INFO7374 Algorithmic Digital Marketing

Summary	In this codelab, we analyzed Elo dunnhumby_The-Complete-Journey dataset to summarize the insights.
Author	Yifu Liu and Pengfei He

About Datasets

product.csv

All product are included in this csv file
Each product is unique

contains :

- PRODUCT_ID
- MANUFACTURER
- DEPARTMENT

transaction.csv

Only products with purchasing quantity are included in the dataset.
Product id are consistent between product.csv

contains:

- household_key
- PRODUCT_ID
- QUANTITY
- COUPON_DISC

coupon.csv

contains:

- COUPON_UPC
- PRODUCT_ID

- CAMPAIGN

Coupon.csv

contains:

- household_key
- Coupon
- Days
- CAMPAIGN

campaign_table.csv

Identifiers that can be used to link to other sources of movie data are contained in the file links.csv.

contains :

- CAMPAIGN
- household_key
- End date
- Description

Campaign_desc.csv

Contains:

- Start date
- End date
- CAMPAIGN
- Description

causal_data.csv

contains:

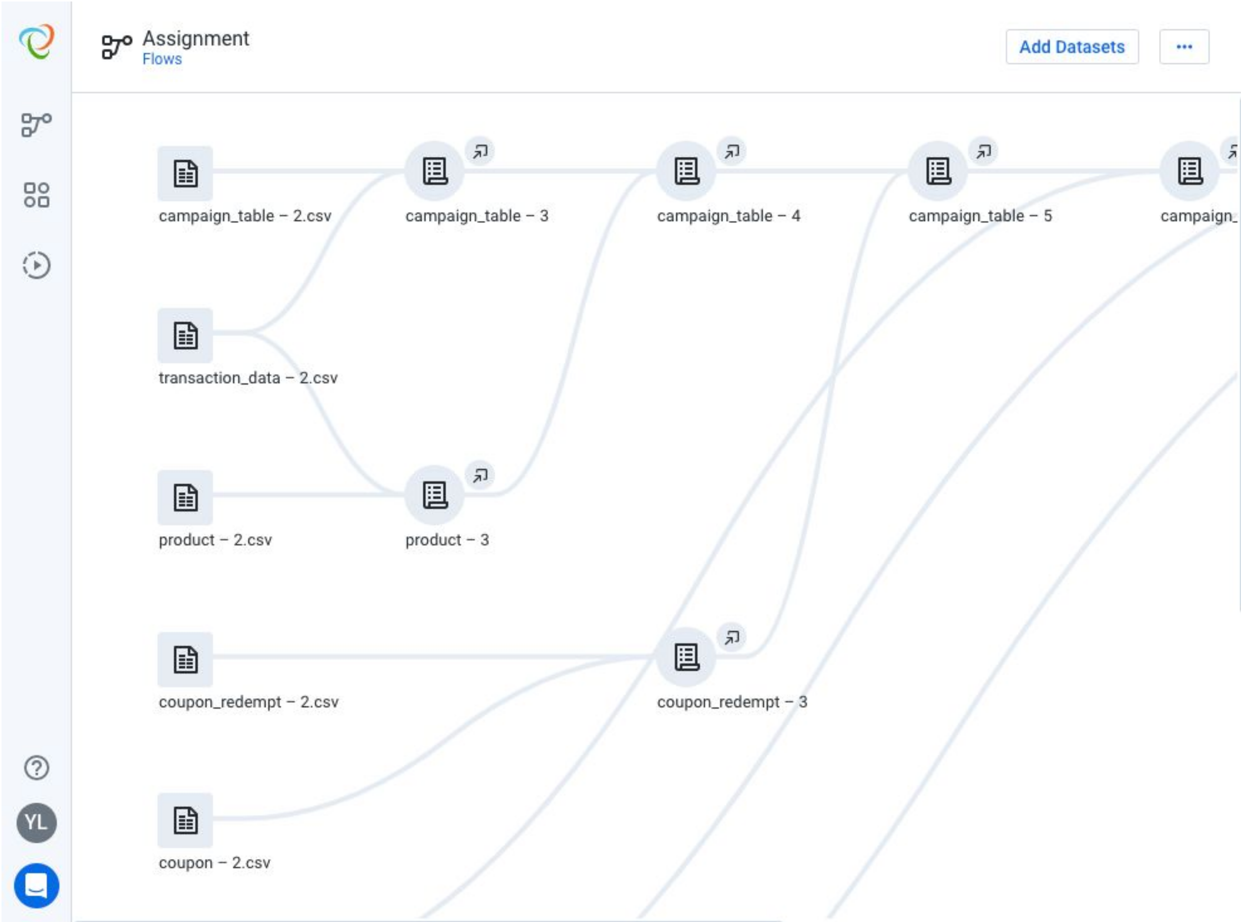
- Store id
- mailer
- display

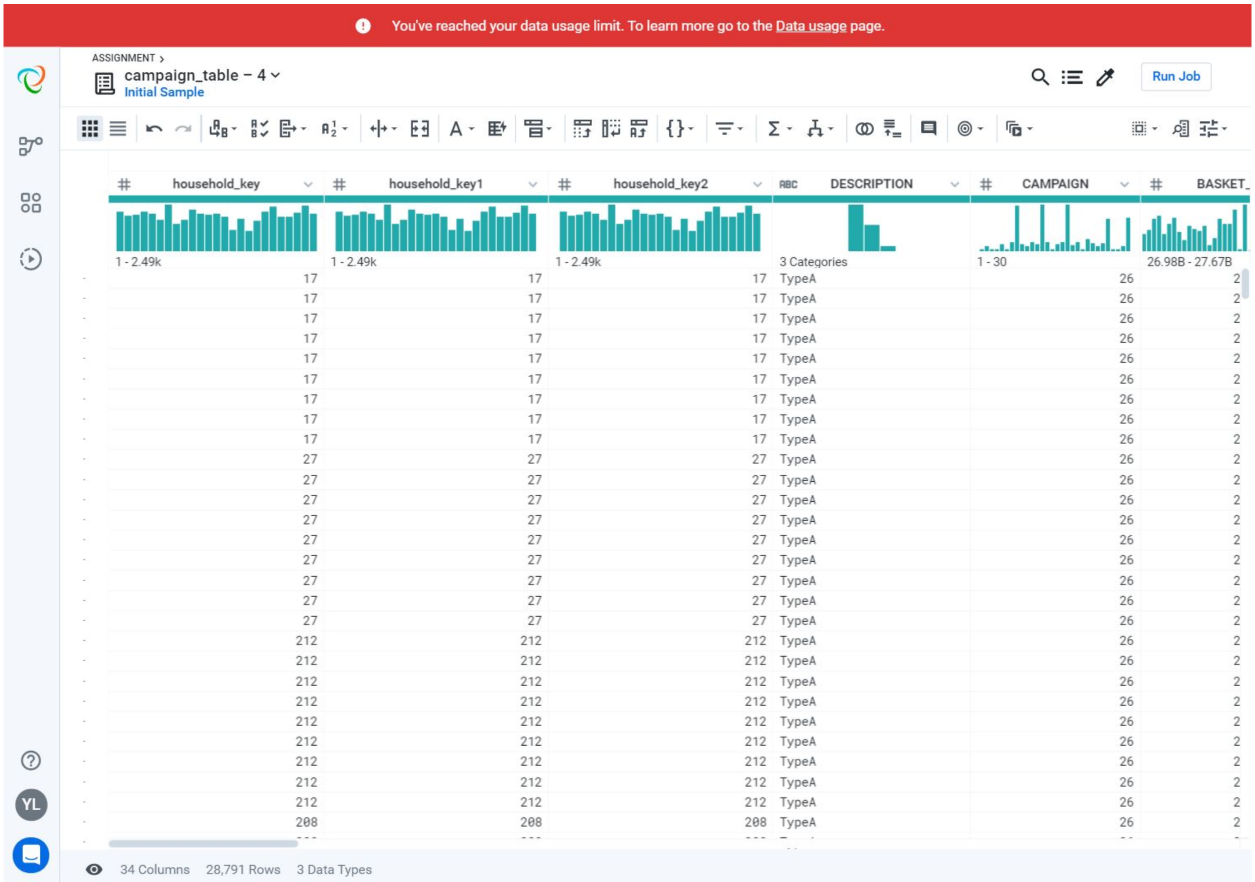
Data Wrangling

Data wrangling, sometimes referred to as **data** munging, is the process of transforming and mapping **data** from one "raw" **data** form into another format with the intent of making it more appropriate and valuable for a variety of downstream purposes such as analytics.

Trifacta

We have used Trifacta for joining the tables
And we could aggregate data and filter


Assignment
Flows
Add Datasets
campaign_table – 2.csv
campaign_table – 3
campaign_table – 4
campaign_table – 5
transaction_data – 2.csv
product – 2.csv
product – 3
coupon_redempt – 2.csv
coupon_redempt – 3
coupon – 2.csv
YL



You've reached your data usage limit. To learn more go to the Data usage page.

ASSIGNMENT >

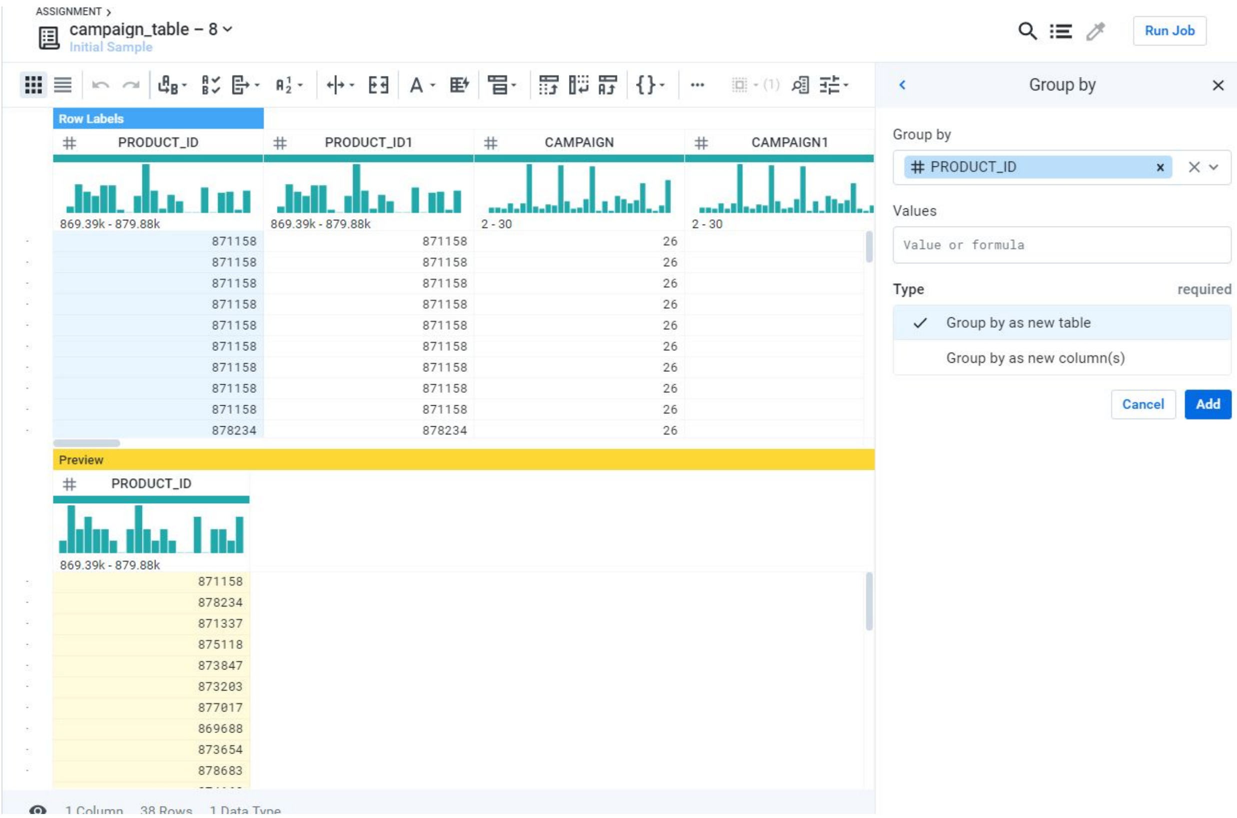
campaign_table – 4

Initial Sample

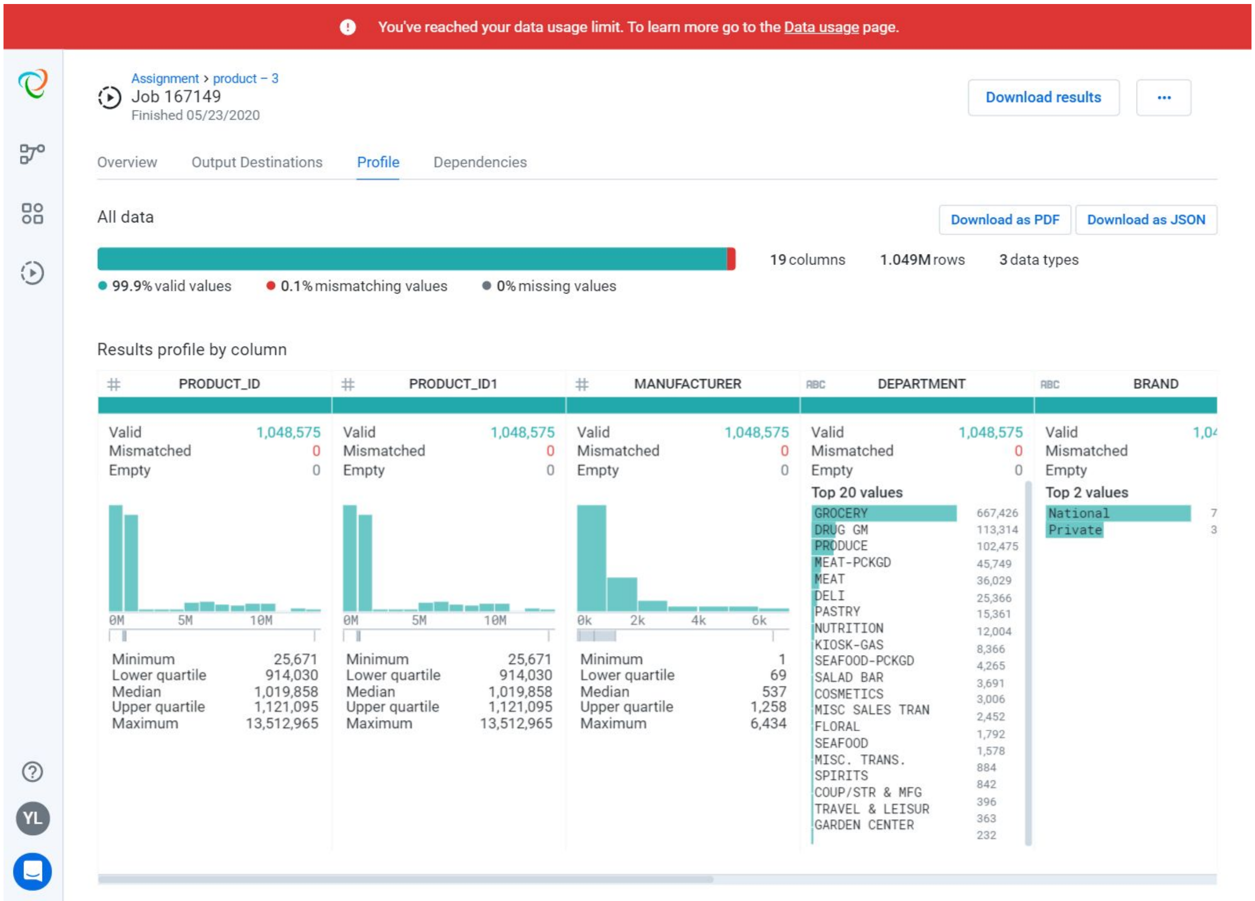
Run Job

household_key	household_key1	household_key2	DESCRIPTION	CAMPAIGN	BASKET_
1 - 2.49k	1 - 2.49k	1 - 2.49k	3 Categories	1 - 30	26.98B - 27.67B
17	17	17	TypeA	26	2
17	17	17	TypeA	26	2
17	17	17	TypeA	26	2
17	17	17	TypeA	26	2
17	17	17	TypeA	26	2
17	17	17	TypeA	26	2
17	17	17	TypeA	26	2
17	17	17	TypeA	26	2
17	17	17	TypeA	26	2
27	27	27	TypeA	26	2
27	27	27	TypeA	26	2
27	27	27	TypeA	26	2
27	27	27	TypeA	26	2
27	27	27	TypeA	26	2
27	27	27	TypeA	26	2
27	27	27	TypeA	26	2
27	27	27	TypeA	26	2
27	27	27	TypeA	26	2
212	212	212	TypeA	26	2
212	212	212	TypeA	26	2
212	212	212	TypeA	26	2
212	212	212	TypeA	26	2
212	212	212	TypeA	26	2
212	212	212	TypeA	26	2
212	212	212	TypeA	26	2
212	212	212	TypeA	26	2
212	212	212	TypeA	26	2
208	208	208	TypeA	26	2

34 Columns 28,791 Rows 3 Data Types


ASSIGNMENT >
campaign_table – 8
Initial Sample
Run Job
Row Labels
PRODUCT_ID
PRODUCT_ID1
CAMPAIGN
CAMPAIGN1
869.39k - 879.88k
869.39k - 879.88k
2 - 30
2 - 30
871158 871158 26
871158 871158 26
871158 871158 26
871158 871158 26
871158 871158 26
871158 871158 26
871158 871158 26
871158 871158 26
871158 871158 26
878234 878234 26
Preview
PRODUCT_ID
869.39k - 879.88k
871158
878234
871337
875118
873847
873203
877017
869688
873654
878683
1 Column 38 Rows 1 Data Type
Group by
Group by
PRODUCT_ID
Values
Value or formula
Type
required
Group by as new table
Group by as new column(s)
Cancel
Add

Data Integration, Profiling and Cleaning



Advantages:

- It is very handy to see all the data in all table, you can check the column easily before joining

- Trifacta is a good tool in the big data tool category of a tech stack.
- Trifacta is an open source tool with GitHub stars and GitHub forks. Here’s a link to Trifacta's open source repository on GitHub

Disadvantages:

- Dataset cant exceed 100MB
- Dataset load is lower than 100MB
- It run slowing during the job

Pandas

Data preprocessing

Missing data handling

```
[5]: transaction_data = pd.read_csv('/Users/check4068/Desktop/算法营销作业/dunnhumby_The-Complete-Journey/dunnhumby - The Compl
coupon = pd.read_csv('/Users/check4068/Desktop/算法营销作业/dunnhumby_The-Complete-Journey/dunnhumby - The Complete Journe
coupon_redempt = pd.read_csv('/Users/check4068/Desktop/算法营销作业/dunnhumby_The-Complete-Journey/dunnhumby - The Complet
campaign_table = pd.read_csv('/Users/check4068/Desktop/算法营销作业/dunnhumby_The-Complete-Journey/dunnhumby - The Complet
product = pd.read_csv('/Users/check4068/Desktop/算法营销作业/dunnhumby_The-Complete-Journey/dunnhumby - The Complete Journ
hh_demographic = pd.read_csv('/Users/check4068/Desktop/算法营销作业/dunnhumby_The-Complete-Journey/dunnhumby - The Complet
campaign_desc = pd.read_csv('/Users/check4068/Desktop/算法营销作业/dunnhumby_The-Complete-Journey/dunnhumby - The Complete
causal_data = pd.read_csv('/Users/check4068/Desktop/算法营销作业/dunnhumby_The-Complete-Journey/dunnhumby - The Complete J
```

```
[6]: campaign_table.info()
campaign_desc.info()
causal_data.info()
coupon.info()
coupon_redempt.info()
hh_demographic.info()
product.info()
transaction_data.info()
campaign_desc = campaign_desc.dropna()
campaign_table = campaign_table.dropna()
causal_data = causal_data.dropna()
coupon = coupon.dropna()
coupon_redempt = coupon_redempt.dropna()
hh_demographic = hh_demographic.dropna()
product = product.dropna()
transaction_data = transaction_data.dropna()
```

Sampling from casual data which is 600MB（over 100MB）

[25]:
```
df2.loc[df2['PRODUCT_ID']==826830]
```

[25]:

	Unnamed: 0	PRODUCT_ID	STORE_ID	WEEK_NO	display	mailer
3570016	5370012	826830	286	17	0	A
3570017	5370013	826830	286	18	0	H
3570018	5370014	826830	286	38	0	A
3570019	5370015	826830	286	39	0	A
3570020	5370016	826830	286	49	0	A
...	...	...	...	...	...	...
3571852	5371848	826830	34280	90	6	A
3571853	5371849	826830	34280	91	6	0
3571854	5371850	826830	34280	92	A	0
3571855	5371851	826830	34280	93	6	0
3571856	5371852	826830	34280	96	0	A

1841 rows × 6 columns

[24]:
```
df2.loc[df2['PRODUCT_ID']==1018588]
```

[24]:

	Unnamed: 0	PRODUCT_ID	STORE_ID	WEEK_NO	display	mailer
10614868	12414864	1018588	286	11	0	A
10614869	12414865	1018588	286	37	0	A
10614870	12414866	1018588	288	11	0	A
10614871	12414867	1018588	288	37	0	A

```
[60]: df.drop(df.index[1:7000000],inplace=True)
      df.drop(df.index[1000000:35786520],inplace=True)
```

```
[61]: len(df)
```

```
[61]: 1000000
```

```
[62]: df.to_csv('/Users/check4068/Desktop/算法营销作业/dunnhumby_The-Complete-Journey/dunnhumt

      df2=pd.read_csv('/Users/check4068/Desktop/算法营销作业/dunnhumby_The-Complete-Journey/du
      print(df2)
```

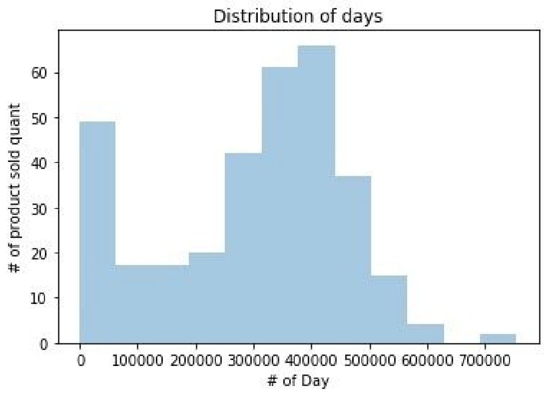
```
        Unnamed: 0  PRODUCT_ID  STORE_ID  WEEK_NO display mailer
0                0       26190       286       70       0      A
1          7000000      869077       366       41       0      A
2          7000001      869077       366       52       0      A
3          7000002      869077       366       56       0      D
4          7000003      869077       366       57       0      A
...            ...         ...       ...      ...     ...    ...
999995     7999994      897158       293       38       0      A
999996     7999995      897158       293       88       0      F
999997     7999996      897158       295       35       0      H
999998     7999997      897158       295       38       0      A
999999     7999998      897158       295       88       0      F

[1000000 rows x 6 columns]
```

We saw the product id between 800000 and 1000000 is very frequent, which is good for joining, matching more % while joining with other tables
So we sample the data between 800000 and 1000000

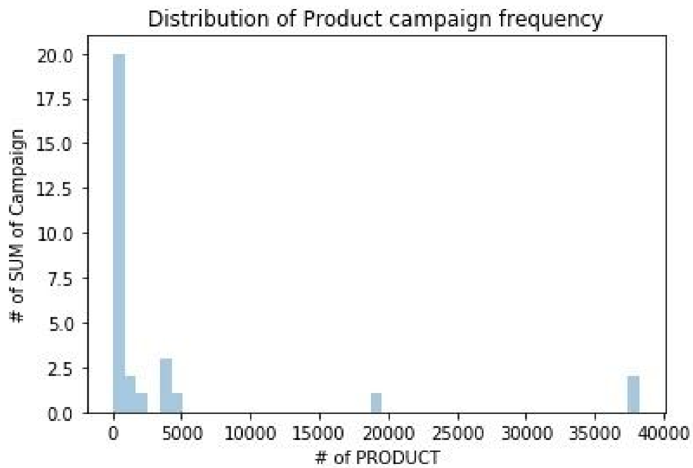
```
memory usage: 30.0 MB
```

```
[155]: a1 = transaction_data.groupby(by=['DAY']).agg({'QUANTITY':'sum'})
a1.head()
p1 = sns.distplot(a1, kde=False, hist=True)
p1.set(title='Distribution of days',
       xlabel='# of Day',
       ylabel='# of product sold quant');
```



We use groupby aggregation function to see the trend of days

```
[150]: a3 = coupon.groupby(by=['CAMPAIGN']).agg({'PRODUCT_ID':'count'})
a3.head()
p3 = sns.distplot(a3, kde=False, hist=True)
p3.set(title='Distribution of Product campaign frequency',
       xlabel='# of PRODUCT',
       ylabel='# of SUM of Campaign');
```



We use groupby aggregation function to see the product campaign frequency

```
[131]: merge1 = coupon.merge(coupon_redempt, on=['COUPON_UPC'], how='inner')
merge1.head()
```

[131]:

	COUPON_UPC	PRODUCT_ID	CAMPAIGN_x	household_key	DAY	CAMPAIGN_y
0	10000089064	27754	9	321	446	9
1	10000089064	27754	9	1773	439	9
2	10000089064	243186	9	321	446	9
3	10000089064	243186	9	1773	439	9
4	10000089064	872316	9	321	446	9

We merge two table in pandas, but we need to find out what we should join on

Advantages:

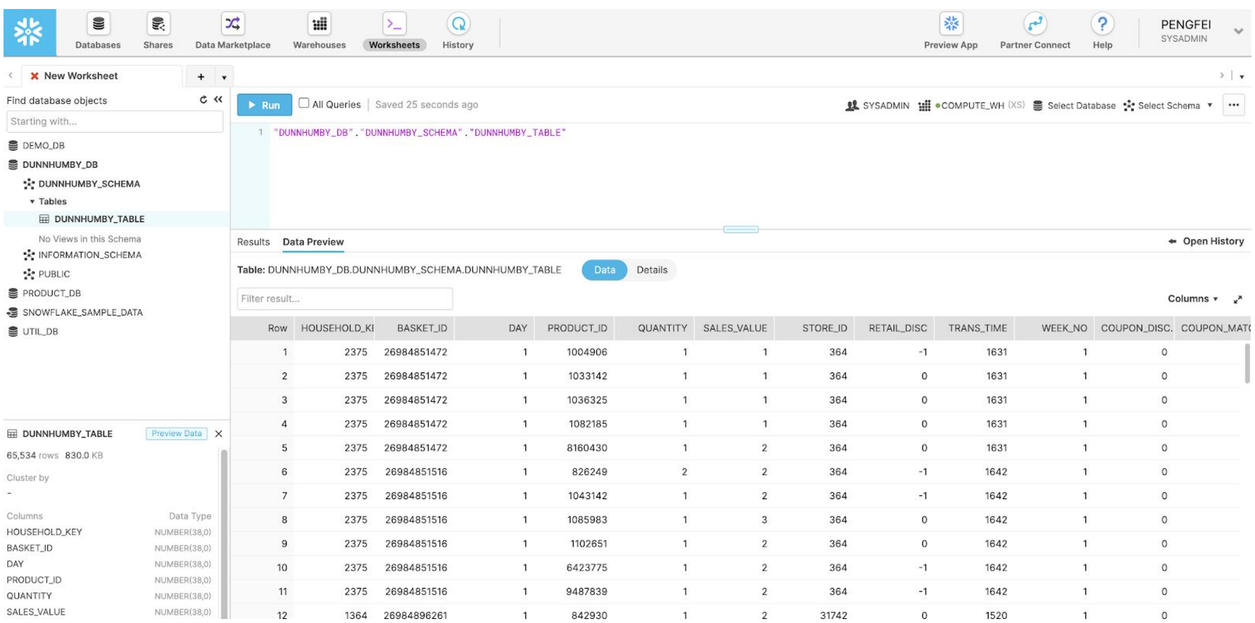
- It is good for data preprecessing
- It is handy to plot a small part of data

Disadvantages:

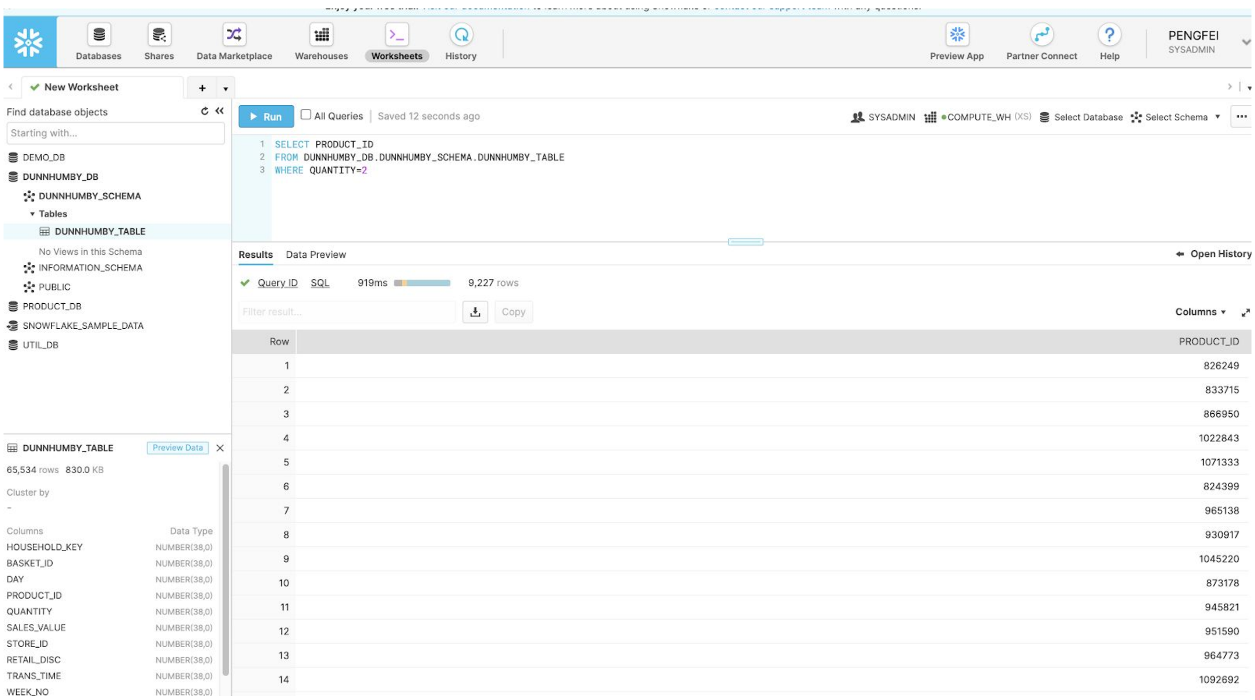
- We need to find out what we should join on during table merging
- Not as handy as trifacta when mutli-table are involved, because it is hard to find out the right coloums in so many coloums

Snowflake and EA

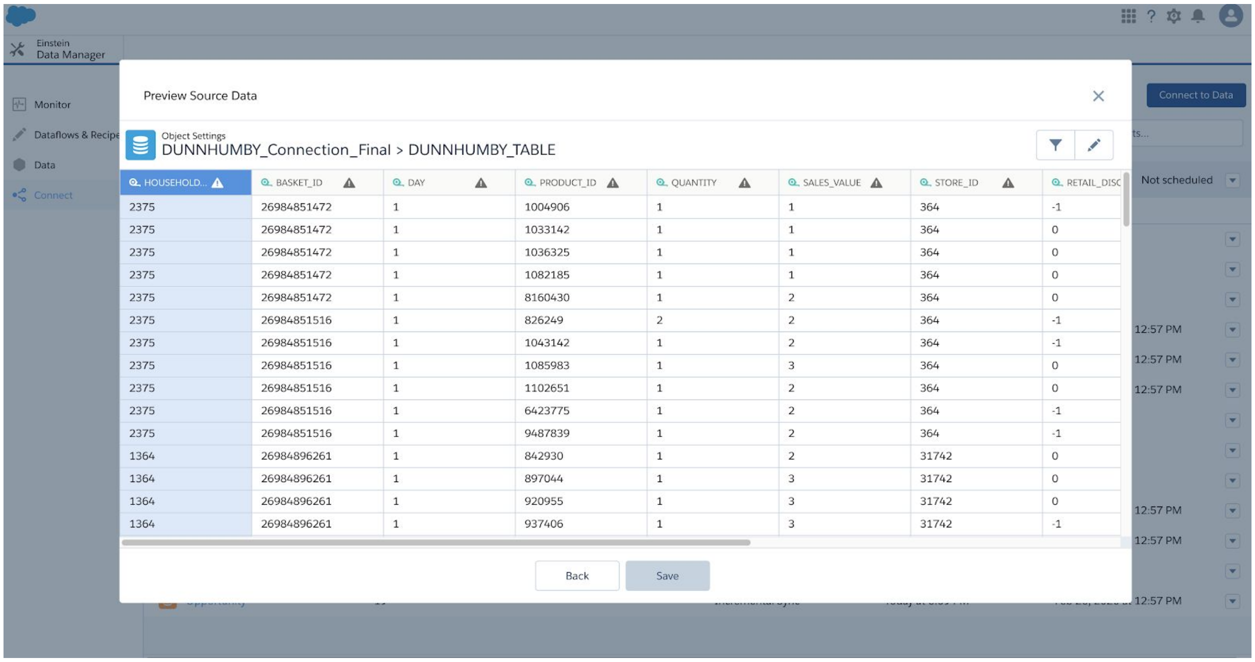
The data is imported into Snowflake, custom warehouse, schema and table is created.



Create custom SQL query to get information



Connected to EA



Advantages:

- User-friendly UI, especially for large dataset
- Strongly computing capabilities in handling huge datas

Disadvantages:

- Dataset has a limitation, so we have to sample the data to use it.
- The free-trial has time-limit.

Questions to Answer

1 Which columns are dimensions, which columns are measures?

Dimensions are columns like: Department, Brand, Commodity_Desc, Sub-Commodity_Desc

Measures are columns like: Curr_size_of_product, Sales_value, Trans_Time, Retail_Disc, Retail_disc, Trans_time,

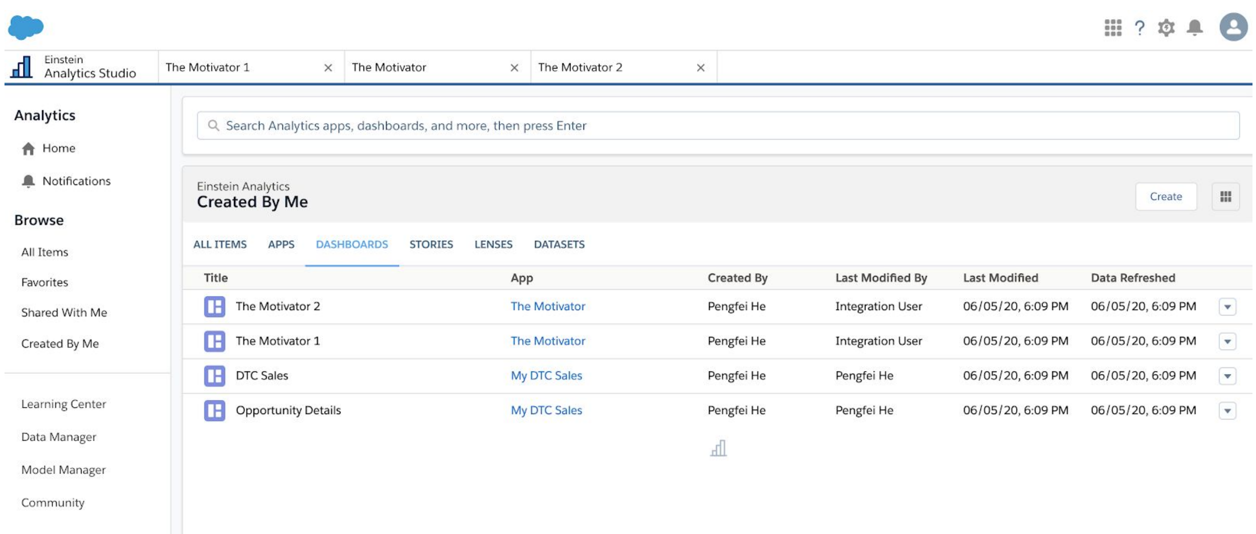
Columns we choose to drop(missing values or null): Coupon_disc, Coupon_match_disc,

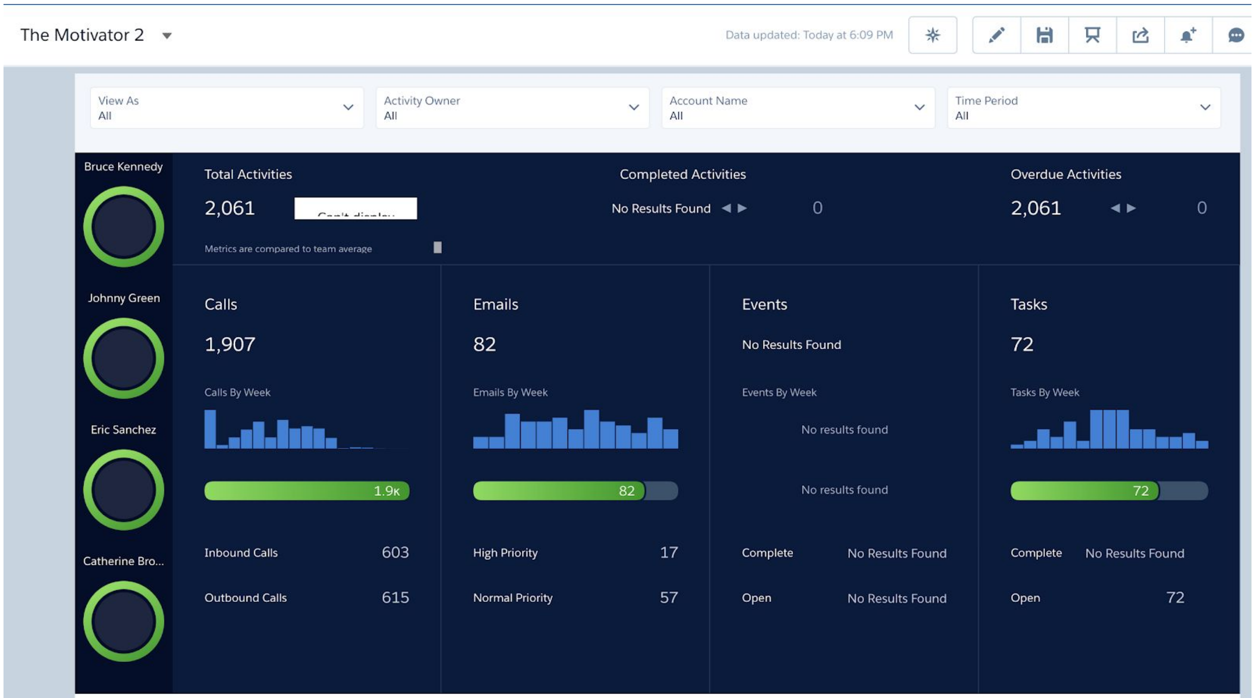
2 How would you generate new dimensions? What will you do to summarize measures?

We mainly use map with lambda expressions to generate new dimensions and use built-in methods from pandas to compute the measures.
For example, we will use like:
table[new dimension] = table[old dimension].map(lambda)
Table_mean = table[measure].mean()

3 Dashboards:

We face some problems in saving the object to EA after connecting. So we used a sample dataset to demonstrate how to implement a dashboard.




The Motivator 2
Data updated: Today at 6:09 PM
View As
All
Activity Owner
All
Account Name
All
Time Period
All
Bruce Kennedy
Johnny Green
Eric Sanchez
Catherine Bro...
Total Activities
2,061
Metrics are compared to team average
Completed Activities
No Results Found
0
Overdue Activities
2,061
0
Calls
1,907
Calls By Week
1.9K
Inbound Calls
603
Outbound Calls
615
Emails
82
Emails By Week
82
High Priority
17
Normal Priority
57
Events
No Results Found
Events By Week
No results found
No results found
Complete
No Results Found
Open
No Results Found
Tasks
72
Tasks By Week
72
Complete
No Results Found
Open
72