(Million) .

Fig –Eliminating duplicate values

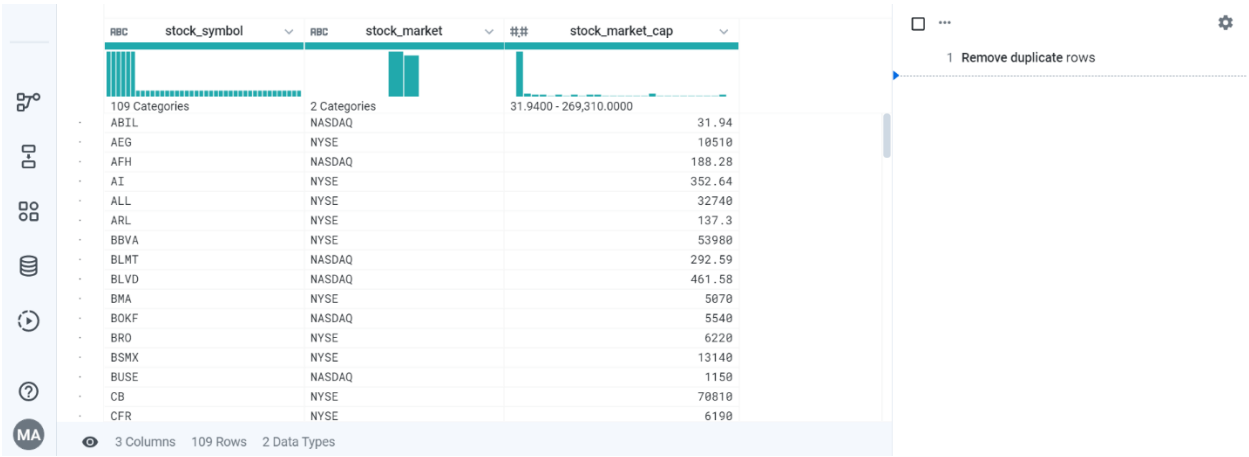


Fig – Data after eliminating duplicate values

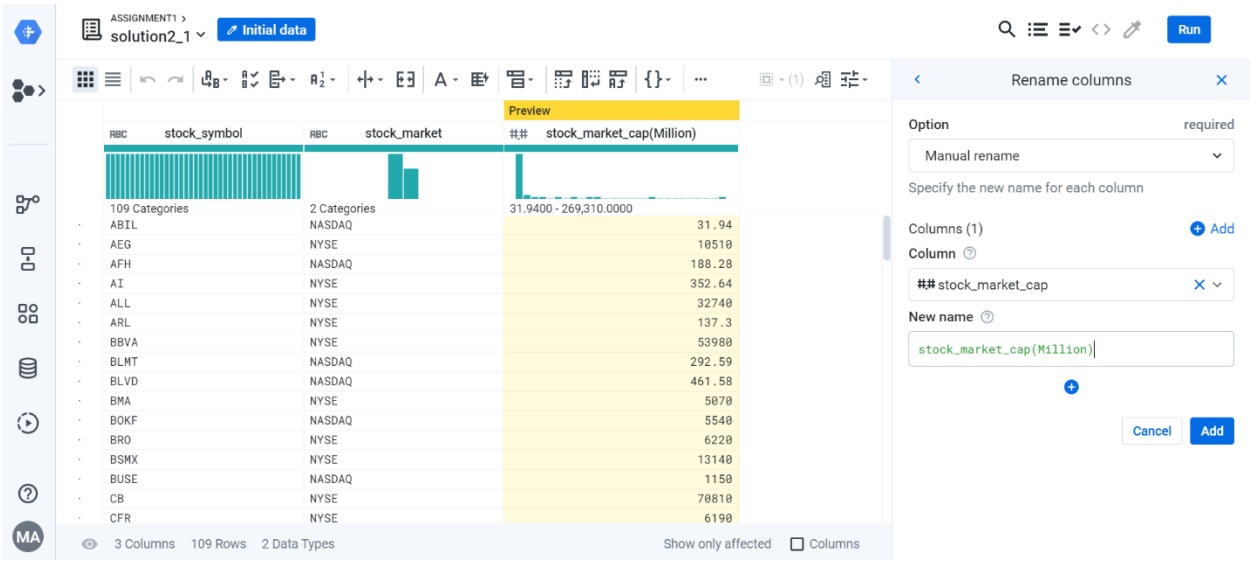


Fig – Renaming the column stock_market_cap(Million)

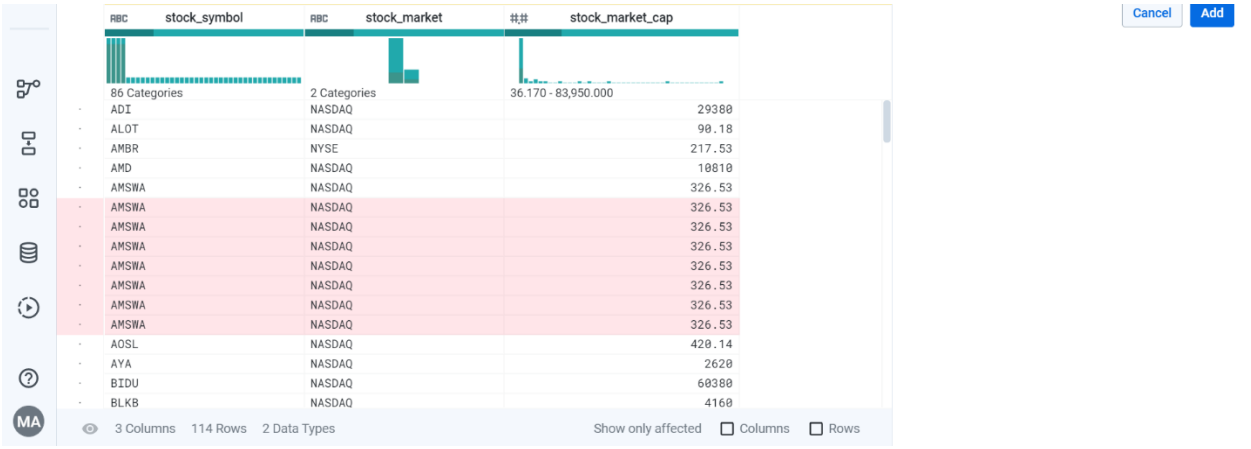


Fig –Eliminating duplicate values

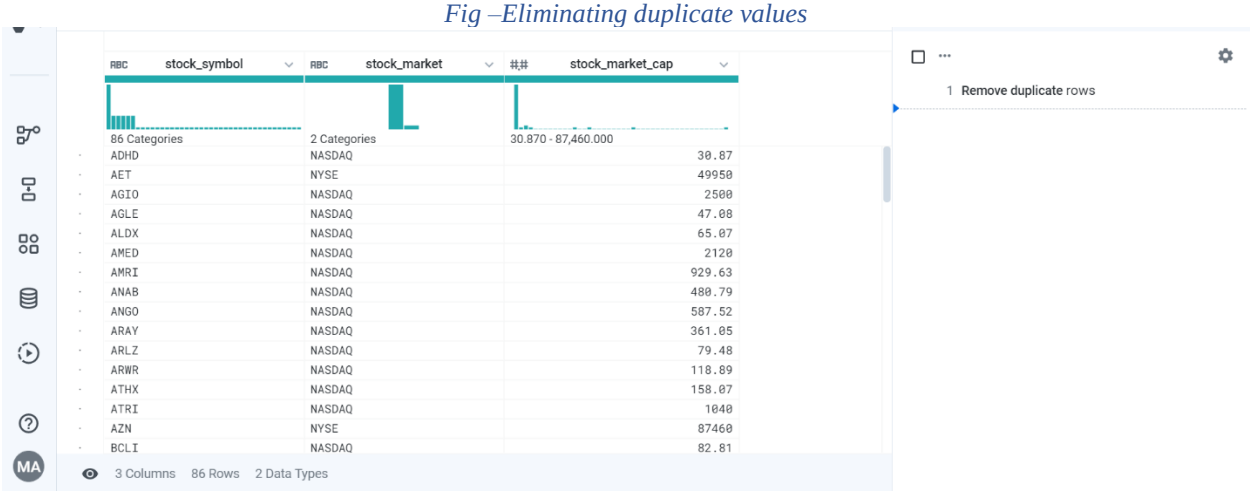


Fig – Data after eliminating duplicate values

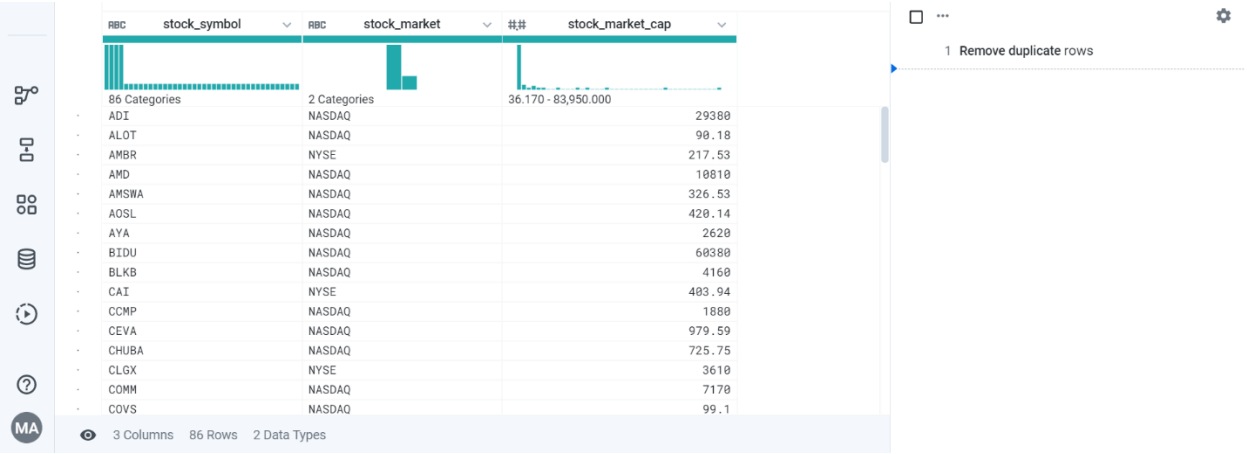


Fig –Eliminating duplicate values

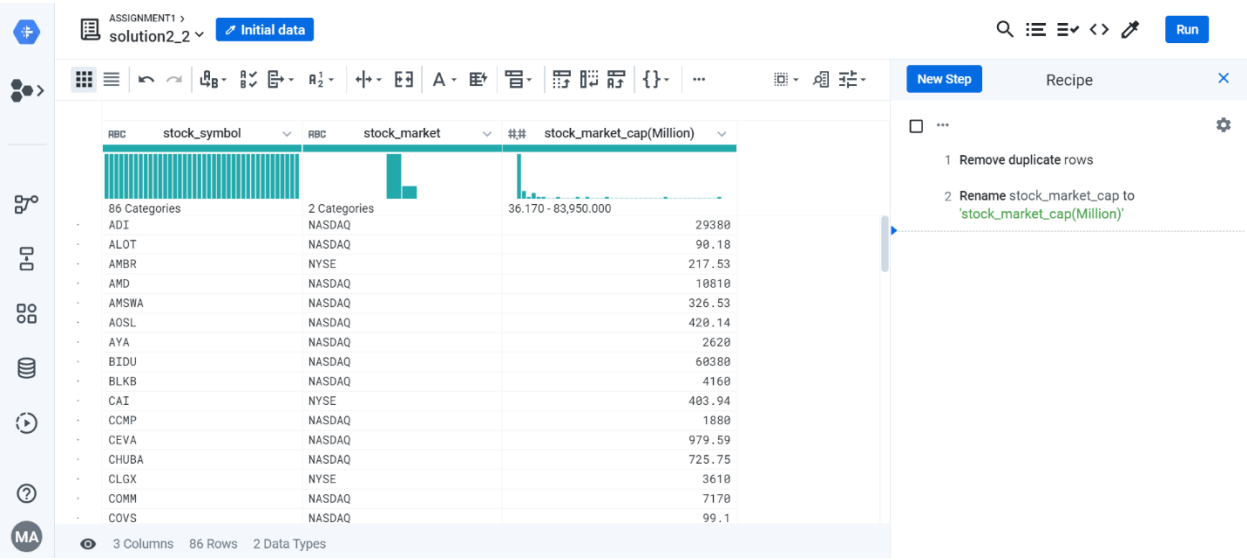


Fig – Data after eliminating duplicate values

Row count for Finance data→ **109**

Row count for Technology data→ **86**

Row count for Health Care data → **86**

Data Flow diagram containing all the operations –

1. Join all the data
2. Converting billion to million
3. Eliminating n/a
4. Retaining the data greater than or equal to $30 M
5. Splitting the data Finance, Health Care and Technology data
6. Renaming the column stock_market_cap(Million) for clarity

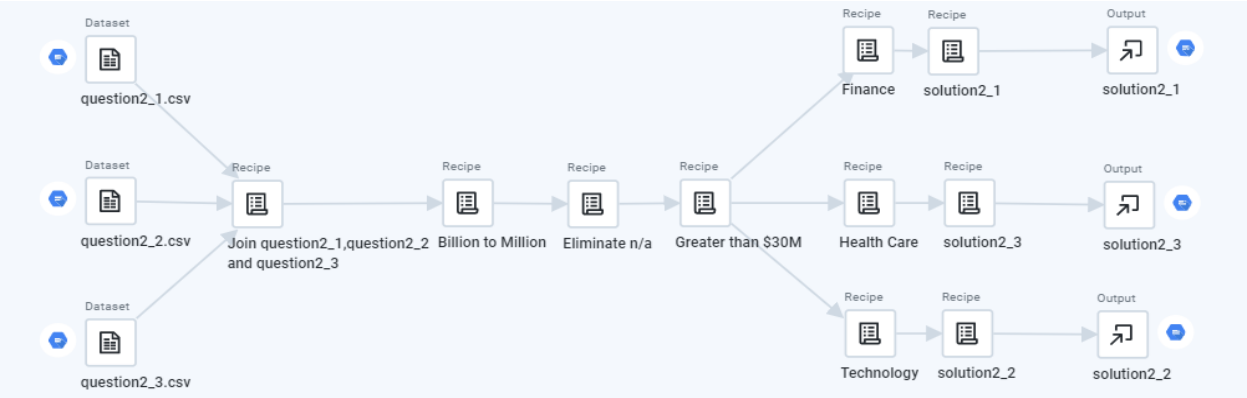


Fig – Data Flow

Task – Running job to create output csv files

Steps –

- Click on Run job
- Click on Edit to include headers

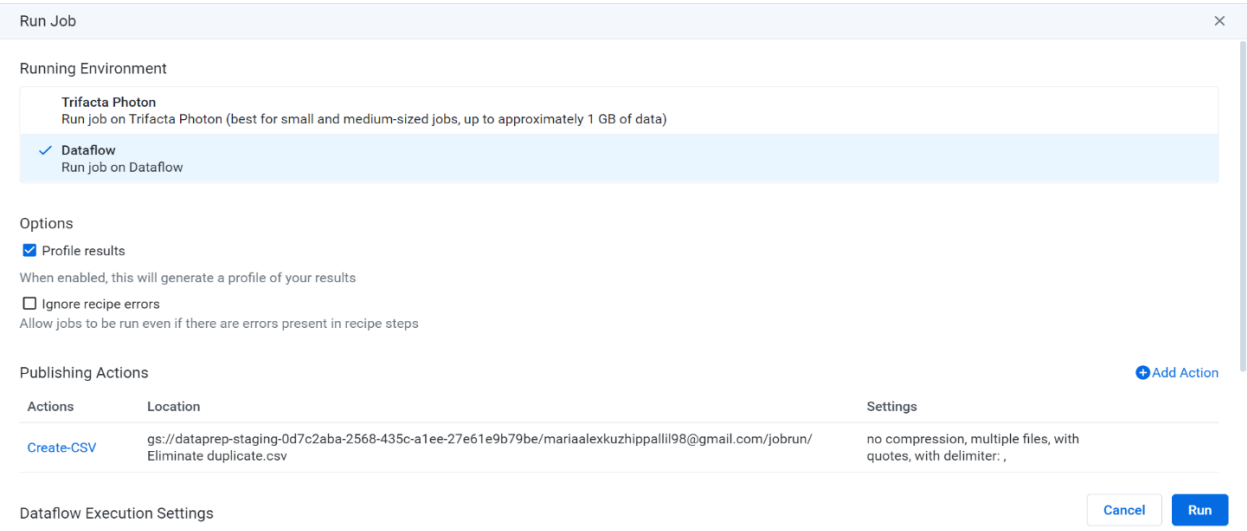


Fig – Run Job to create csv output files

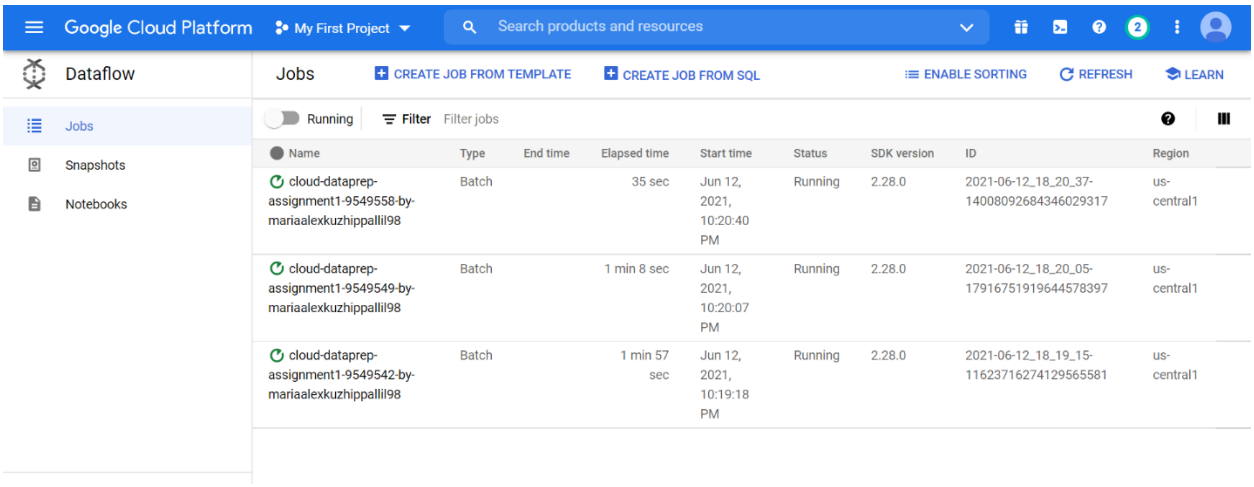


Fig – Snapshot of running jobs.

Comparison of all tools

Google DataPrep

- Great visualization compared to openrefine.

Fig - Beautiful histograms of columns showing distribution and highlights blank/null records in red (Google DataPrep).

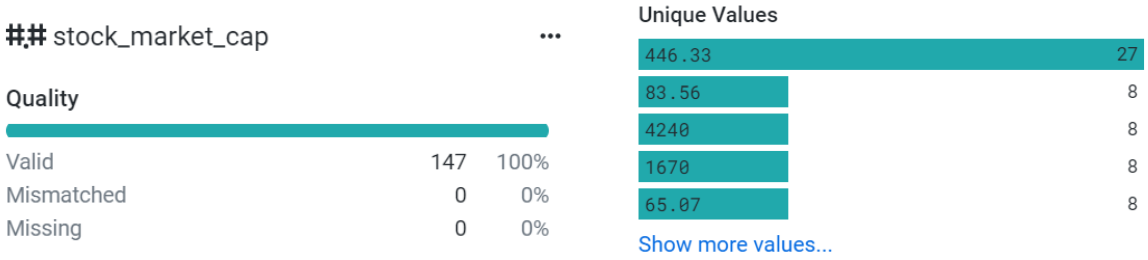


Fig - Great indicators displaying matched, missing, mismatched and unique values (Google DataPrep).

- Data flow concept makes it easier to keep track of the data manipulations performed and helps in making change in particular recipe if needed with greater ease compared to open refine which is comparatively more difficult.
- Great clarity – Clearly states all the steps carried out in details section.

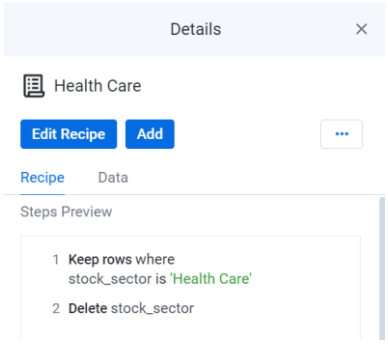


Fig - Sample representation of details section with steps preview

- Great ease of use even for novice users and interactive.

- Excellent tool for data preparation and exploratory analysis in terms of removing duplicate, deleting null/blank values, converting to different data types, renaming, and deleting columns etc.
- Quick in executing the ETL and easy to find completed, failed, and queued jobs.

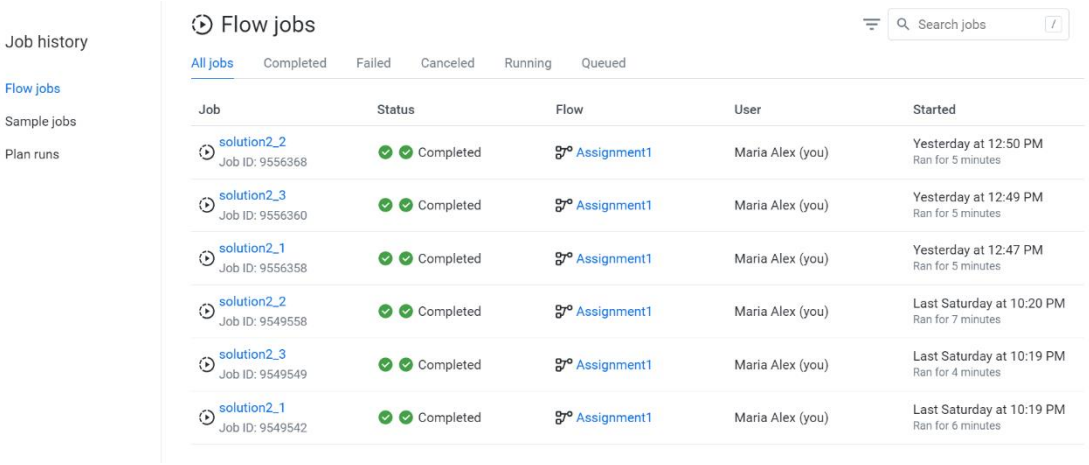


Fig - Sample representation of execution time

- Expensive after trial expires and need to supply credit card details for using the trial.

OpenRefine

- From history of already performed actions, it is possible to undo or redo.

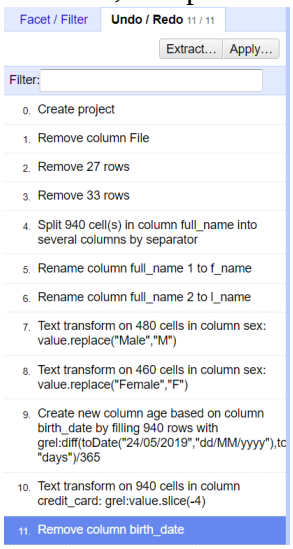
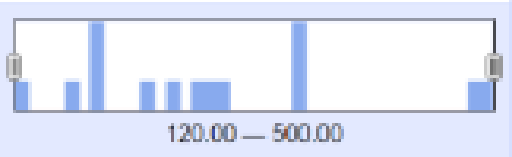


Fig – Undo / Redo for history of operations

- Faceting is an excellent feature that allows to group values. Along with it the slider, check box makes its convenient to filter date without using GREL.



Sample of filtering data using slider

- Works well with csv, json etc input and output.
- Query language GREL can be used to make.
- Free

Microsoft SSIS compared to OpenRefine and Google Dataprep

- Difficult for novice users to grasp compared to openrefine and dataprep.

(Note - From our experience we found it less convenient and lots of installation procedure hence we attempted question1 and question2).

- SSIS memory usage is high.

Submitted Files

1. assignment1.pdf – Project Report
2. Question 1
   - solution1.tar – Open refine solutions from subsection of question 1 1 - 11
   - solution1.csv - Open refine csv file from subsection of question 1 1 - 11
   - solution1_json.tar - Open refine solutions from subsection of question 1 12 - 17
   - solution1_json.csv - Open refine csv file from subsection of question 1 1 - 17
3. Question 2
   - solution2_1.csv – Finance data
   - solution2_2.csv – Technology data
   - solution2_2.csv – Health Care data
   - Dataflow_Assignment1