

Operation Analytics and Investigating Metric Spike

Project Description:

The project operation analytics and investigating metric spike is analysis done for the complete end to end operations of a company. With the help of this, the company then finds the areas on which it must improve upon. The main objective of this project is to help the operations team and provide them accurate answers from the database using sql and various other techniques .

In this project ,

1. Case study 1 we are given with job data which contain tables like job_id, actor_id, event, language, time_spent, organization, date with the help of this we have to perform analysis for following things
 - a) Number of job reviewed: Calculation of number of jobs reviewed per day
 - b) Throughput : Calculation of 7-day rolling average of throughput
 - c) Percent share of each language : Calculation of percent share of each language
 - d) Duplicate rows : displaying duplicates from table
2. Case study 2 we are given with we are provided with three tables users , events , email_events With the help of this we have to perform analysis to find following things
 - a) User engagement : weekly engagement of users
 - b) Weekly retention : Users retention after sign up
 - c) Weekly engagement : weekly engagement per device
 - d) Email engagement : email engagement metric

Approach :

The main approach towards this project to help and assess the ops team and investors with the insights they need is by first loading the database performing various sql queries to get the exact insights the team needs quickly.

Tech used : The main software used during the project is MySql Workbench 8.0 CE since it is very easy to install and use .

LOADING THE DATABASE

1) JOB DATA

The screenshot shows the MySQL Workbench interface. On the left, the 'SCHEMAS' pane displays a tree view of the database structure, including tables like 'event_emails', 'users', and 'job_data'. The main editor window shows a SQL script for creating a database, using it, and creating a table named 'job_data'. The table has columns: 'ds' (date), 'job_id' (int not null), 'actor_id' (int not null), 'e_vent' (varchar(100) not null), 'language_name' (varchar(100) not null), 'time_spent' (int not null), and 'org' (varchar(100) not null). An 'Insert into' statement follows, populating the table with various data points including dates, job IDs, actor IDs, event names, language names, time spent, and organization names. The script ends with a 'select * from job_data;' statement.

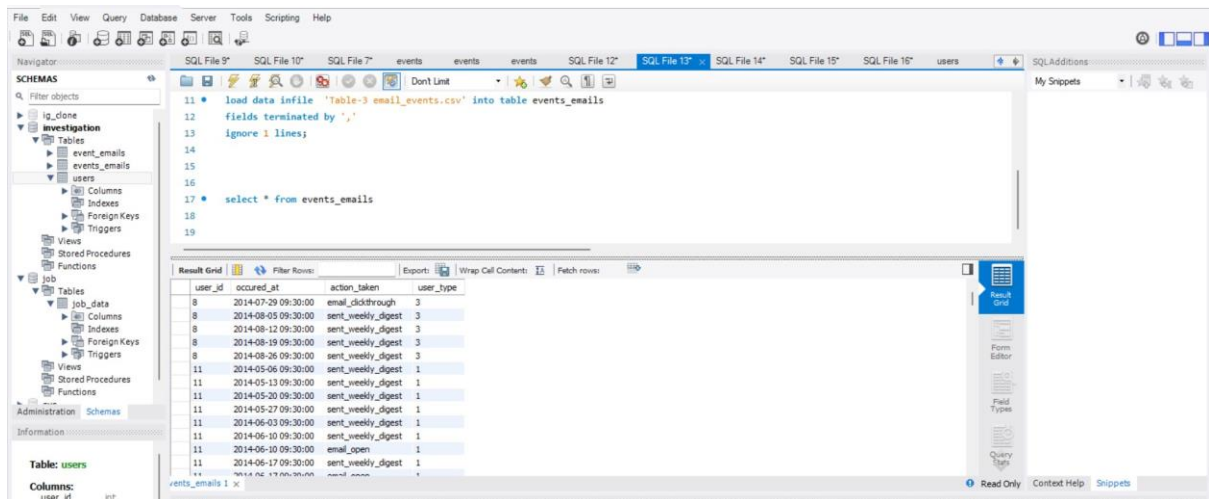
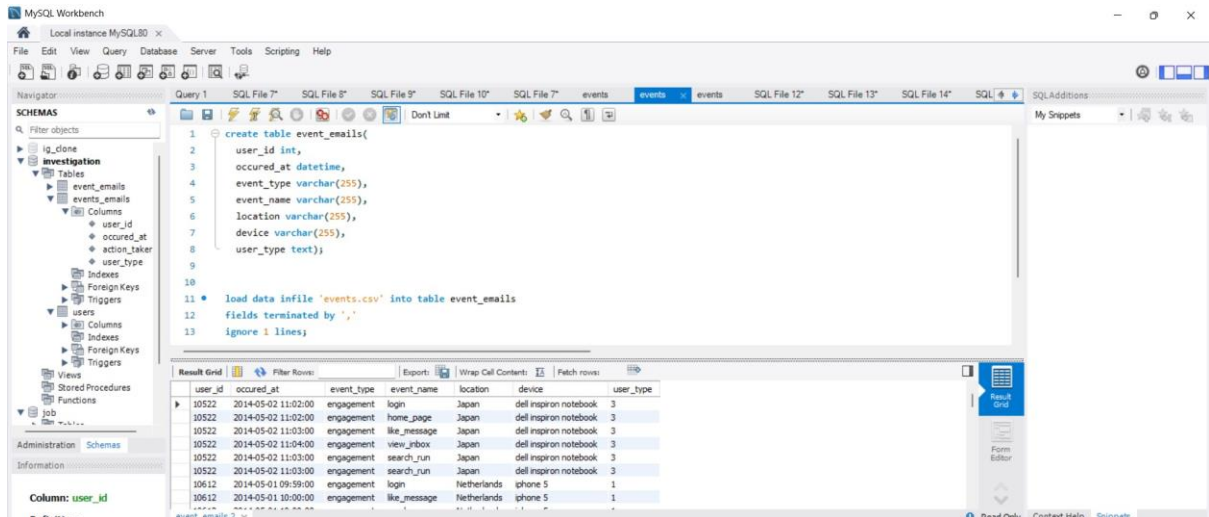
```
1 • create database job;
2 • use job;
3 • create table job_data(
4     ds date,
5     job_id int not null,
6     actor_id int not null,
7     e_vent varchar(100)not null,
8     language_name varchar(100) not null,
9     time_spent int not null,
10    org varchar(100) not null );
11 • Insert into job_data(ds,job_id,actor_id,e_vent,language_name,time_spent,org)
12 values('2020-11-30', 21, 1001, 'skip', 'English', 15, 'A'),
13 ('2020-11-30', 22, 1006, 'transfer', 'Arabic', 25, 'B'),
14 ('2020-11-29', 23, 1003, 'decision', 'Persian', 20, 'C'),
15 ('2020-11-28', 23, 1005, 'transfer', 'Persian', 22, 'D'),
16 ('2020-11-28', 25, 1002, 'decision', 'Hindi', 11, 'B'),
17 ('2020-11-27', 11, 1007, 'decision', 'French', 104, 'D'),
18 ('2020-11-26', 23, 1004, 'skip', 'Persian', 56, 'A'),
19 ('2020-11-25', 20, 1003, 'transfer', 'Italian', 45, 'C');
20
21 • select * from job_data;
22
23
```

2) Investigation metric data

The screenshot shows the MySQL Workbench interface. The main editor window displays a SQL script for loading data from a CSV file into the 'event_emails' table. The script uses the 'LOAD DATA INFILE' statement, specifying the file 'users.csv', the table 'event_emails', and the delimiter as a comma. It also includes an 'ignore 1 lines;' statement. Below the script, the 'Result Grid' shows the data loaded into the 'users' table. The table has columns: 'user_id', 'created_at', 'company_id', 'language', 'activated_at', and 'state'. The data is displayed in a grid format, showing 10 rows of user information.

```
8
9
10 • load data infile 'users.csv' into table event_emails
11 fields terminated by ','
12 ignore 1 lines;
13
14
15
```

user_id	created_at	company_id	language	activated_at	state
437	10-02-2013 09:57	8635	portugese	10-02-2013 09:59	active
438	10-02-2013 15:54	5	spanish		pending
439	11-02-2013 08:48	2147	german	11-02-2013 08:49	active
440	11-02-2013 08:14	1705	english		pending
441	11-02-2013 11:33	25	english	11-02-2013 11:35	active
442	11-02-2013 08:33	27	arabic		pending
443	11-02-2013 14:24	4822	english	11-02-2013 14:26	active
444	11-02-2013 13:45	10576	spanish		pending
445	11-02-2013 12:32	73	french		pending
446	11-02-2013 14:00	5878	english		pending
447	11-02-2013 03:06	11949	french		pending
448	11-02-2013 13:37	9602	chinese		pending
449	11-02-2013 18:08	12098	indian		pending



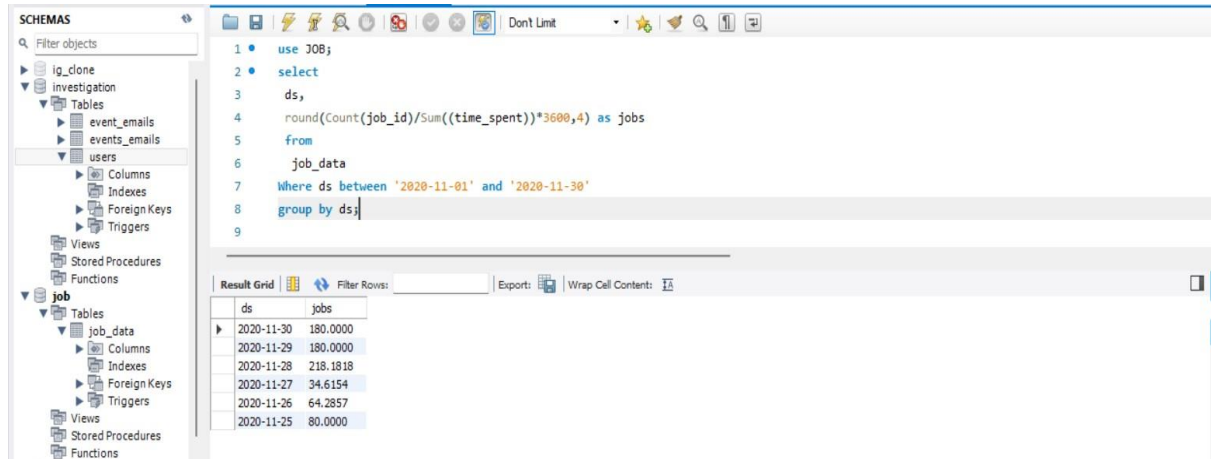
In loading users, events and email_events tables we have to use load infile statement since the data is too large and mysql is slow in loading data through is import option

Insights :

A) CASE STUDY 1 (JOB DATA)

1. Number of jobs reviewed

We need to calculate number of jobs reviewed per hour per day for November



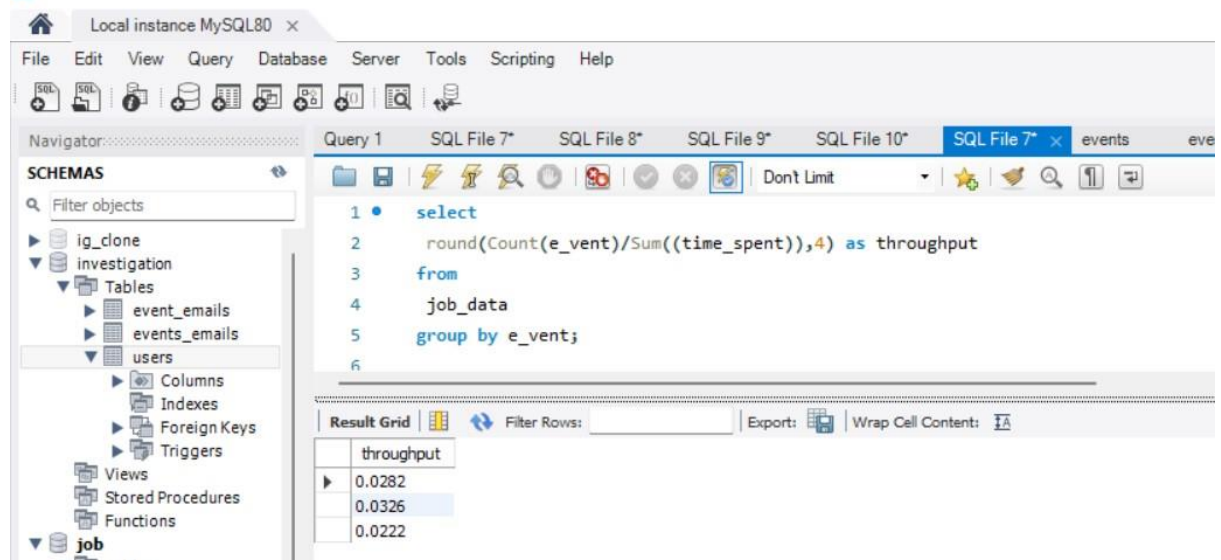
```
1 use job;
2 select
3   ds,
4   round(Count(job_id)/Sum((time_spent))*3600,4) as jobs
5 from
6   job_data
7 where ds between '2020-11-01' and '2020-11-30'
8 group by ds;
```

ds	jobs
2020-11-30	180.0000
2020-11-29	180.0000
2020-11-28	218.1818
2020-11-27	34.6154
2020-11-26	64.2857
2020-11-25	80.0000

2. Throughput:

There are no of events happening per second. We need to calculate 7 day rolling average throughput

MySQL Workbench



```
1 select
2   round(Count(e_event)/Sum((time_spent)),4) as throughput
3 from
4   job_data
5 group by e_event;
```

throughput
0.0282
0.0326
0.0222

Daily metric keeps on fluctuating every time because of many reasons and is unstable this in turn might generate wrong result therefore we always use 7day rolling average is preferable to use for generating perfect analysis

3. Percentage share of each language:

Calculation of percentage share of each language in 30 days

The screenshot shows the MySQL Workbench interface. The left sidebar displays the 'SCHEMAS' tree with the 'job' database selected. The main query editor contains the following SQL code:

```
2 language_name as Languages,  
3 Round(((count(language_name)/(select count(*) from job_data))*100),1)percentage  
4 from  
5 job_data  
6 group by language_name;
```

The 'Result Grid' at the bottom shows the output of the query:

Languages	percentage
English	12.5
Arabic	12.5
Persian	37.5
Hindi	12.5
French	12.5
Italian	12.5

4. Duplicate rows :

Shows rows having duplicate values

The screenshot shows the MySQL Workbench interface. The left sidebar displays the 'SCHEMAS' tree with the 'job' database selected. The main query editor contains the following SQL code:

```
2 actor_id,  
3 count(*) as Dups  
4 from  
5 job_data  
6 group by actor_id
```

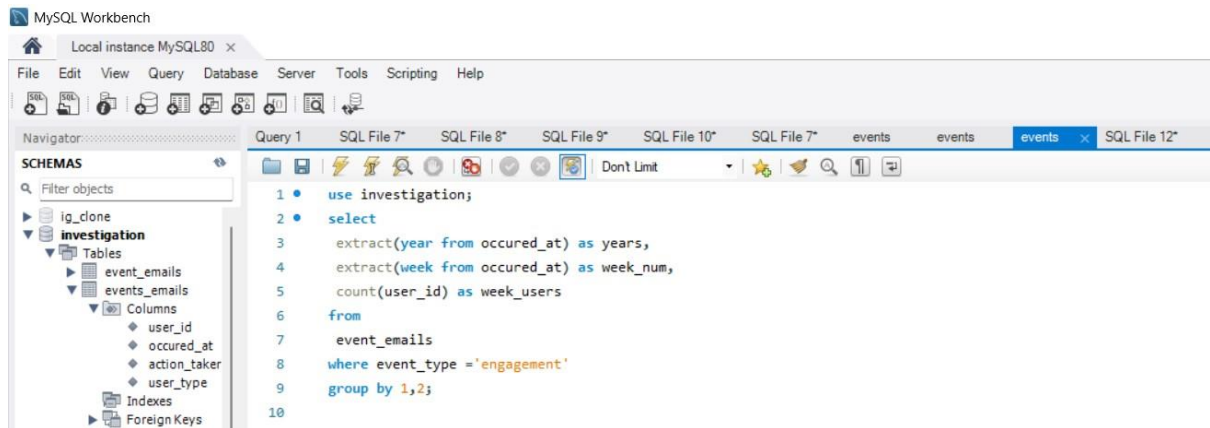
The 'Result Grid' at the bottom shows the output of the query:

actor_id	Dups
1001	1
1006	1
1003	2
1005	1
1002	1
1007	1
1004	1

B) Investigating metric spike Case Study 2

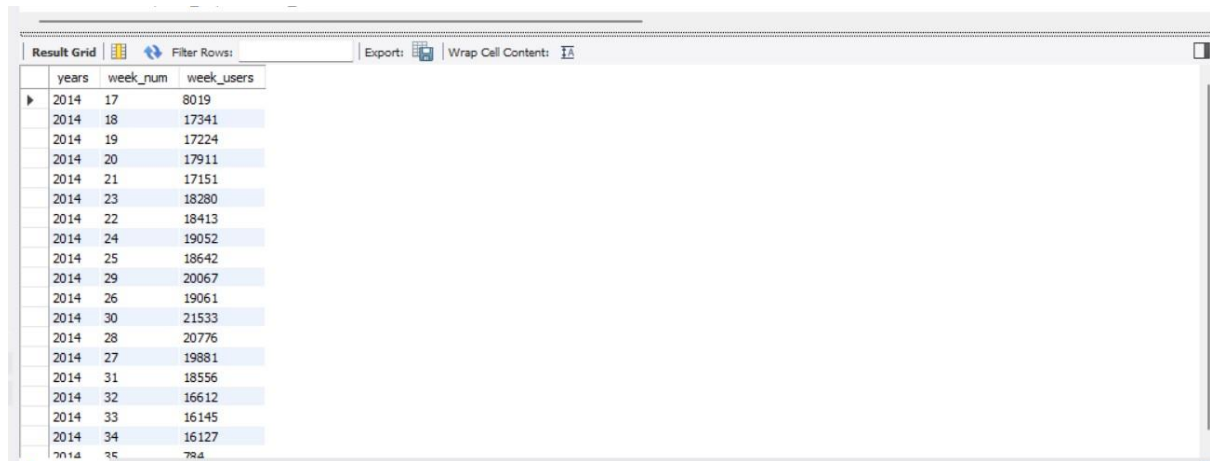
1) User engagement :

Weekly engagement of users



The screenshot shows the MySQL Workbench interface. The left sidebar displays the 'SCHEMAS' tree with the 'investigation' database selected. The main query editor contains the following SQL code:

```
1 • use investigation;
2 • select
3     extract(year from occurred_at) as years,
4     extract(week from occurred_at) as week_num,
5     count(user_id) as week_users
6 from
7     event_emails
8 where event_type = 'engagement'
9 group by 1,2;
10
```

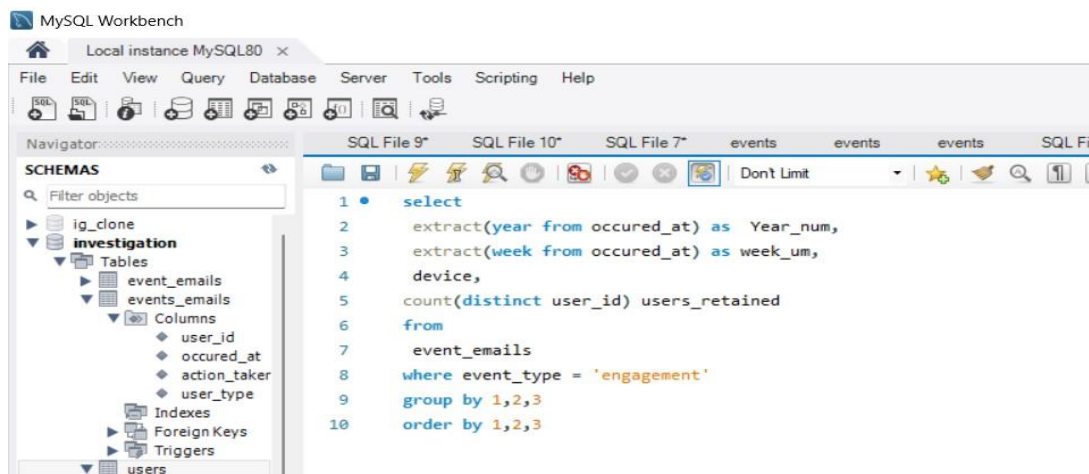


The screenshot shows the 'Result Grid' of the query. The table has three columns: 'years', 'week_num', and 'week_users'. The data is as follows:

years	week_num	week_users
2014	17	8019
2014	18	17341
2014	19	17224
2014	20	17911
2014	21	17151
2014	23	18280
2014	22	18413
2014	24	19052
2014	25	18642
2014	29	20067
2014	26	19061
2014	30	21533
2014	28	20776
2014	27	19881
2014	31	18556
2014	32	16612
2014	33	16145
2014	34	16127
2014	35	794

2) Weekly retention after sign up :

Calculation of users weekly retention after sign up cohort



The screenshot shows the MySQL Workbench interface. The left sidebar displays the 'SCHEMAS' tree with the 'investigation' database selected. The main query editor contains the following SQL code:

```
1 • select
2     extract(year from occurred_at) as Year_num,
3     extract(week from occurred_at) as week_um,
4     device,
5     count(distinct user_id) users_retained
6 from
7     event_emails
8 where event_type = 'engagement'
9 group by 1,2,3
10 order by 1,2,3
```


Year_num	week_um	device	users_retained
2014	17	acer aspire desktop	9
2014	17	acer aspire notebook	20
2014	17	amazon fire phone	4
2014	17	asus chromebook	21
2014	17	dell inspiron desktop	18
2014	17	dell inspiron notebook	46
2014	17	hp pavilion desktop	14
2014	17	htc one	16
2014	17	ipad air	27
2014	17	ipad mini	19
2014	17	iphone 4s	21
2014	17	iphone 5	65
2014	17	iphone 5s	42
2014	17	kindle fire	6
2014	17	lenovo thinkpad	86
2014	17	mac mini	6
2014	17	macbook air	54
2014	17	macbook pro	143
2014	17	nexus 10	16
2014	17	nexus 5	40
2014	17	nexus 7	18
2014	17	nokia lumia 635	17
2014	17	samsung galaxy tablet	8
2014	17	samsung galaxy note	7
2014	17	samsung galaxy s4	52
2014	17	windows surface	10
2014	18	acer aspire desktop	26
2014	18	acer aspire notebook	33
2014	18	amazon fire phone	9
2014	18	asus chromebook	42
2014	18	dell inspiron desktop	58
2014	18	dell inspiron notebook	77

2014	18 hp pavilion desktop	37
2014	18 htc one	19
2014	18 ipad air	52
2014	18 ipad mini	30
2014	18 iphone 4s	46
2014	18 iphone 5	113
2014	18 iphone 5s	73
2014	18 kindle fire	27
2014	18 lenovo thinkpad	153
2014	18 mac mini	13
2014	18 macbook air	121
2014	18 macbook pro	252
2014	18 nexus 10	30
2014	18 nexus 5	73
2014	18 nexus 7	30
2014	18 nokia lumia 635	33
2014	18 samsung galaxy tablet	11
2014	18 samsung galaxy note	15
2014	18 samsung galaxy s4	82
2014	18 windows surface	10
2014	19 acer aspire desktop	23
2014	19 acer aspire notebook	41
2014	19 amazon fire phone	12
2014	19 asus chromebook	27
2014	19 dell inspiron desktop	36
2014	19 dell inspiron notebook	83
2014	19 hp pavilion desktop	40
2014	19 htc one	30
2014	19 ipad air	55
2014	19 ipad mini	36
2014	19 iphone 4s	44
2014	19 iphone 5	115
2014	19 iphone 5s	79

2014	19 kindle fire	21
2014	19 lenovo thinkpad	178
2014	19 mac mini	18
2014	19 macbook air	112
2014	19 macbook pro	266
2014	19 nexus 10	25
2014	19 nexus 5	87
2014	19 nexus 7	41
2014	19 nokia lumia 635	23
2014	19 samsung galaxy tablet	6
2014	19 samsung galaxy note	11
2014	19 samsung galaxy s4	91
2014	19 windows surface	16
2014	20 acer aspire desktop	23
2014	20 acer aspire notebook	40
2014	20 amazon fire phone	11
2014	20 asus chromebook	41
2014	20 dell inspiron desktop	52
2014	20 dell inspiron notebook	84
2014	20 hp pavilion desktop	30
2014	20 htc one	29
2014	20 ipad air	59
2014	20 ipad mini	32
2014	20 iphone 4s	55
2014	20 iphone 5	125
2014	20 iphone 5s	79
2014	20 kindle fire	23
2014	20 lenovo thinkpad	173
2014	20 mac mini	26
2014	20 macbook air	119
2014	20 macbook pro	256
2014	20 nexus 10	22
2014	20 nexus 5	103

2014	20 nexus 7	32
2014	20 nokia lumia 635	22
2014	20 samsung galaxy tablet	9
2014	20 samsung galaxy note	18
2014	20 samsung galaxy s4	93
2014	20 windows surface	21
2014	21 acer aspire desktop	29
2014	21 acer aspire notebook	47
2014	21 amazon fire phone	5
2014	21 asus chromebook	38
2014	21 dell inspiron desktop	41
2014	21 dell inspiron notebook	80
2014	21 hp pavilion desktop	44
2014	21 htc one	21
2014	21 ipad air	51
2014	21 ipad mini	23
2014	21 iphone 4s	45
2014	21 iphone 5	137
2014	21 iphone 5s	74
2014	21 kindle fire	30
2014	21 lenovo thinkpad	167
2014	21 mac mini	18
2014	21 macbook air	110
2014	21 macbook pro	247
2014	21 nexus 10	25
2014	21 nexus 5	91
2014	21 nexus 7	29
2014	21 nokia lumia 635	25
2014	21 samsung galaxy tablet	6
2014	21 samsung galaxy note	20
2014	21 samsung galaxy s4	84
2014	21 windows surface	17
2014	22 acer aspire desktop	25

These are some of the results

3) Weekly Engagement per device :

```

1  select
2      extract(week from occurred_at) as week_num,
3      count(distinct case when device IN('dell inspiron notebook') then user_id else null end) AS 'Dell Inspiron Notebook',
4      count(distinct case when device IN('dell inspiron desktop') then user_id else null end) AS 'Dell Inspiron Desktop',
5      count(distinct case when device IN('hp pavilion desktop') then user_id else null end) AS 'Hp Pavilion Desktop',
6      count(distinct case when device IN('kindle fire') then user_id else null end) AS 'Kindle Fire',
7      count(distinct case when device IN('iphone 4s') then user_id else null end) AS 'iPhone 4s',
8      count(distinct case when device IN('iphone 5') then user_id else null end) AS 'iPhone 5',
9      count(distinct case when device IN('ipad mini') then user_id else null end) AS 'Ipad Mini',
10     count(distinct case when device IN('ipad air') then user_id else null end) AS 'Ipad Air',
11     count(distinct case when device IN('macbook pro') then user_id else null end) AS 'Macbook Pro',
12     count(distinct case when device IN('macbook air') then user_id else null end) AS 'Macbook Air',
13     count(distinct case when device IN('windows surface') then user_id else null end) AS 'Windows Surface',
14     count(distinct case when device IN('samsung galaxy s4') then user_id else null end) AS 'Samsung Galaxy S4',
15     count(distinct case when device IN('samsung galaxy tablet') then user_id else null end) AS 'Samsung Galaxy Tablet',
16     count(distinct case when device IN('samsung galaxy note') then user_id else null end) AS 'Samsung Galaxy Note',
17     count(distinct case when device IN('lenovo thinkpad') then user_id else null end) AS 'Lenovo Thinkpad',
18     count(distinct case when device IN('acer aspire notebook') then user_id else null end) AS 'Acer Aspire Notebook',
19     count(distinct case when device IN('asus chromebook') then user_id else null end) AS 'Asus Chromebook',
20     count(distinct case when device IN('nexus 7') then user_id else null end) AS 'Nexus 7',
21     count(distinct case when device IN('nexus 5') then user_id else null end) AS 'Nexus 5',
22     count(distinct case when device IN('nexus 10') then user_id else null end) AS 'Nexus 10',
23     count(distinct case when device IN('nokia lumia 635') then user_id else null end) AS 'Nokia Lumia 635',
24     count(distinct case when device IN('htc one') then user_id else null end) AS 'Htc One',
25     count(distinct case when device IN('amazon fire phone') then user_id else null end) AS 'Amazon Fire Phone'
26  from
27      event_emails
28  where event_type = 'engagement'
29  group by week_num
30  order by week_num;
31

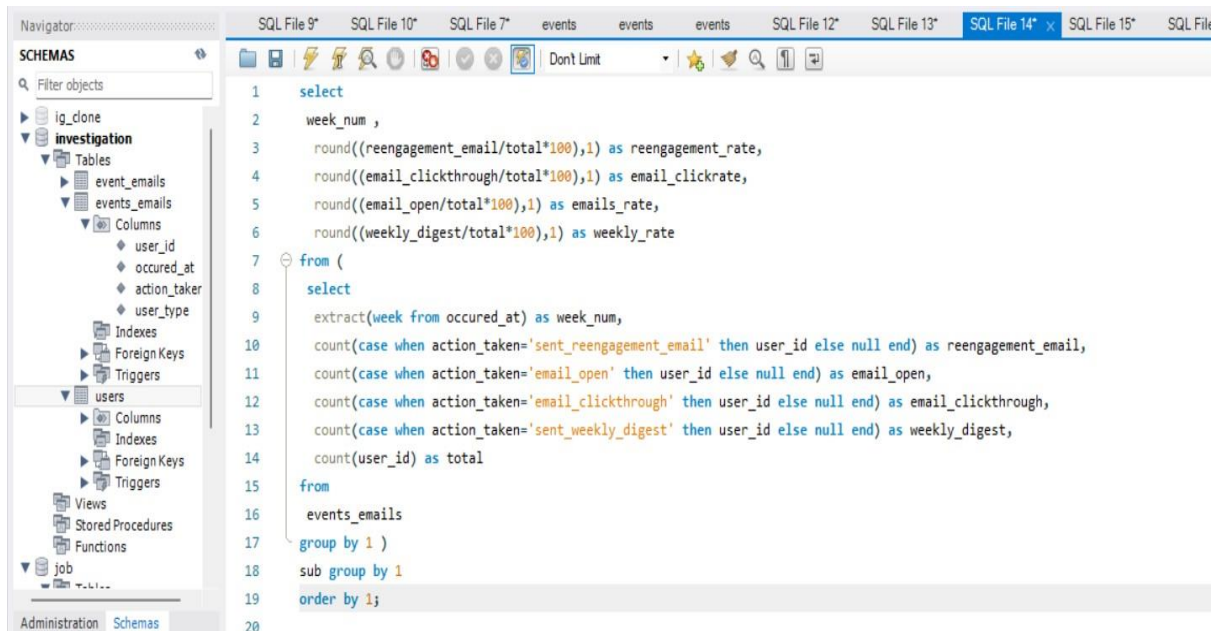
```

A	B	C	D	E	F	G	H	I	J	K	L
week_num	Dell Inspiron Notebook	Dell Inspiron Desktop	Hp Pavilion Desktop	Kindle Fire	iPhone 4s	iPhone 5	Ipad Mini	Ipad Air	Macbook Pro	Macbook Air	Windows Surface
17	46	18	14	6	21	0	19	27	143	54	10
18	77	58	37	27	46	0	30	52	252	121	10
19	83	36	40	21	44	0	36	55	266	112	16
20	84	52	30	23	55	0	32	59	256	119	21
21	80	41	44	30	45	0	23	51	247	110	17
22	92	52	38	21	45	0	34	58	251	145	15
23	103	53	54	25	53	0	33	41	266	124	14
24	99	59	56	25	53	0	39	57	255	152	22
25	105	52	52	24	40	0	30	57	275	121	22
26	89	60	46	26	50	0	43	56	269	134	21
27	89	53	56	25	67	0	35	55	302	142	33
28	103	56	56	31	61	0	35	54	295	148	33
29	113	54	58	37	60	0	34	52	295	148	28
30	127	54	42	25	65	0	35	70	322	159	19
31	113	44	51	14	56	0	27	55	321	147	19
32	104	57	51	12	34	0	30	48	307	125	10
33	110	37	38	14	35	0	28	40	312	133	15
34	105	49	36	13	50	0	25	39	292	136	18
35	9	1	1	3	6	0	2	0	17	10	3

M	N	O	P	Q	R	S	T	U	V	W
Samsung Galaxy S4 ▾	Samsung Galaxy Tablet ▾	Samsung Galaxy Note ▾	Lenovo Thinkpad ▾	Acer Aspire Notebook ▾	Asus Chromebook ▾	Nexus 7 ▾	Nexus 5 ▾	Nexus 10 ▾	Nokia Lumia 635 ▾	Htc One ▾
52	0	7	86	20	21	18	40	16	17	16
82	0	15	153	33	42	30	73	30	33	19
91	0	11	178	41	27	41	87	25	23	30
93	0	18	173	40	41	32	103	22	22	29
84	0	20	167	47	38	29	91	25	25	21
105	0	19	176	41	52	45	96	27	25	24
99	0	14	176	43	49	36	88	45	31	20
101	0	20	165	40	43	49	87	38	35	20
99	0	14	197	47	38	51	89	29	37	21
112	0	9	192	35	49	46	87	29	42	23
116	0	15	202	49	52	40	84	37	31	27
122	0	10	220	49	50	39	85	26	35	26
123	0	16	209	53	49	45	77	25	43	31
103	0	15	206	60	56	62	84	36	34	31
100	0	14	207	55	56	38	69	24	28	13
82	0	12	179	55	62	25	67	30	28	18
80	0	13	191	46	49	30	70	23	27	19
90	0	13	193	63	47	33	70	25	17	25
6	0	1	16	3	6	2	4	2	2	2

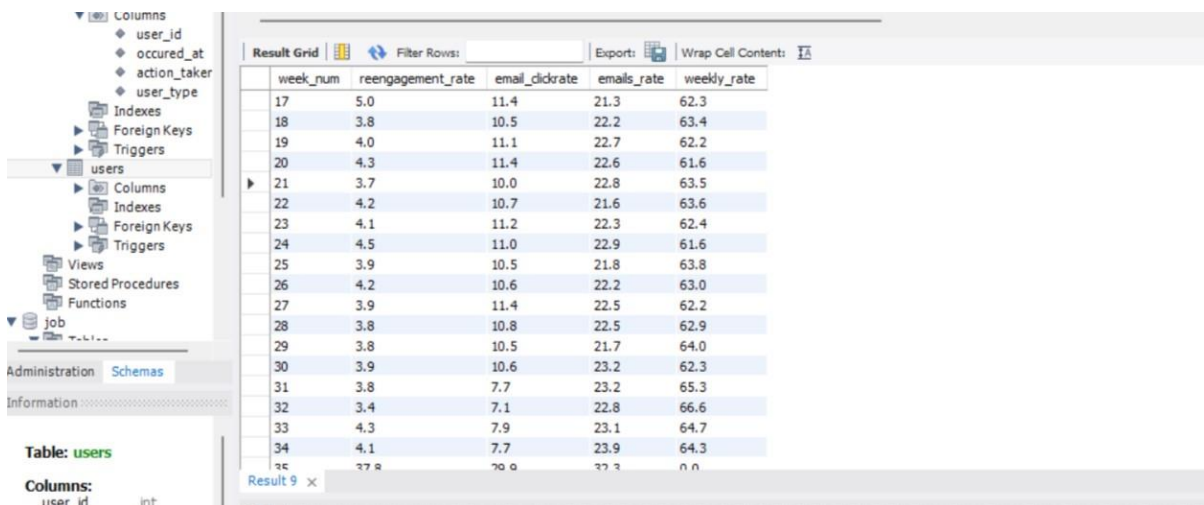
	S	T	U	V	W	X
1	Nexus 7 ▾	Nexus 5 ▾	Nexus 10 ▾	Nokia Lumia 635 ▾	Htc One ▾	Amazon Fire Phone ▾
2	18	40	16	17	16	4
3	30	73	30	33	19	9
4	41	87	25	23	30	12
5	32	103	22	22	29	11
6	29	91	25	25	21	5
7	45	96	27	25	24	5
8	36	88	45	31	20	16
9	49	87	38	35	20	11
10	51	89	29	37	21	13
11	46	87	29	42	23	13
12	40	84	37	31	27	10
13	39	85	26	35	26	6
14	45	77	25	43	31	12
15	62	84	36	34	31	12
16	38	69	24	28	13	14
17	25	67	30	28	18	12
18	30	70	23	27	19	14
19	33	70	25	17	25	11
20	2	4	2	2	2	0

4) Email Engagement :



The screenshot shows a SQL IDE interface with a Navigator on the left and a SQL editor on the right. The Navigator displays a schema named 'investigation' with tables 'event_emails' and 'events_emails'. The SQL editor contains a query that calculates engagement rates for different email actions over time.

```
1 select
2     week_num ,
3     round((reengagement_email/total*100),1) as reengagement_rate,
4     round((email_clickthrough/total*100),1) as email_clickrate,
5     round((email_open/total*100),1) as emails_rate,
6     round((weekly_digest/total*100),1) as weekly_rate
7 from (
8     select
9         extract(week from occurred_at) as week_num,
10        count(case when action_taken='sent_reengagement_email' then user_id else null end) as reengagement_email,
11        count(case when action_taken='email_open' then user_id else null end) as email_open,
12        count(case when action_taken='email_clickthrough' then user_id else null end) as email_clickthrough,
13        count(case when action_taken='sent_weekly_digest' then user_id else null end) as weekly_digest,
14        count(user_id) as total
15    from
16        events_emails
17    group by 1 )
18 sub group by 1
19 order by 1;
```



The screenshot shows the result grid of the query, displaying engagement metrics for each week. The columns are week_num, reengagement_rate, email_clickrate, emails_rate, and weekly_rate. The data is presented in a table with alternating light blue and white rows.

week_num	reengagement_rate	email_clickrate	emails_rate	weekly_rate
17	5.0	11.4	21.3	62.3
18	3.8	10.5	22.2	63.4
19	4.0	11.1	22.7	62.2
20	4.3	11.4	22.6	61.6
21	3.7	10.0	22.8	63.5
22	4.2	10.7	21.6	63.6
23	4.1	11.2	22.3	62.4
24	4.5	11.0	22.9	61.6
25	3.9	10.5	21.8	63.8
26	4.2	10.6	22.2	63.0
27	3.9	11.4	22.5	62.2
28	3.8	10.8	22.5	62.9
29	3.8	10.5	21.7	64.0
30	3.9	10.6	23.2	62.3
31	3.8	7.7	23.2	65.3
32	3.4	7.1	22.8	66.6
33	4.3	7.9	23.1	64.7
34	4.1	7.7	23.9	64.3

Results :

During this project I have learned how important operation analytics is and how metrics helps the operation team as well as investors in knowing actual insights and help them plan their further moves . I also got experience to use various advanced level sql queries which in turn would help me become a better analyst.