SECTION 1: DATA LOADING AND SEPARATION (using AWS- EMR and AWS - S3):

1) Given the 2 files.
   - Events.csv
   - Users.csv

2) I noticed that the files files were relatively large and I thought that in the practical scenario the files would be much larger.

3) I set up a 3 node hadoop cluster on AWS to group and analyse the data.

I uploaded the files to Amazon S3 and created external tables to reference the files

<
```
drop table whisper_events;
CREATE EXTERNAL TABLE whisper_events(test_cohort STRING, event_name STRING, user_id
STRING, whisper_id STRING, extra_information STRING, time_generated STRING)
ROW FORMAT DELIMITED FIELDS TERMINATED BY ','
STORED AS TEXTFILE LOCATION 's3n://whisper_test/events';
msck repair table whisper_events;


drop table whisper_users;
CREATE EXTERNAL TABLE whisper_users(user_id STRING, dtype_appversion STRING, ts_created
STRING)
ROW FORMAT DELIMITED FIELDS TERMINATED BY ','
STORED AS TEXTFILE LOCATION 's3n://whisper_test/users';
msck repair table whisper_users;
```
>

4) I quickly ran a group by clause on the events.csv file on *test_cohort* and identified 6 test groups. [A ... F]

<
```
Select test_cohort from whisper_events group by test_cohort;
```
>

5) The next step would be to perform an **inner join** on users.csv file with the events.csv file on user_id for each separate test_cohort and write the results back to HDFS.

<
```
create external table test_cohort_A(test_cohort STRING, event_name STRING, user_id STRING,
                                    whisper_id STRING, extra_information STRING, time_generated STRING,
dtype_appversion STRING, ts_created STRING
                                   )
row format delimited fields terminated by ',' lines terminated by '\n'
STORED AS TEXTFILE LOCATION 's3n://whisper_test/events_test_cohorts/A';
```

INSERT OVERWRITE TABLE test_cohort_A select e.test_cohort, e.event_name, e.user_id, e.whisper_id, e.extra_information, e.time_generated, u.dtype_appversion, u.ts_created from whisper_events e, whisper_users u where e.user_id = u.user_id and e.test_cohort like '%A%';

‘”SAME STATEMENT WOULD REPEAT FOR B, C, D, E, F ”’

>

We would not have 6 tables created and the files would be at the HDFS location.

I can now easily download these files from S3 using boto for python and run further analysis and enrichments.

SECTION 2: DATA CLEANING AND ENRICHMENT (USING PYTHON AND PANDAS)

1) Convert the events.csv time_generated field to date_time format using pandas.
   pd.to_datetime(v['time_generated'], unit='ms') (Time in millisecond)
2) Convert the users.csv ts_created field to date_time format using pandas.
   pd.to_datetime(user_df['ts_created'], unit='us') (Time in microsecond)

Code Snippet: Function to read csv and enrich it with the conversion to date time using pandas:

<

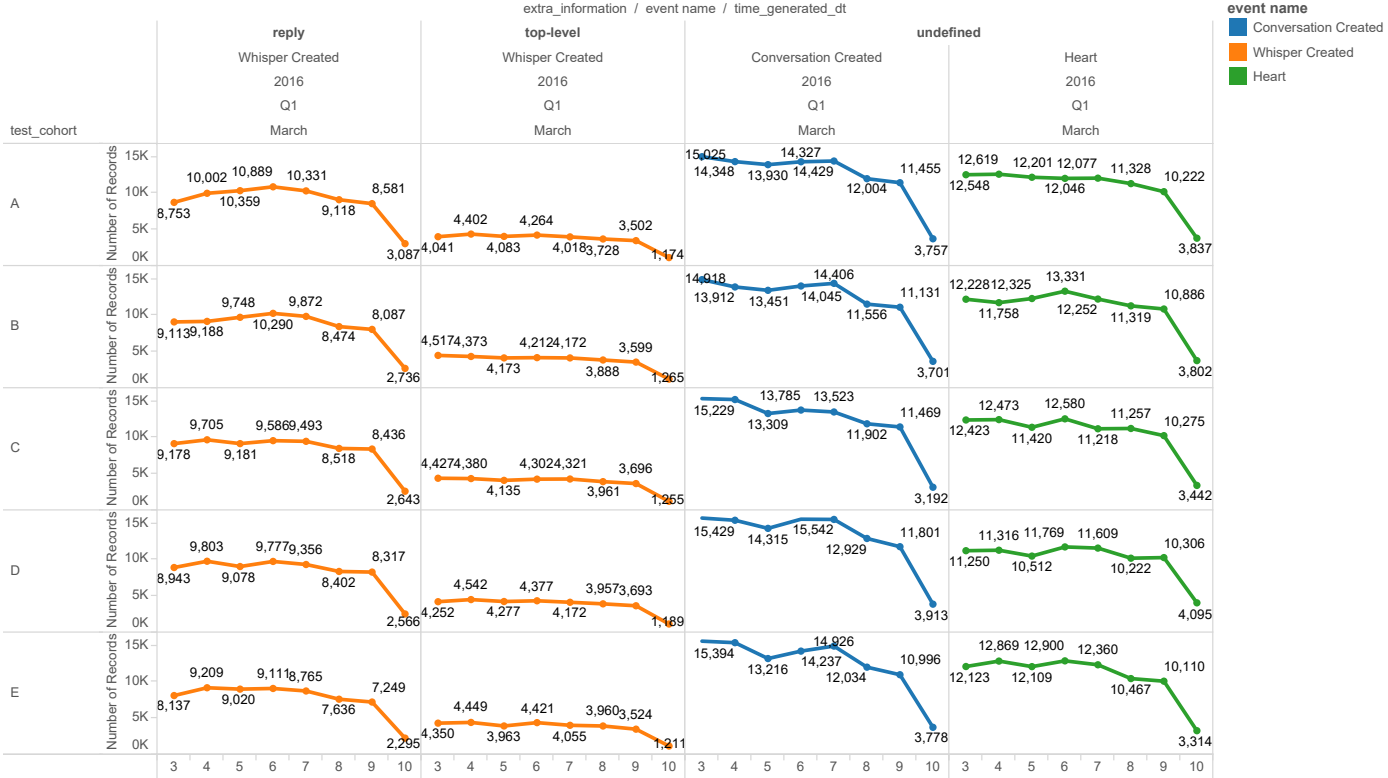
```
Import pandas as pd

def convert_test_cohort_to_dt(file_path_list):
  for file_path in file_path_list:
    df = pd.read_csv(file_path, names=['test_cohort', 'event name', 'user_id', 'whisper_id', 'extra_information',    'time_generated', 'dtype_appversion', 'ts_created'], encoding='utf-8')
    t_generated_dt_df = pd.DataFrame(pd.to_datetime(df['time_generated'], unit='ms'))
    ts_created_dt_df = pd.DataFrame(pd.to_datetime(df['ts_created'], unit='us'))
    t_generated_dt_df = t_generated_dt_df.rename(columns={'time_generated': 'time_generated_dt'})
    ts_created_dt_df = ts_created_dt_df.rename(columns={'ts_created': 'ts_created_dt'})
    df = pd.concat([df, t_generated_dt_df, ts_created_dt_df], axis=1, join_axes=[df.index])
    df[['test_cohort', 'event name', 'user_id', 'whisper_id', 'extra_information', 'time_generated', 'time_generated_dt', 'dtype_appversion', 'ts_created', 'ts_created_dt']].to_csv(file_path+".csv", index=False)
```

>

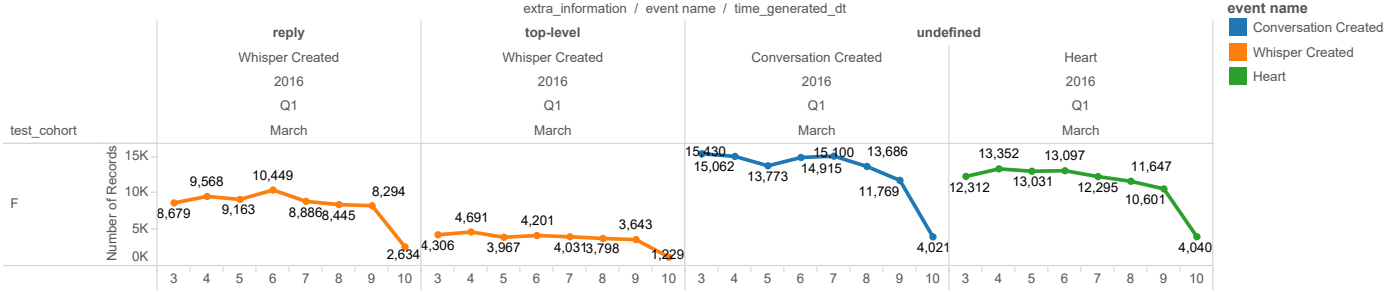
SECTION 3: RUNNING ANALYSIS USING TABLEAU

test_cohorts



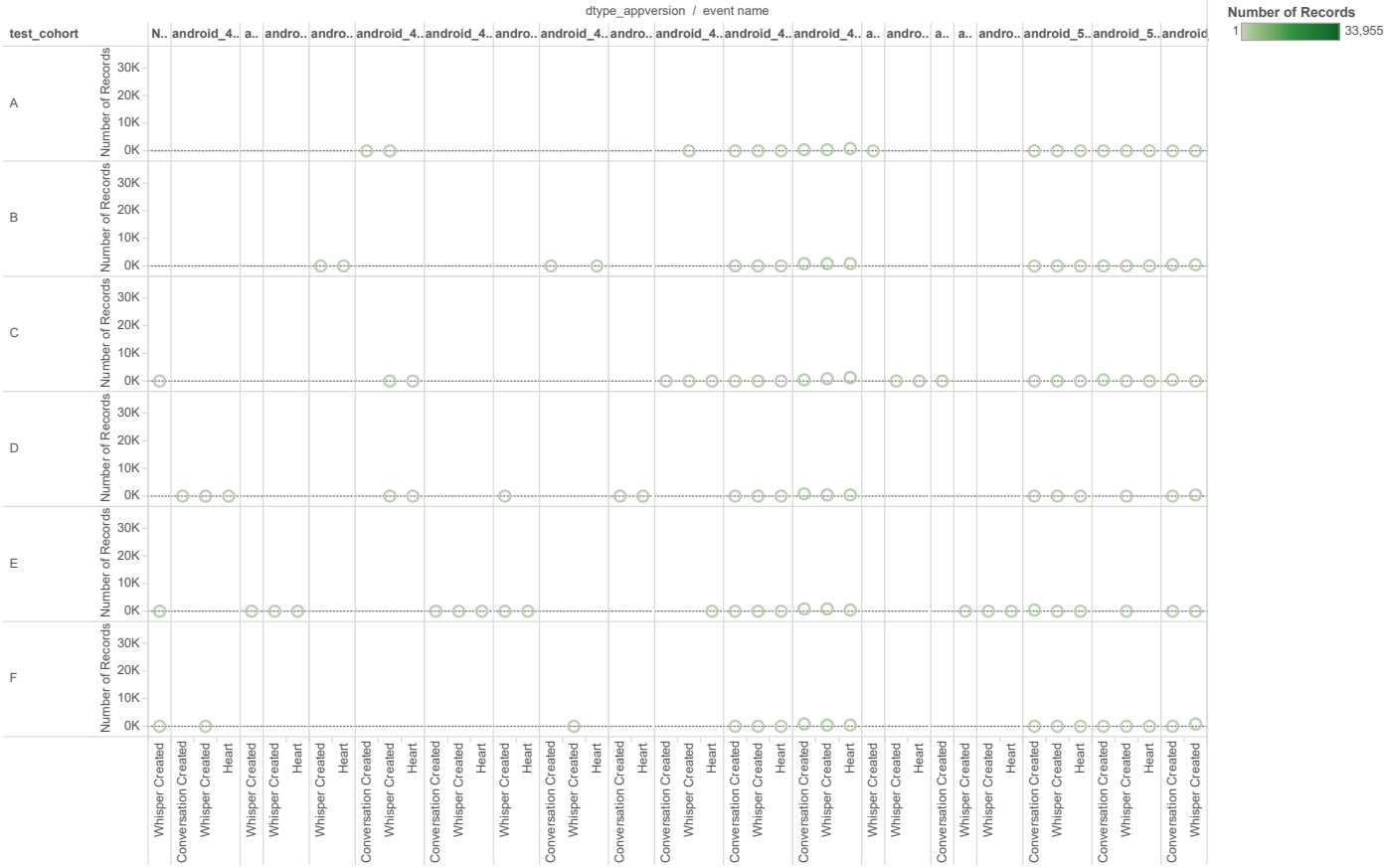
The trend of sum of Number of Records for time_generated_dt Day broken down by extra_information, event name, time_generated_dt Year, time_generated_dt Quarter and time_generated_dt Month vs. test_cohort. Color shows details about event name. The marks are labeled by sum of Number of Records. The view is filtered on sum of Number of Records, which ranges from 1,091 to 33,955.

test_cohorts



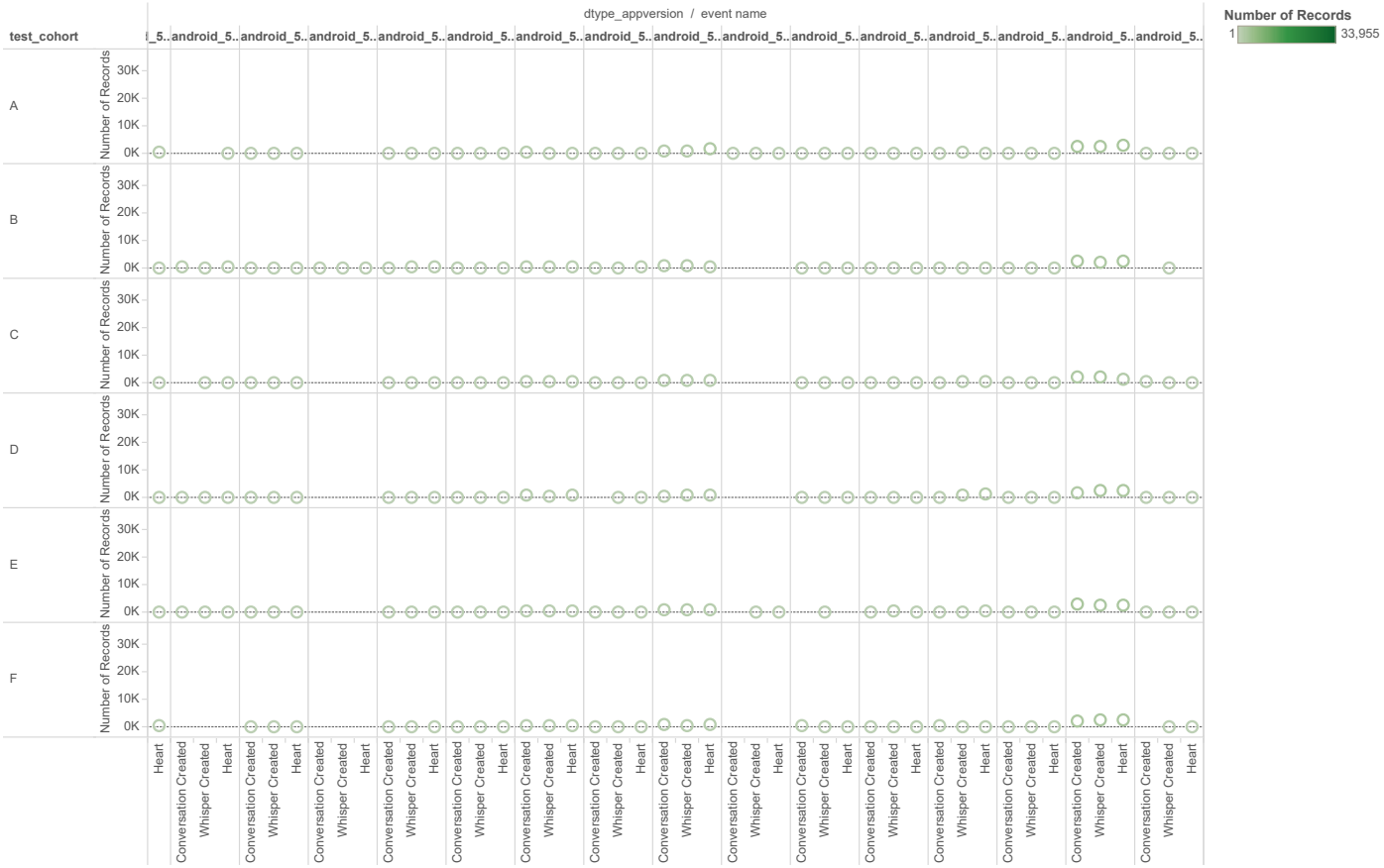
The trend of sum of Number of Records for time_generated_dt Day broken down by extra_information, event name, time_generated_dt Year, time_generated_dt Quarter and time_generated_dt Month vs. test_cohort. Color shows details about event name. The marks are labeled by sum of Number of Records. The view is filtered on sum of Number of Records, which ranges from 1,091 to 33,955.

test_cohorts



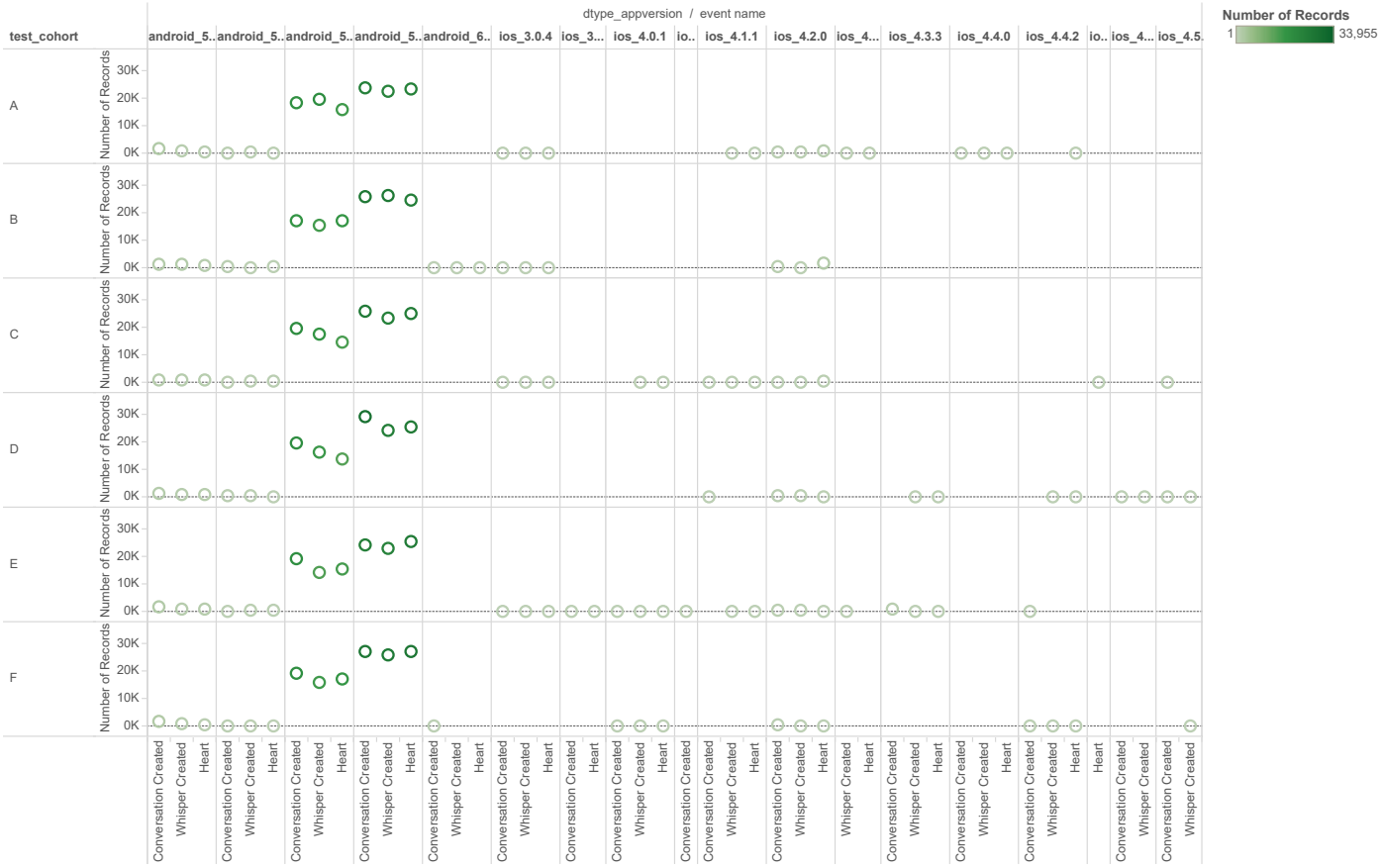
Sum of Number of Records for each event name broken down by dtype_appversion vs. test_cohort. Color shows sum of Number of Records.

test_cohorts



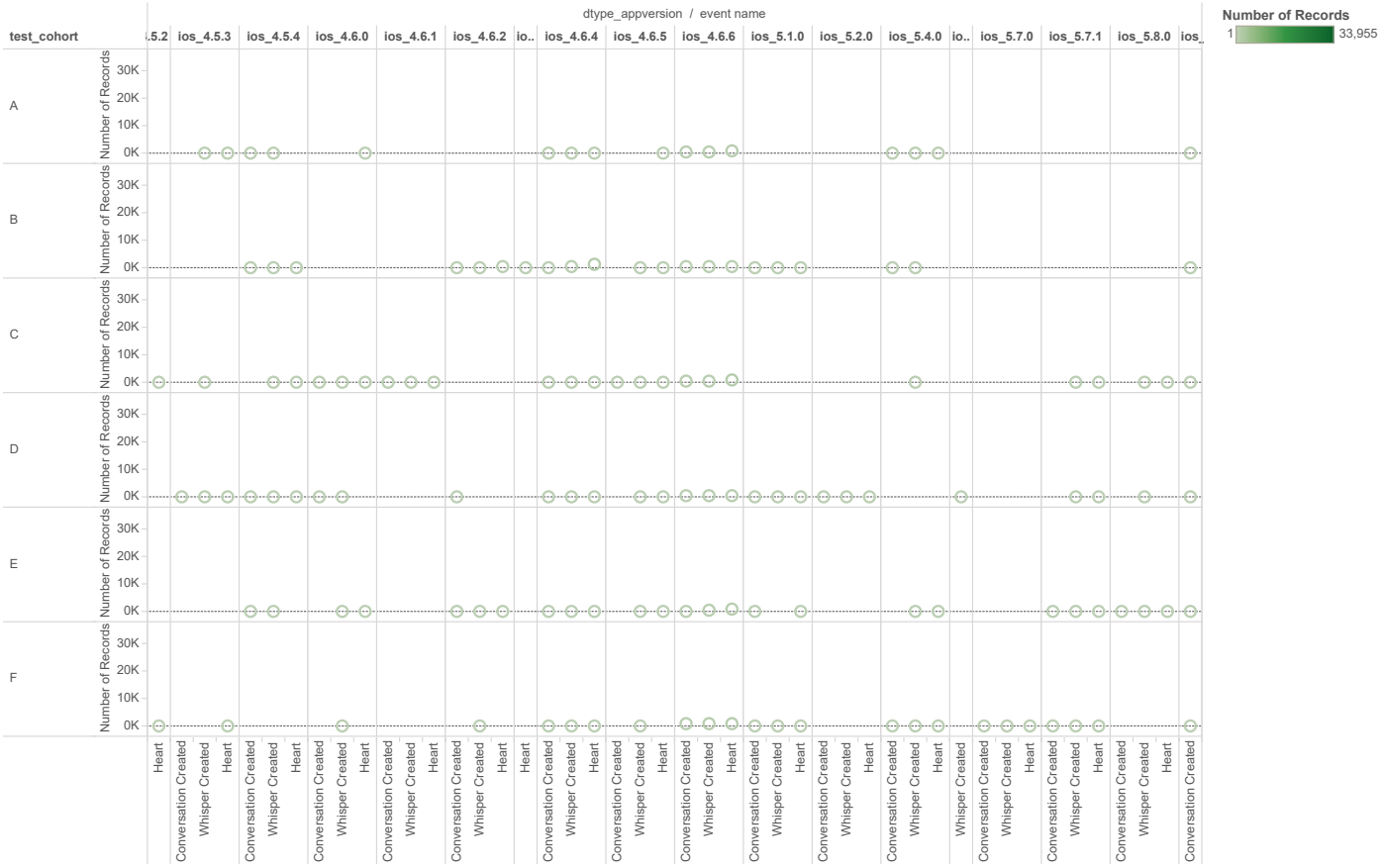
Sum of Number of Records for each event name broken down by dtype_appversion vs. test_cohort. Color shows sum of Number of Records.

test_cohorts



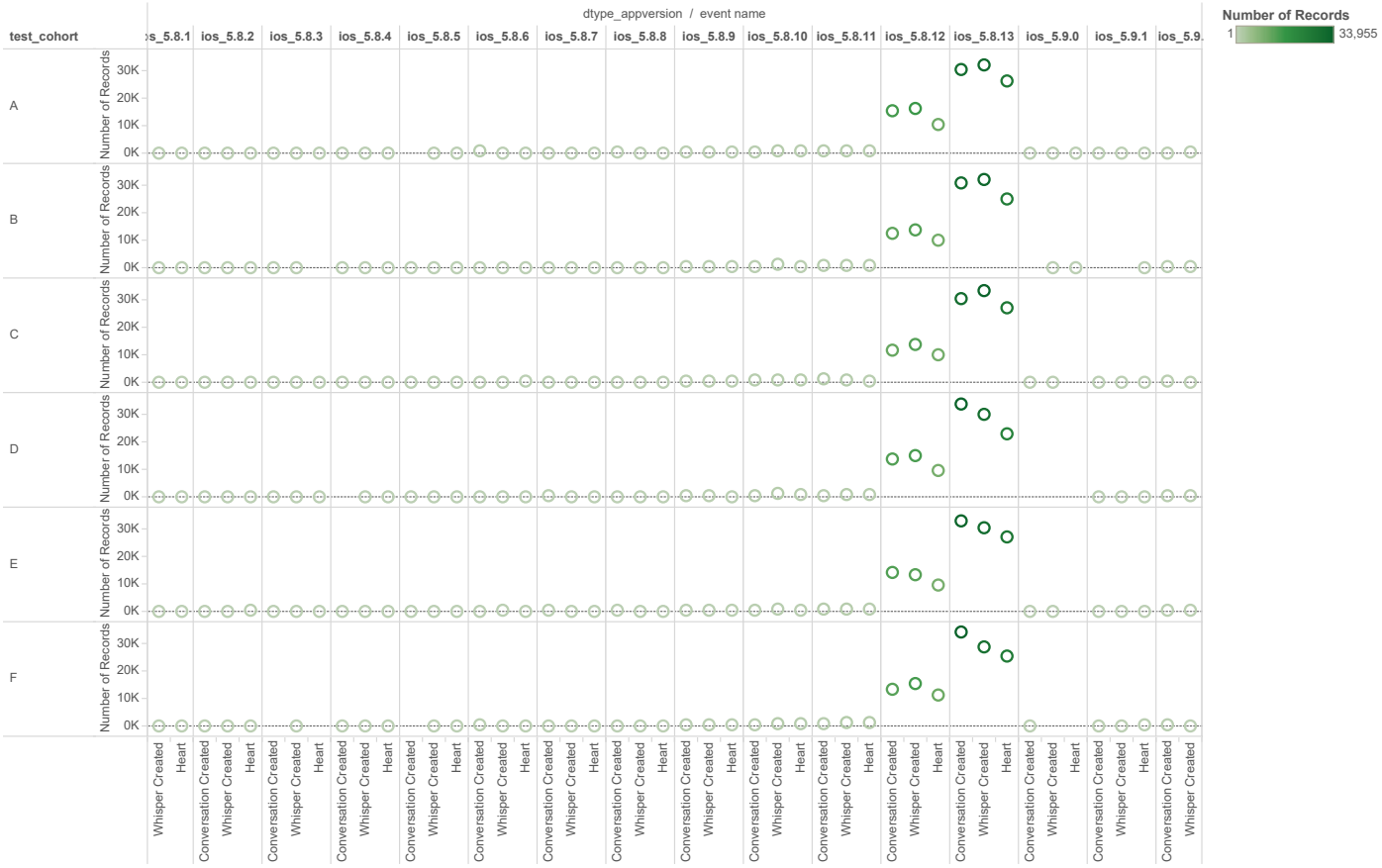
Sum of Number of Records for each event name broken down by dtype_appversion vs. test_cohort. Color shows sum of Number of Records.

test_cohorts



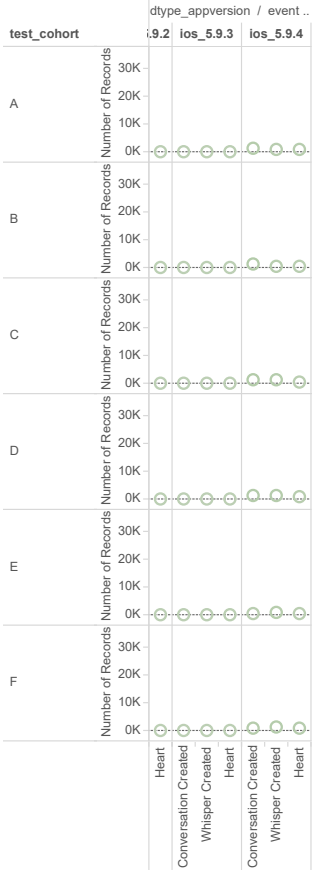
Sum of Number of Records for each event name broken down by dtype_appversion vs. test_cohort. Color shows sum of Number of Records.

test_cohorts



Sum of Number of Records for each event name broken down by dtype_appversion vs. test_cohort. Color shows sum of Number of Records.

test_cohorts



Sum of Number of Records for each event name broken down by dtype_appversion vs. test_cohort. Color shows sum of Number of Records.

SECTION 4: CONCLUSION

1) After running a very detailed analysis on the data and event logs. I can see a very similar pattern in each of the cohort.
2) **Figure 1**: you can see that on the 4th of March the number of whisper replies created increases across all of the cohorts this could be because of the spike in interest because of the change in presentation. I.e. it caused more users to reply to the existing whispers. It is also interesting to note that the number of conversations created also had a downward trend from the 4th till the 5th.
3) But each cohort had a similar trend. Which is good that the change affected each cohort group in a similar manner.
4) **Figure 2:** I wanted to check how the change affected users across devices. Since the logs only contained data from March 3rd till March 11th. We need to check previos logs and see if there is any existing users were affected.
5) Focusing on the major groups IOS-5.8.13 noticed a higher whisper creation rate for cohort A, B, and C. While the number of conversations were high across all the cohorts in this group.
6) **Figure 3** shown below is optional and gives an hourly trend based on each event_name. It can be a longer and more detailed trend analysis and redundant. But I've always found some people more interested in looking at the subtle differences in each test cohort.
7) The most fascinating point is that on the 3rd hour of every day we see a spike in the whispers created, while on the 9th hour there is a minima.

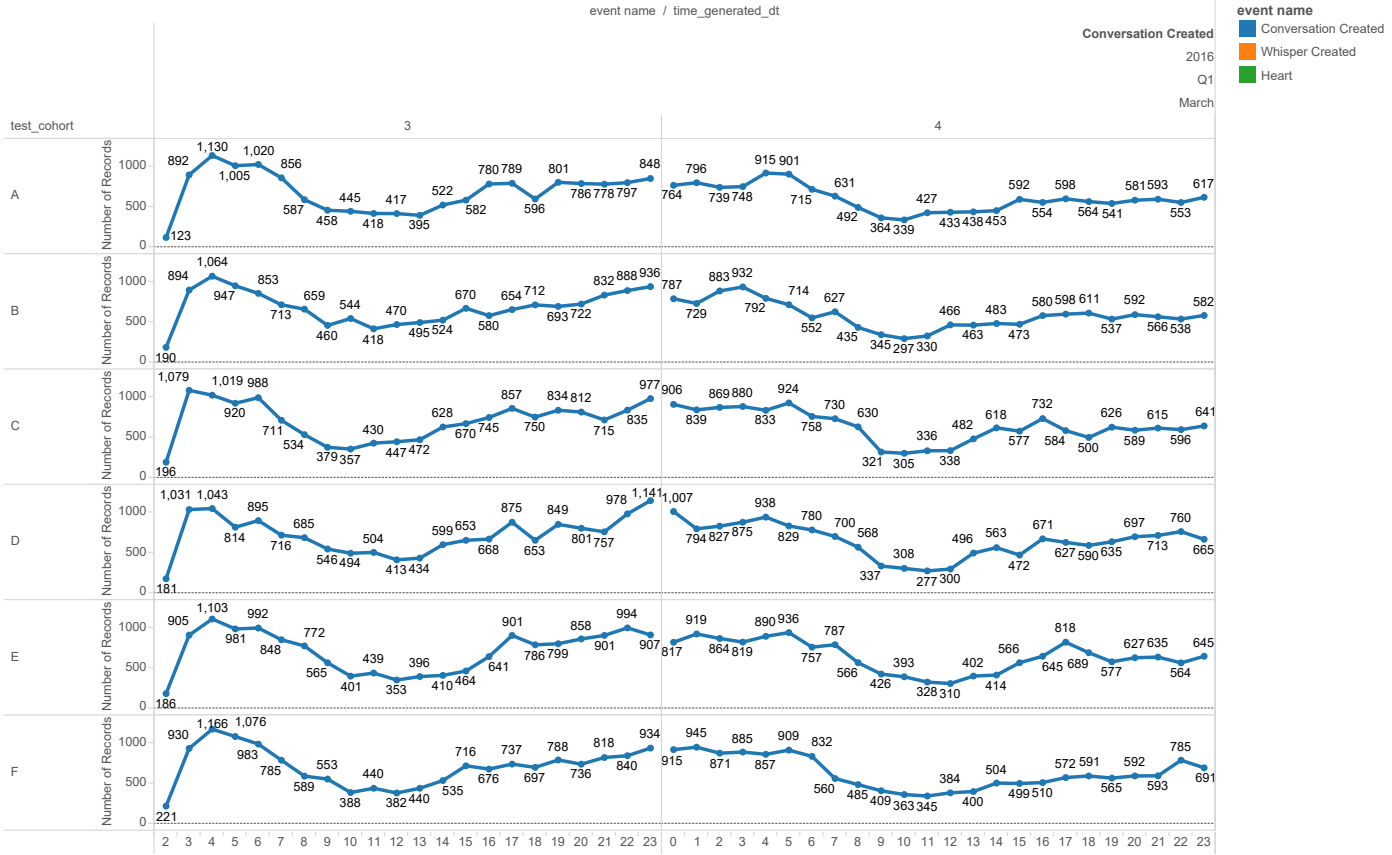
Technology choices:

1. A very similar analysis could be produced using R or matlab. I preferred Tableau because I own a licensed version and prefer its strong data visualization properties.
2. I enjoy using Python as a scripting language and with data engineering packages like pandas and with the anaconda distro. It just becomes very useable.
3. I Used HIVE on AWS to store and partition the data. I always store the data on S3 and create an external table reference from HIVE to the location. In our case we had no partitioned data but if the data was client partitioned or date partitioned its much easier to reference all the data.

THANK YOU,
OLIVER LEWIS.

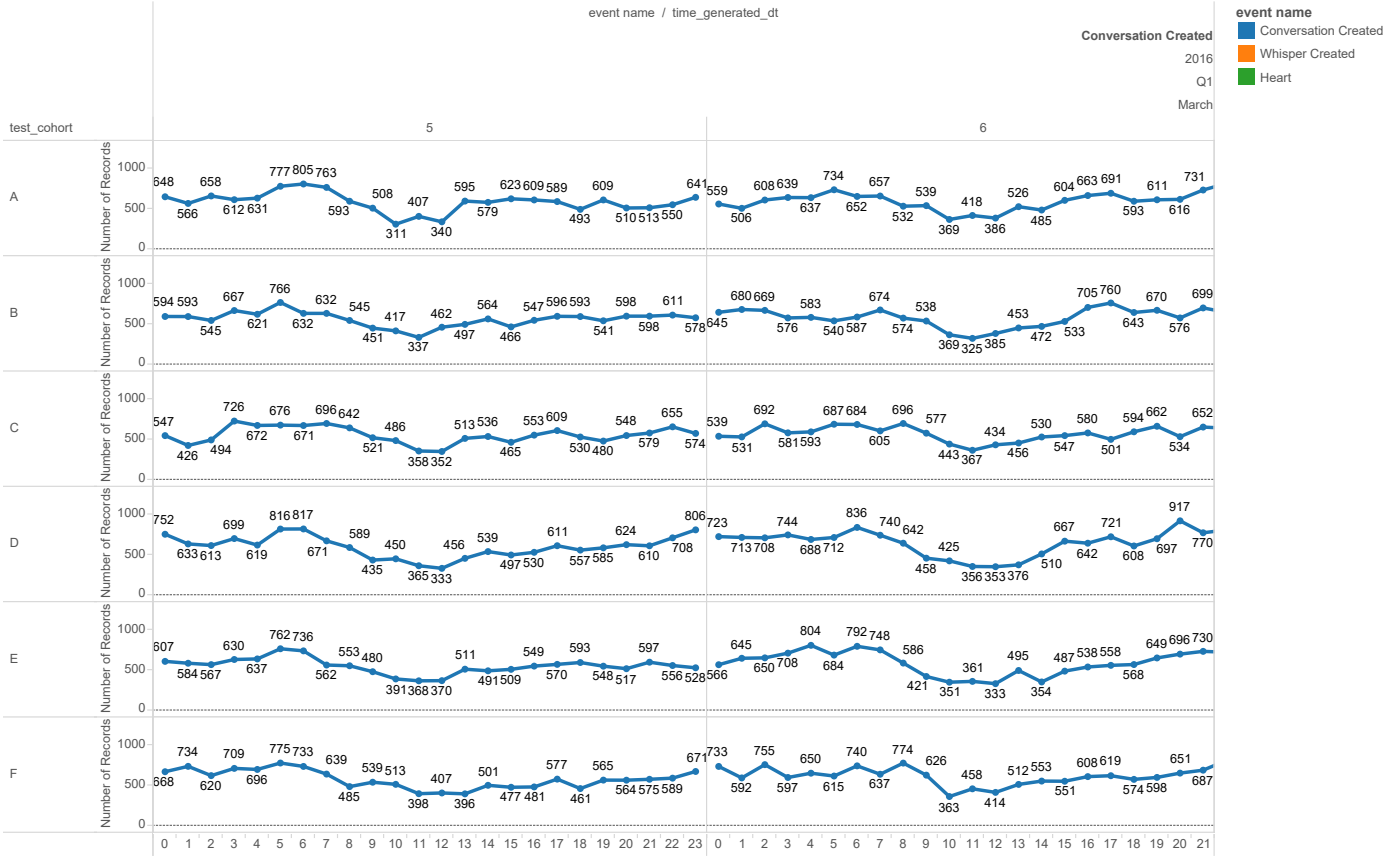
FIGURE 3 BELOW:

test_cohorts



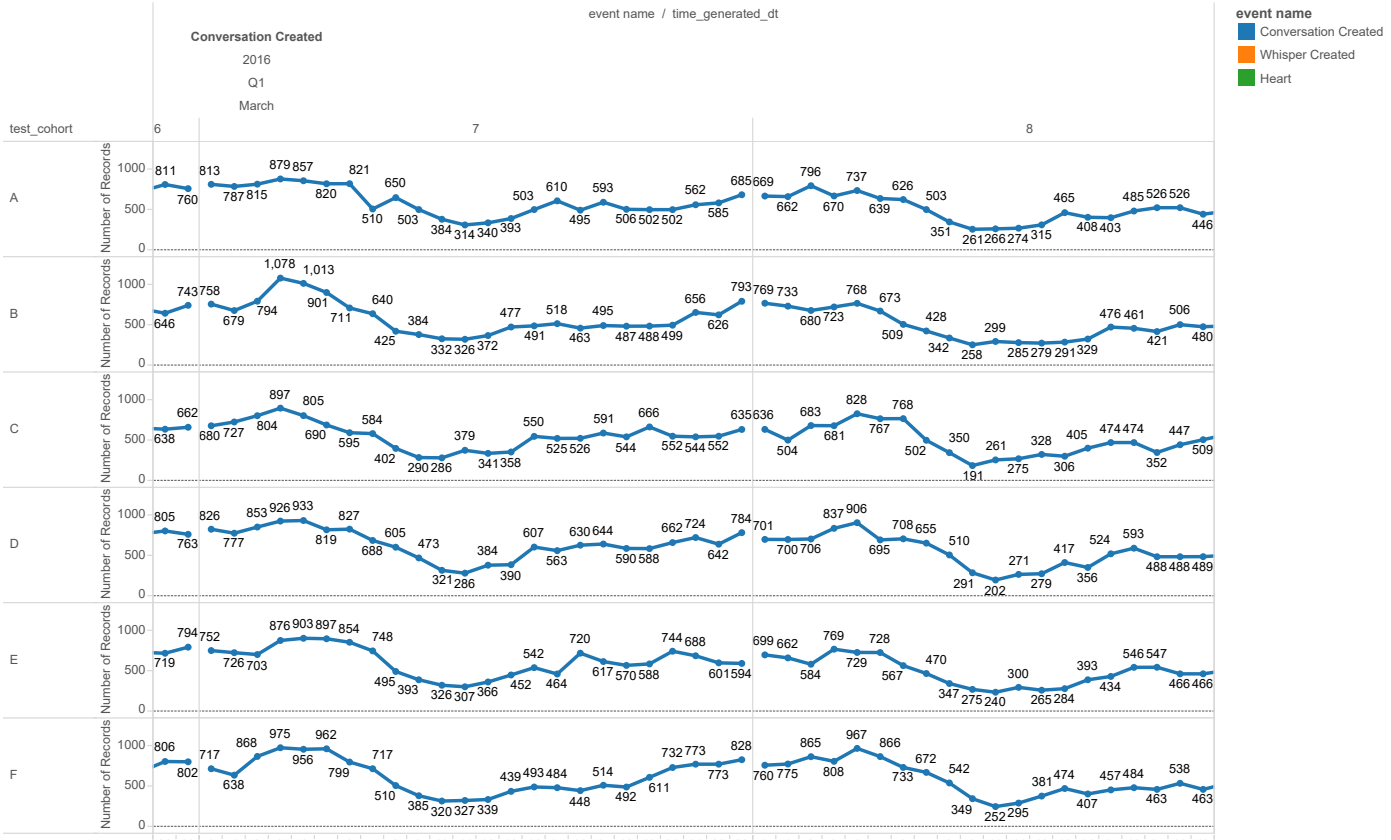
The trend of sum of Number of Records for time_generated_dt Hour broken down by event name, time_generated_dt Year, time_generated_dt Quarter, time_generated_dt Month and time_generated_dt Day vs. test_cohort. Color shows details about event name. The marks are labeled by sum of Number of Records.

test_cohorts



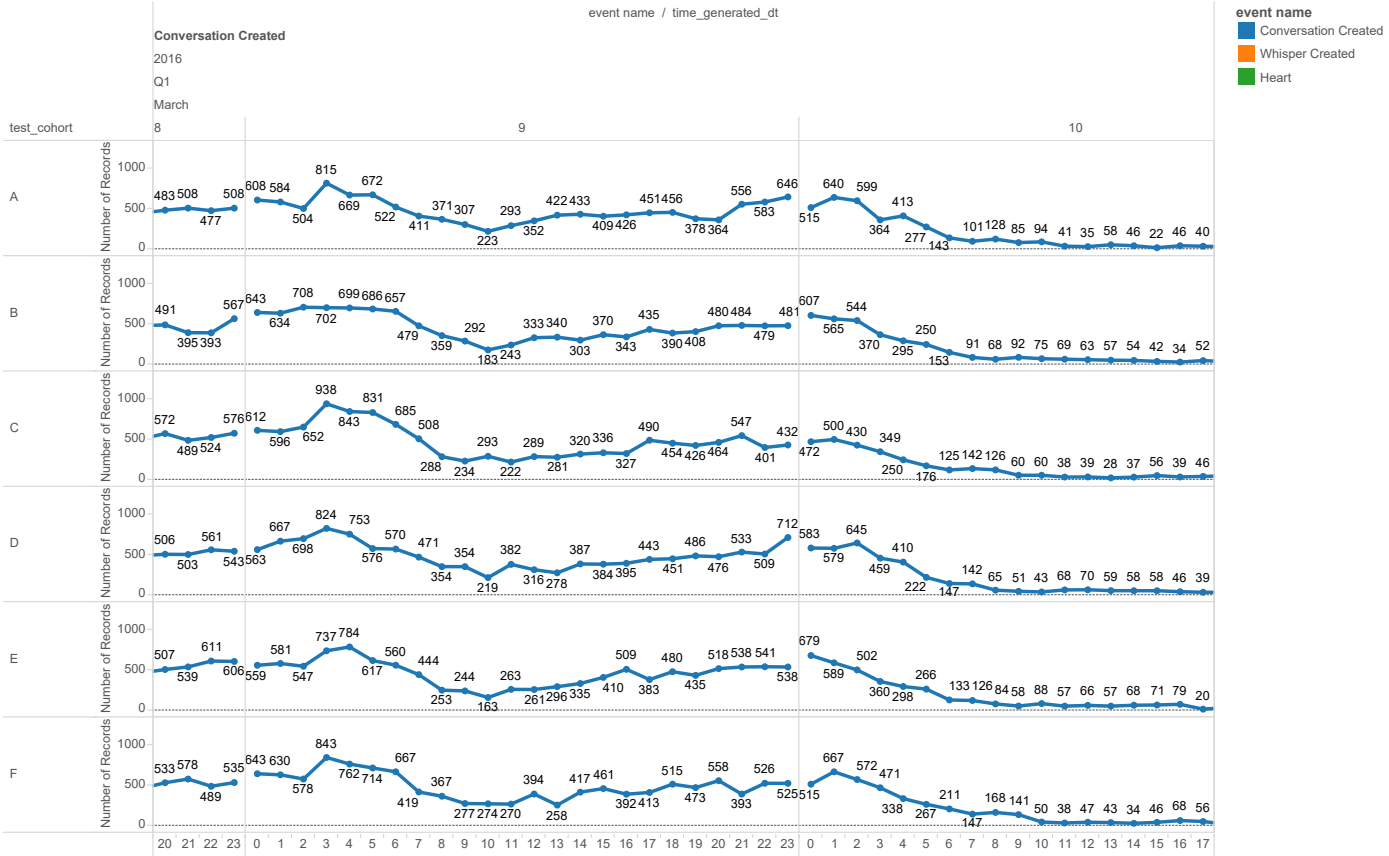
The trend of sum of Number of Records for time_generated_dt Hour broken down by event name, time_generated_dt Year, time_generated_dt Quarter, time_generated_dt Month and time_generated_dt Day vs. test_cohort. Color shows details about event name. The marks are labeled by sum of Number of Records.

test_cohorts



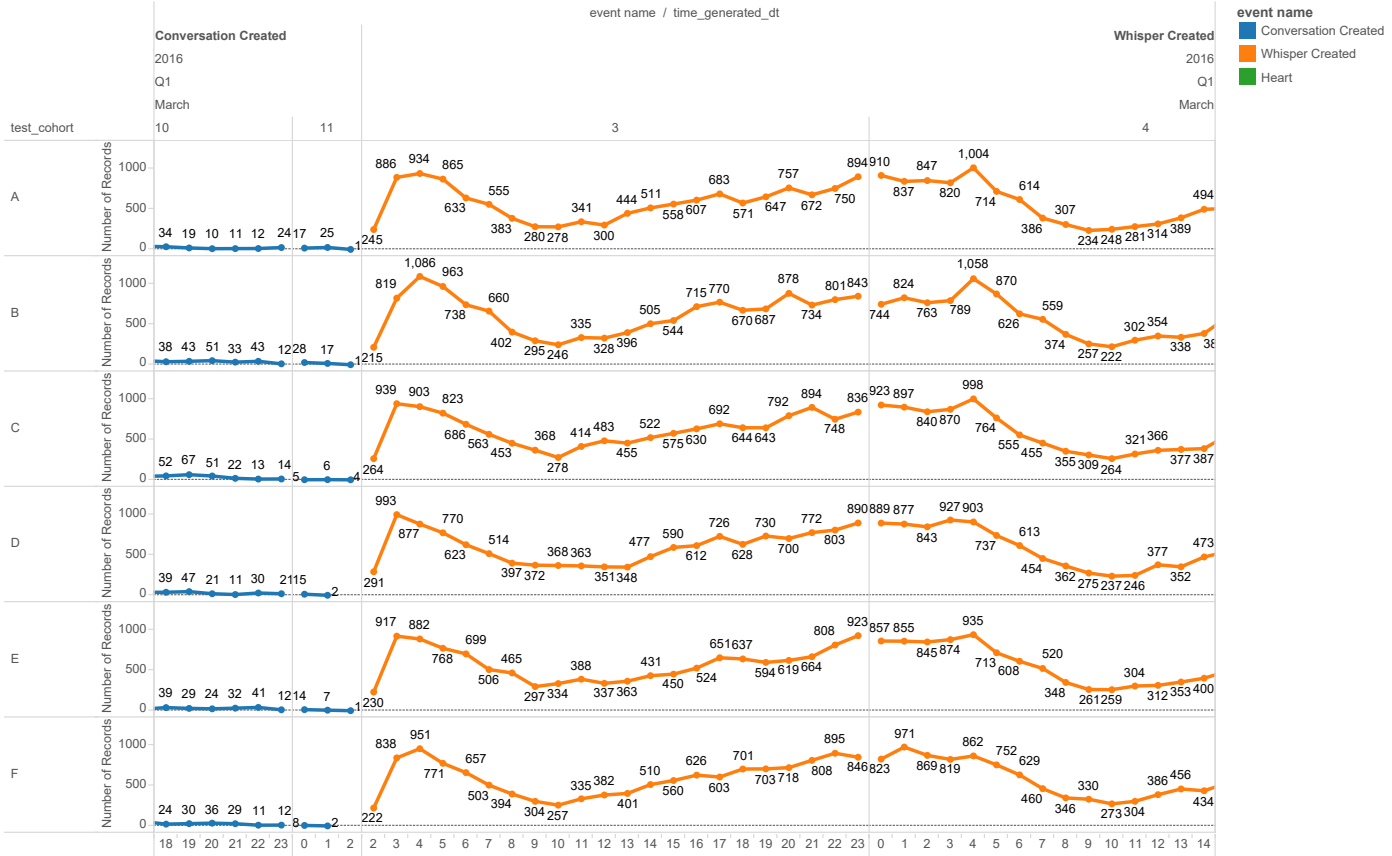
The trend of sum of Number of Records for time_generated_dt Hour broken down by event name, time_generated_dt Year, time_generated_dt Quarter, time_generated_dt Month and time_generated_dt Day vs. test_cohort. Color shows details about event name. The marks are labeled by sum of Number of Records.

test_cohorts



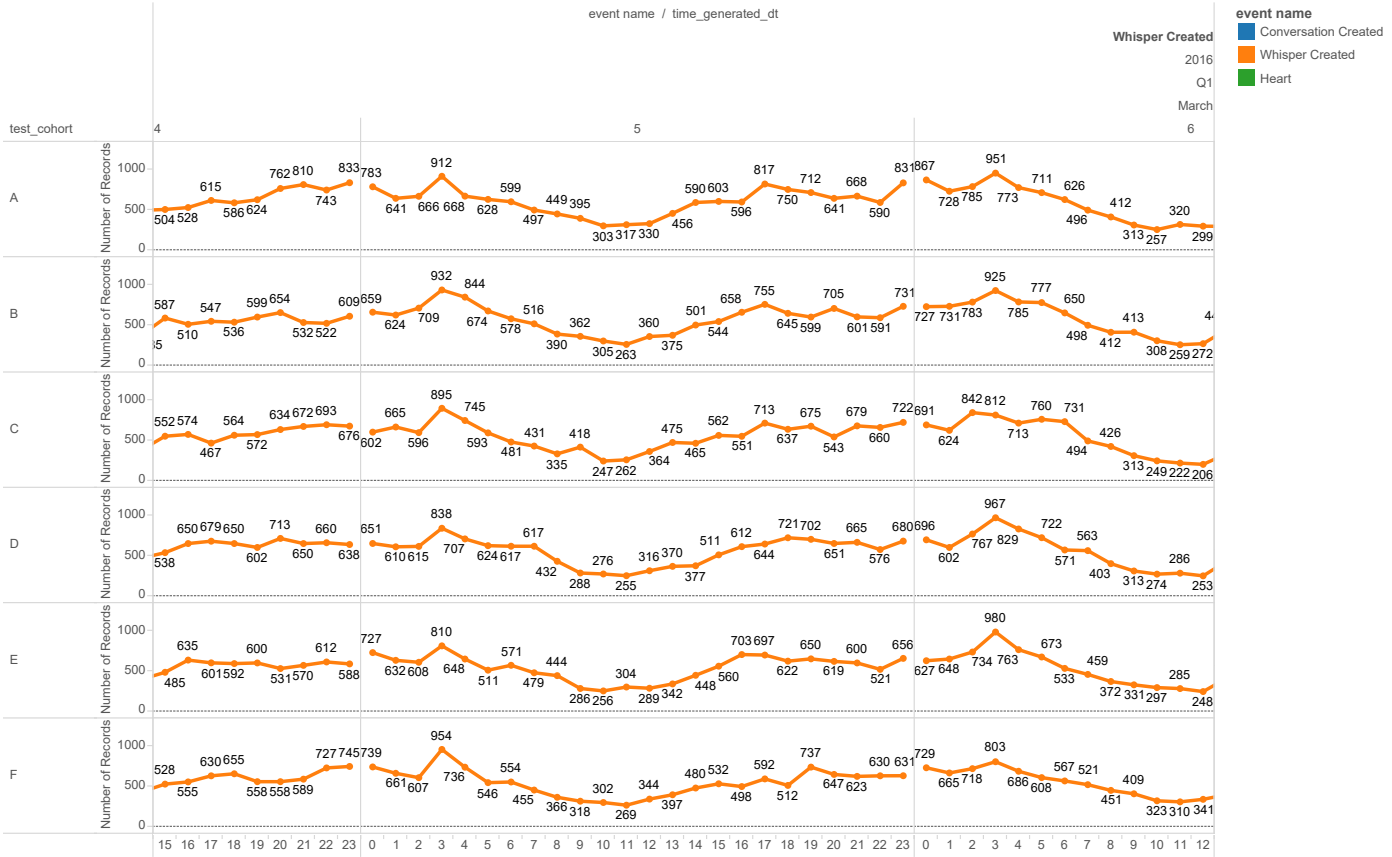
The trend of sum of Number of Records for time_generated_dt Hour broken down by event name, time_generated_dt Year, time_generated_dt Quarter, time_generated_dt Month and time_generated_dt Day vs. test_cohort. Color shows details about event name. The marks are labeled by sum of Number of Records.

test_cohorts



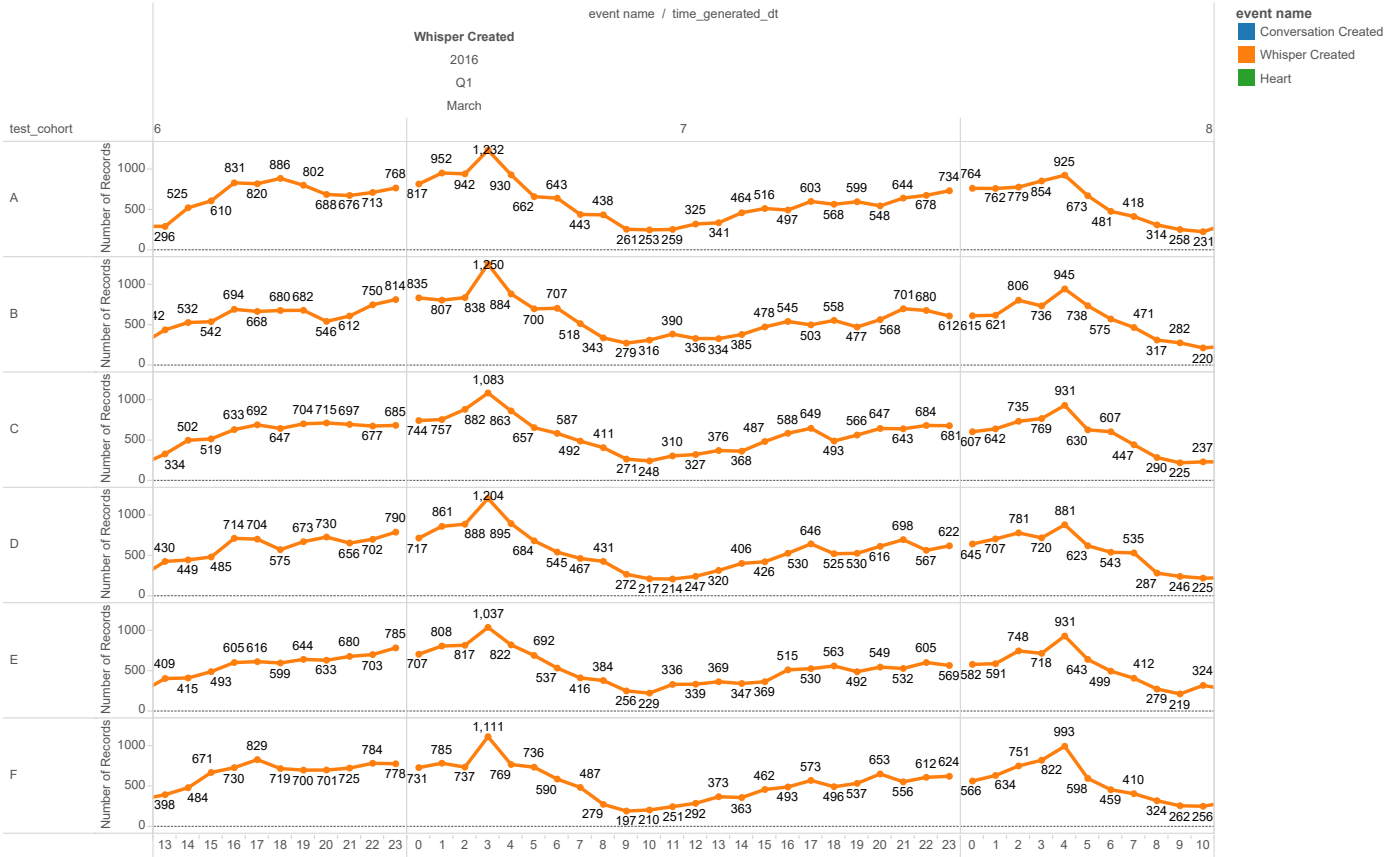
The trend of sum of Number of Records for time_generated_dt Hour broken down by event name, time_generated_dt Year, time_generated_dt Quarter, time_generated_dt Month and time_generated_dt Day vs. test_cohort. Color shows details about event name. The marks are labeled by sum of Number of Records.

test_cohorts



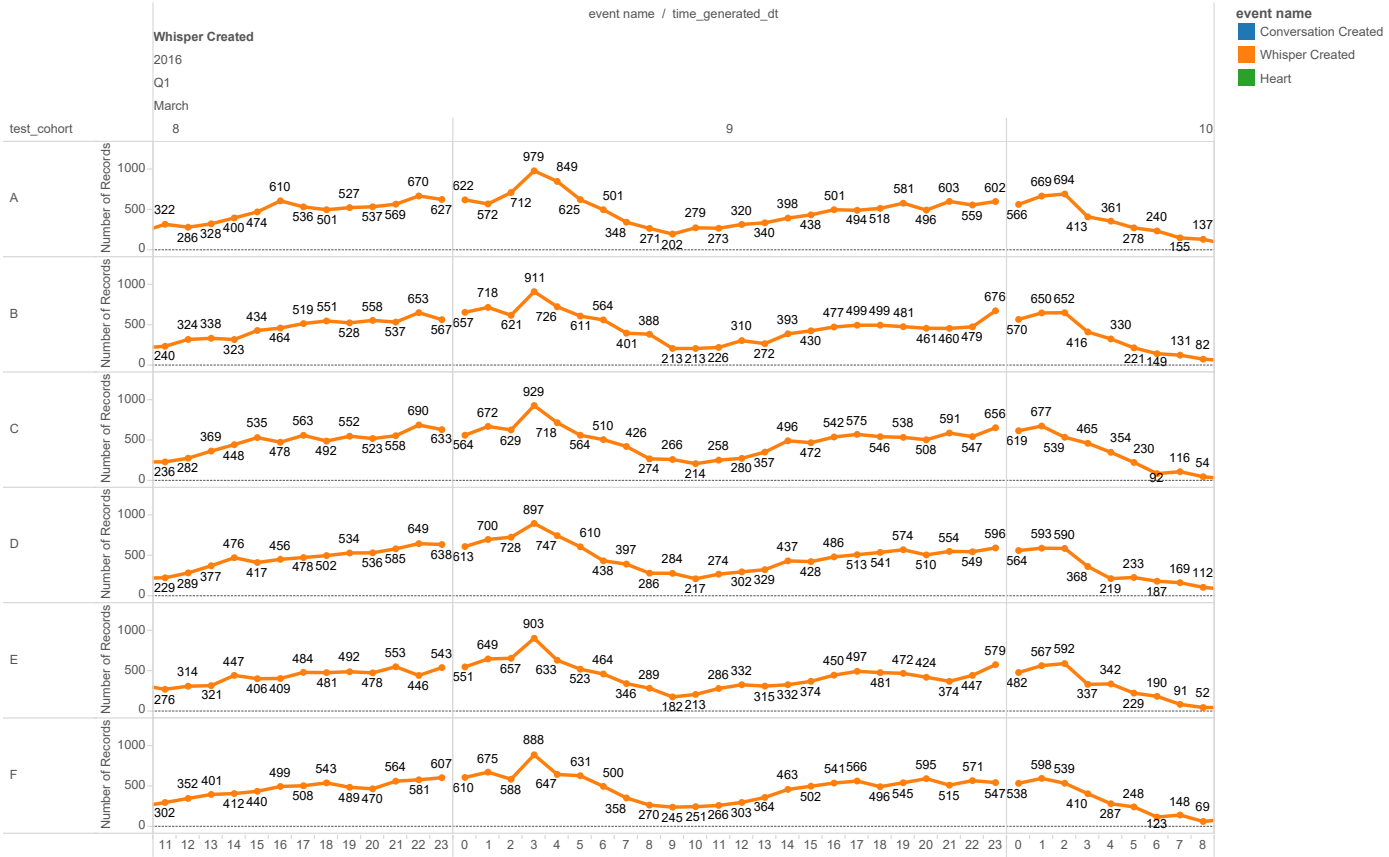
The trend of sum of Number of Records for time_generated_dt Hour broken down by event name, time_generated_dt Year, time_generated_dt Quarter, time_generated_dt Month and time_generated_dt Day vs. test_cohort. Color shows details about event name. The marks are labeled by sum of Number of Records.

test_cohorts



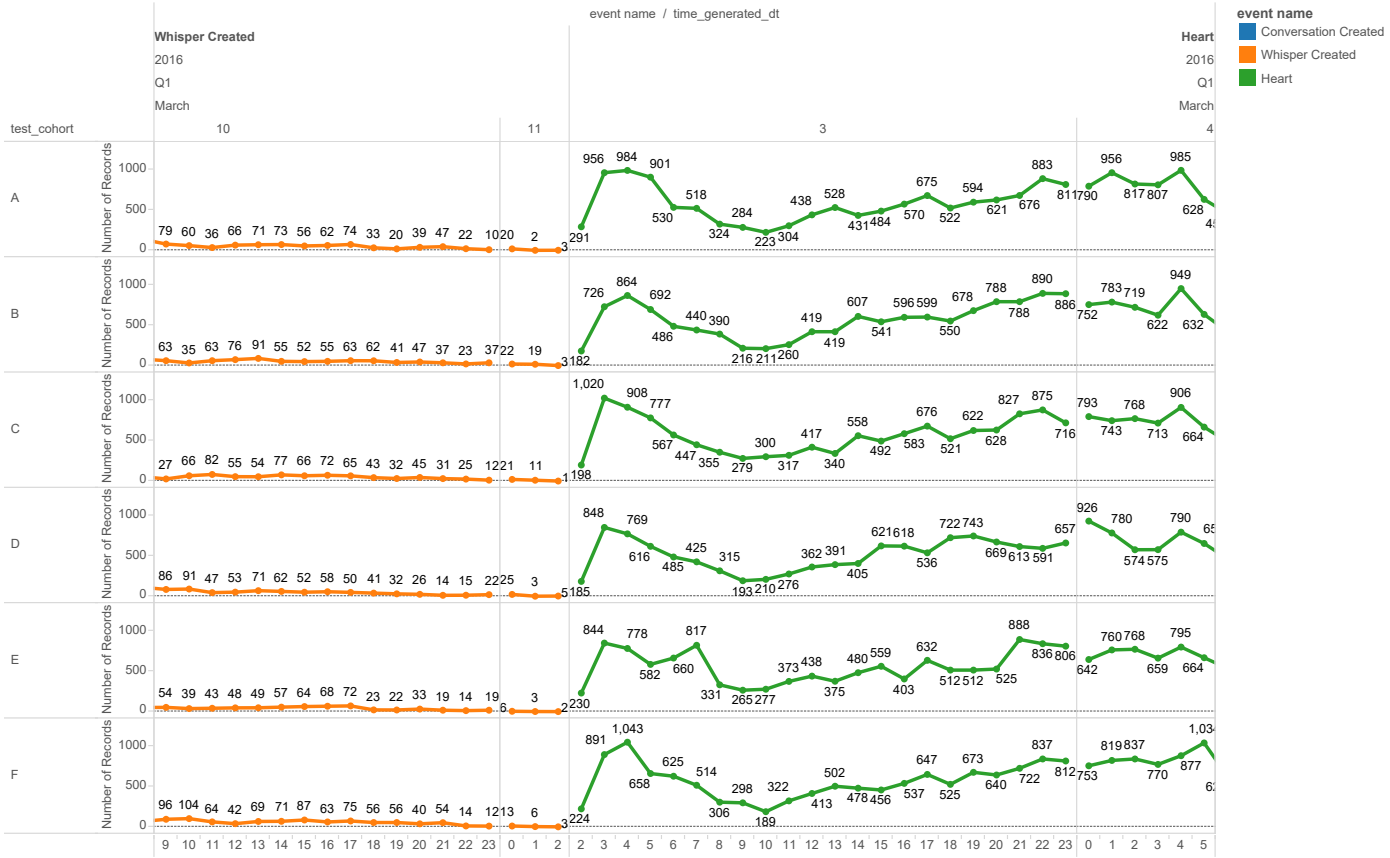
The trend of sum of Number of Records for time_generated_dt Hour broken down by event name, time_generated_dt Year, time_generated_dt Quarter, time_generated_dt Month and time_generated_dt Day vs. test_cohort. Color shows details about event name. The marks are labeled by sum of Number of Records.

test_cohorts



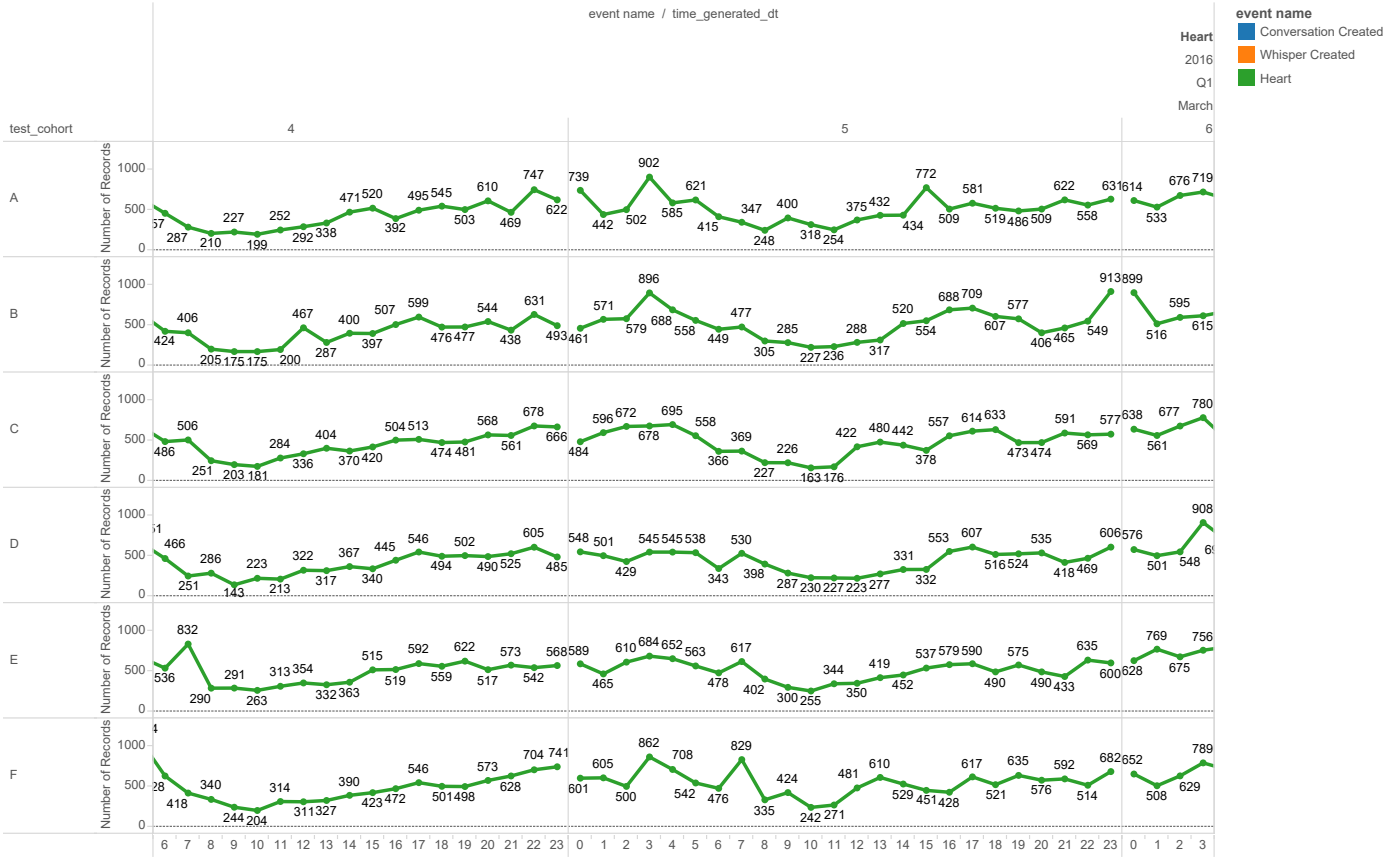
The trend of sum of Number of Records for time_generated_dt Hour broken down by event name, time_generated_dt Year, time_generated_dt Quarter, time_generated_dt Month and time_generated_dt Day vs. test_cohort. Color shows details about event name. The marks are labeled by sum of Number of Records.

test_cohorts



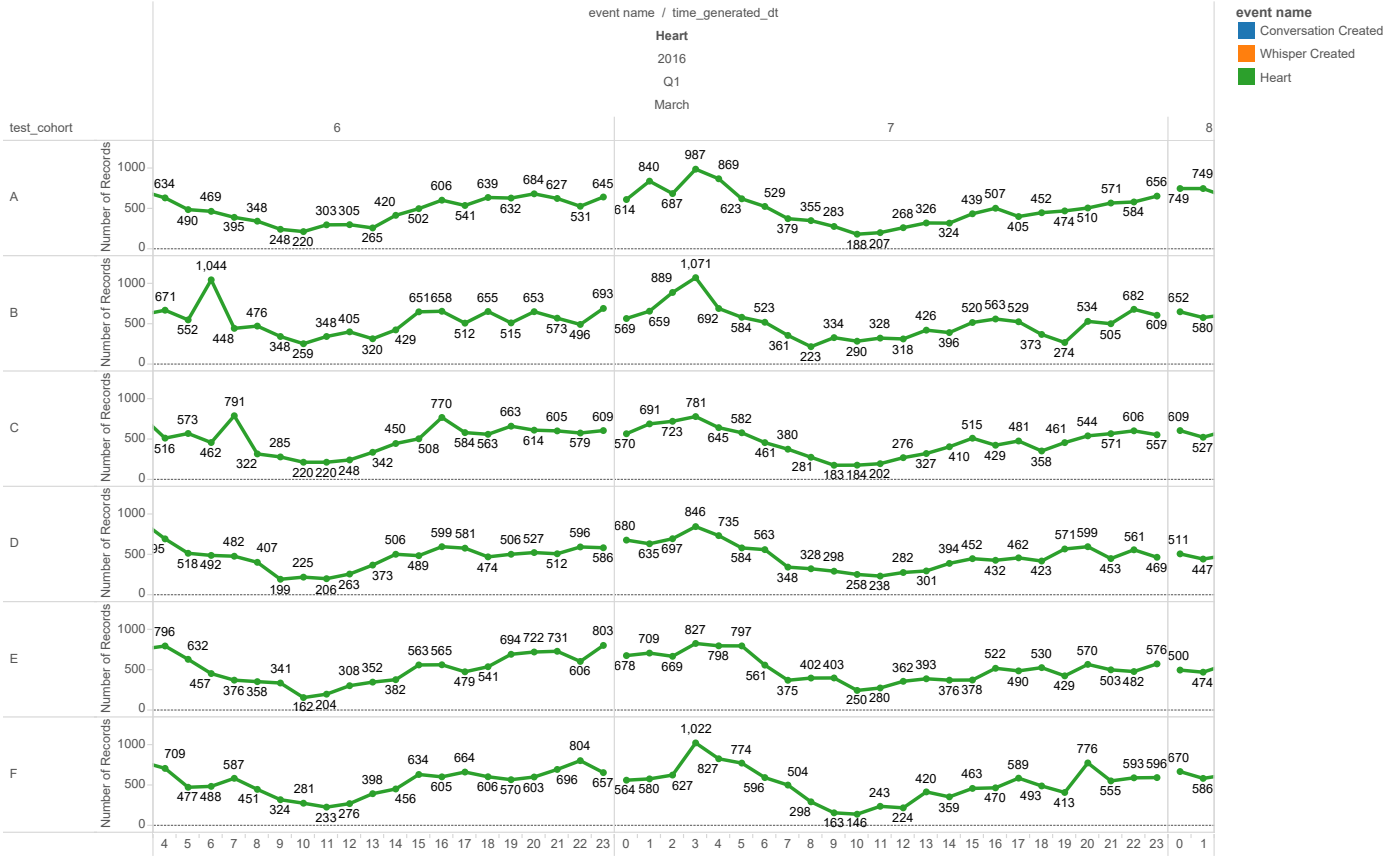
The trend of sum of Number of Records for time_generated_dt Hour broken down by event name, time_generated_dt Year, time_generated_dt Quarter, time_generated_dt Month and time_generated_dt Day vs. test_cohort. Color shows details about event name. The marks are labeled by sum of Number of Records.

test_cohorts



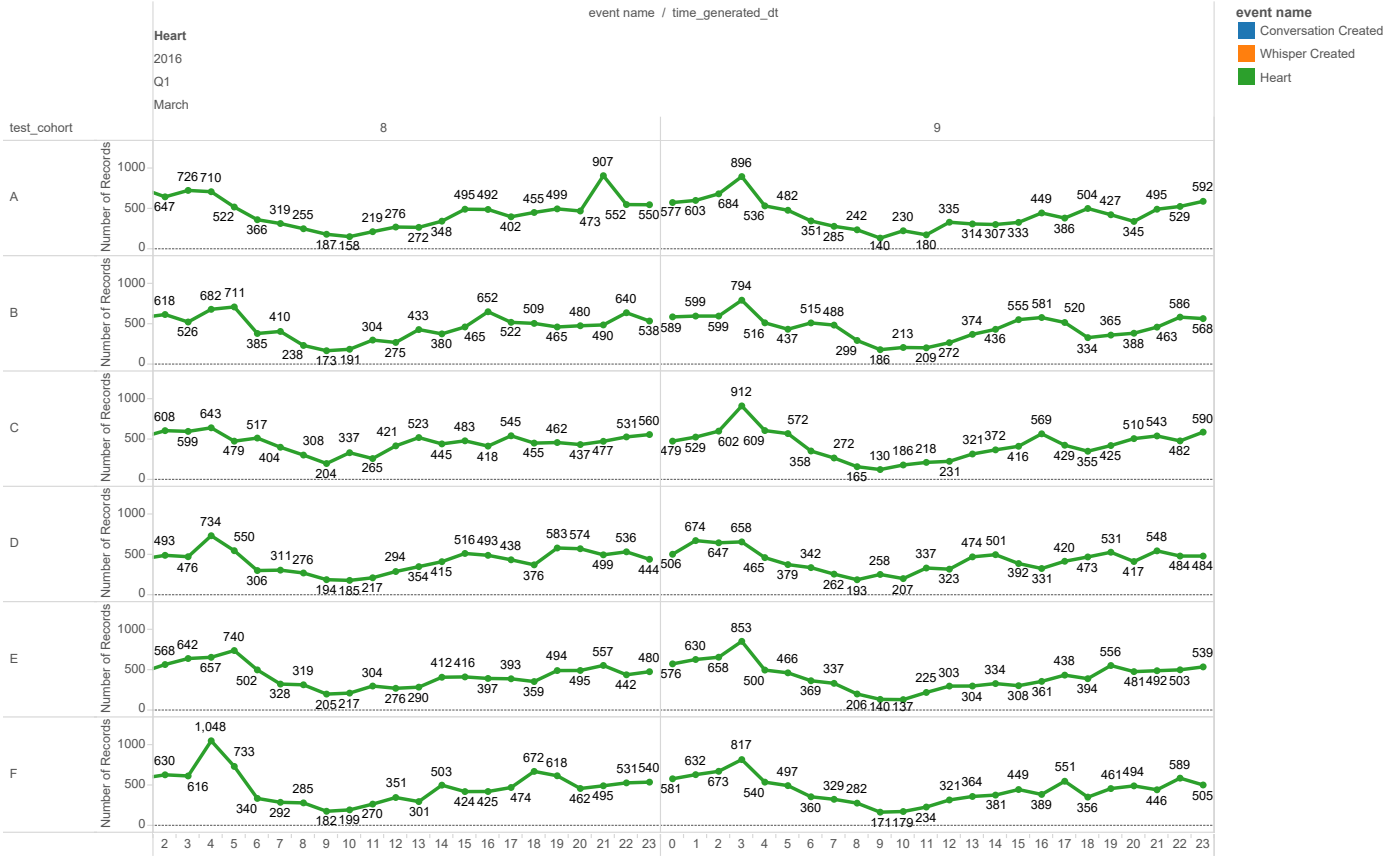
The trend of sum of Number of Records for time_generated_dt Hour broken down by event name, time_generated_dt Year, time_generated_dt Quarter, time_generated_dt Month and time_generated_dt Day vs. test_cohort. Color shows details about event name. The marks are labeled by sum of Number of Records.

test_cohorts



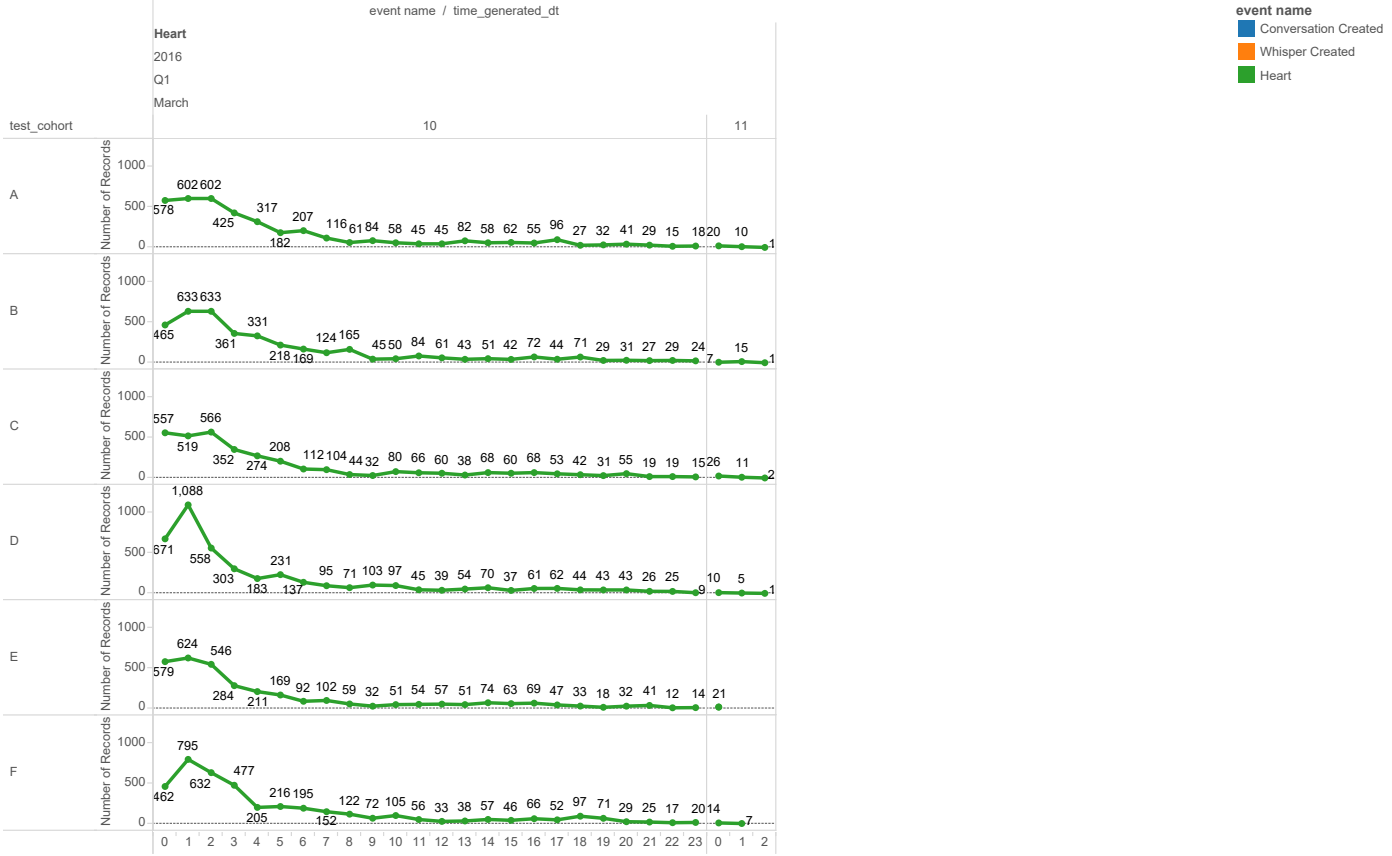
The trend of sum of Number of Records for time_generated_dt Hour broken down by event name, time_generated_dt Year, time_generated_dt Quarter, time_generated_dt Month and time_generated_dt Day vs. test_cohort. Color shows details about event name. The marks are labeled by sum of Number of Records.

test_cohorts



The trend of sum of Number of Records for time_generated_dt Hour broken down by event name, time_generated_dt Year, time_generated_dt Quarter, time_generated_dt Month and time_generated_dt Day vs. test_cohort. Color shows details about event name. The marks are labeled by sum of Number of Records.

test_cohorts



The trend of sum of Number of Records for time_generated_dt Hour broken down by event name, time_generated_dt Year, time_generated_dt Quarter, time_generated_dt Month and time_generated_dt Day vs. test_cohort. Color shows details about event name. The marks are labeled by sum of Number of Records.