

## Project 3

# Operation analytics and investigating metric spikes

## Project description:-

To interpret the given into tables and the analyse the given questions/tasks. The datasets had data of the jobs based on which I have to derive certain insights.

## Approach:-

I have spent some time on understanding the data/table given.

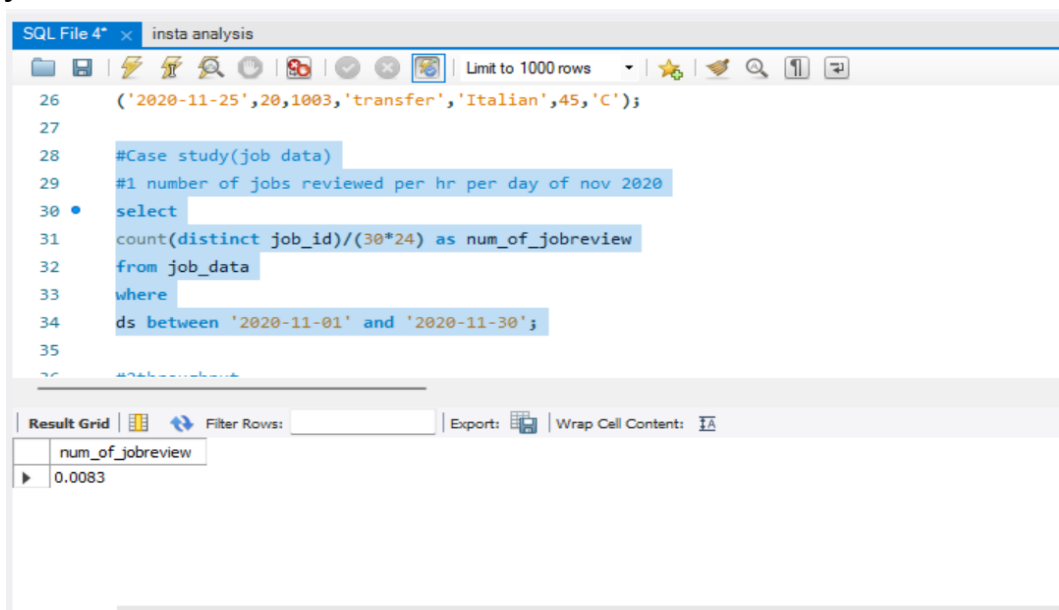
I just feed the data from the data sets into the tables in sql and then started deriving the useful insights from them. Starting with project 3 as my new database and tables using structures and tables provided.

## Execution:-

### Case study 1-Job data analysis

## Tasks

### A) jobs reviewed over time



The screenshot shows a SQL IDE window titled 'SQL File 4\*' with a sub-tab 'insta analysis'. The query editor contains the following SQL code:

```
26 ('2020-11-25',20,1003,'transfer','Italian',45,'C');
27
28 #Case study(job data)
29 #1 number of jobs reviewed per hr per day of nov 2020
30 • select
31 count(distinct job_id)/(30*24) as num_of_jobreview
32 from job_data
33 where
34 ds between '2020-11-01' and '2020-11-30';
35
36 #34k constant
```

The bottom panel shows the 'Result Grid' with the following data:

num_of_jobreview
0.0083

### B)Throughput analysis:-

It's the number of events happening per sec  
Here we are deriving 7 day rolling average of throughput.

The screenshot shows a SQL query in a file named 'insta analysis'. The query calculates a 7-day rolling average of throughput. It uses a subquery to count distinct job IDs per date, then applies a rolling average over 7 days.

```
36 #2throughput
37 • select ds, jobs_reviewed,
38 avg(jobs_reviewed) over (order by ds rows between 6 preceding and current row)
39 as throughput_7_rolling_avg
40 from
41 (select ds, count(distinct job_id) as jobs_reviewed
42 From job_data
43 where ds between '2020-11-01' and '2020-11-30'
44 group by ds
45 order by ds
46 )a;
```

The result grid shows the following data:

ds	jobs_reviewed	throughput_7_rolling_avg
2020-11-25	1	1.0000
2020-11-26	1	1.0000
2020-11-27	1	1.0000
2020-11-28	2	1.2500
2020-11-29	1	1.2000

### C)Language share analysis

Percentage share of each language in last 30 days

The screenshot shows a SQL query in a file named 'insta analysis'. The query calculates the percentage share of each language based on the number of jobs. It uses a subquery to count distinct job IDs per language, then cross joins it with the total count of jobs.

```
49 #percentage share of each language over last 30 days
50 • select language, num_jobs,
51 100.0* num_jobs/total_jobs as pct_share_jobs
52 from(
53 select language, count(distinct job_id) as num_jobs
54 from job_data
55 group by language) a
56 cross join(
57 select count(distinct job_id) as total_jobs
58 from job_data )b;
```

The result grid shows the following data:

language	num_jobs	pct_share_jobs
Arabic	1	16.66667
English	1	16.66667
French	1	16.66667
Hindi	1	16.66667
Italian	1	16.66667

## D) Duplicate row detection

To identify duplicate rows in the data

The screenshot shows a SQL editor window titled "SQL File 4\*" with a tab "insta analysis". The query is as follows:

```
59
60
61 #duplicate row detection
62 #identify duplicate rows in the data
63 select * from(
64   select *,
65   row_number()over(partition by job_id as rownum
66   from job_data
67   )a
68   where rownum>1;
69
```

Below the query, the "Result Grid" shows two rows of data:

ds	job_id	actor_id	event	language	time_spent	org	rownum
2020-11-28	23	1005	transfer	Persian	22	D	2
2020-11-26	23	1004	skip	Persian	56	A	3

The interface includes a "Filter Rows" field, an "Export" button, and a "Wrap Cell Content" option. A "Result Grid" button is visible on the right side.

## Case study 2

(Investigating metric spike):

### A) USER Engagement:

To measure the activeness of a user:

The screenshot shows a SQL editor window titled "project3\*" with a tab "insta analysis" and a sub-tab "email\_events". The query is as follows:

```
102
103 # case study 2
104 #1)user engagement
105 select
106   extract(week from occurred_at) as num_week,
107   count(distinct user_id) as no_of_distinct_user
108   from events group by num_week;
109
```

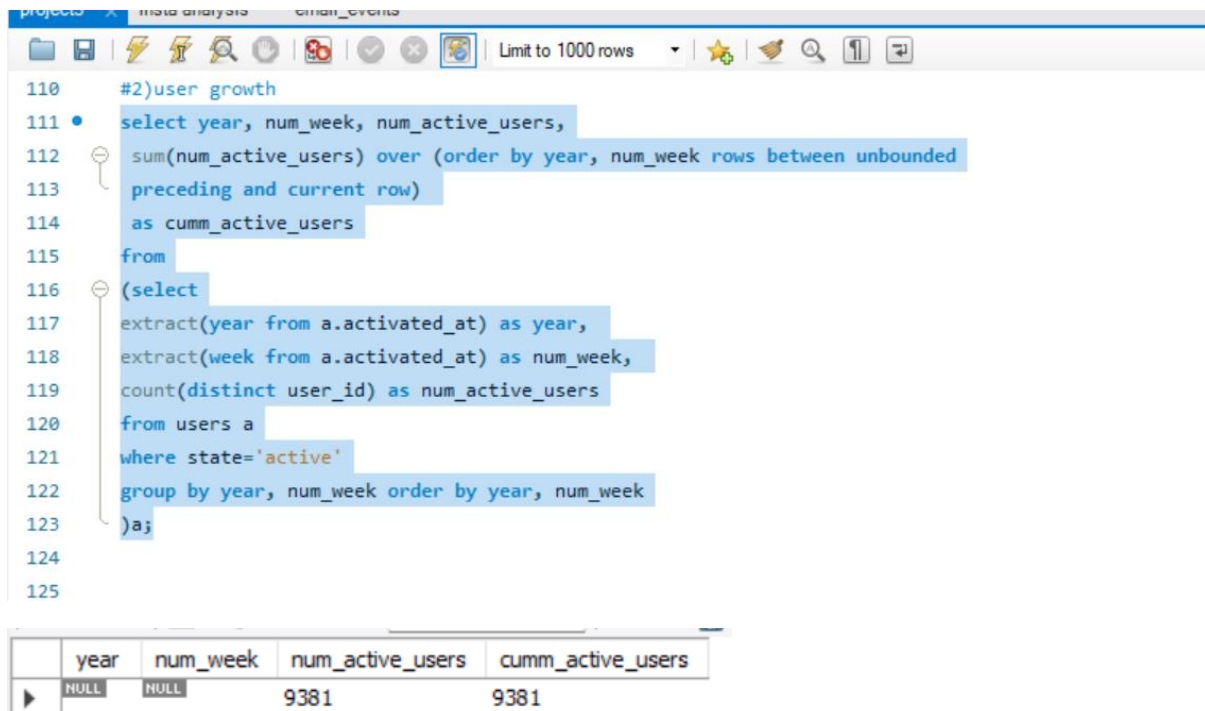
Below the query, the "Result Grid" shows one row of data:

num_week	no_of_distinct_user
NULL	4675

The interface includes a "Filter Rows" field, an "Export" button, and a "Wrap Cell Content" option. A "Result Grid" button is visible on the right side.

## B) User growth:-

Amount of users growing over time for a product.



The screenshot shows a SQL IDE window with a query editor and a results table. The query is for user growth, calculating cumulative active users over time. The results table shows the first row with NULL values for year and num\_week, and 9381 for both num\_active\_users and cumm\_active\_users.

```
110 #2)user growth
111 • select year, num_week, num_active_users,
112       sum(num_active_users) over (order by year, num_week rows between unbounded
113       preceding and current row)
114       as cumm_active_users
115 from
116 (select
117   extract(year from a.activated_at) as year,
118   extract(week from a.activated_at) as num_week,
119   count(distinct user_id) as num_active_users
120 from users a
121 where state='active'
122 group by year, num_week order by year, num_week
123 )a;
```

year	num_week	num_active_users	cumm_active_users
NULL	NULL	9381	9381

## C) Weekly retention:-

Users getting retained weekly after signing-up for a product

```
4
5
6 #3)weekly retention
7 • select count(user_id),
8       sum(case when retention_week =1 then 1 else 0 end) as
9       per_week_retention
10
11 from(
12   select a.user_id,
13   a.sign_up_week,
14   b.engagement_week,
15   b.engagement_week-a.sign_up_week as retention_week
16
```

```

136
137 from
138 (
139 (select distinct user_id, extract(week from occurred_at) as sign_up_week
140 from events
141 where event_type = 'signup_flow'
142 and event_name = 'complete_signup'
143 and extract(week from occurred_at)=18) a
144 left join
145 (select distinct user_id, extract(week from occurred_at) as engagement_week
146 from events
147 where event_type = 'engagement')b
148 on a.user_id = b.user_id
149 )
150 group by user_id
151 order by user_id;

```

#### D) Weekly engagement:-

To measure the activeness of a user. Measuring if the user finds quality in a product/service weekly.

```

#4) weekly engagement
select
extract(year from occurred_at) as year_num,
extract(week from occurred_at) as week_num,
device,
count(distinct user_id) as no_of_users
from events
where event_type='engagement'
group by 1,2,3
order by 1,2,3;

#5) email engagement

```

#### E) Email engagement:-

User engaging with the email service

```
project3* x insta analysis email_events
Limit to 1000 rows
#5) email engagement
select
100.0* sum(case when email_cat='email_opened' then 1 else 0 end)
/sum(case when email_cat='email sent' then 1 else 0 end) as email_opening_rate,
100.0*sum(case when email_cat = 'email_clicked' then 1 else 0 end)
/sum(case when email_cat='email_sent' then 1 else 0 end)
as email_clicking_rate
from
(select *,
case when action in ('sent weekly digest', 'sent_reengagement_email')
then 'email_sent'
when action in ('email_open')
then 'email_opened'
when action in ('email clickthrough')
then 'email_clicked'
end as email_cat
from tutorial.yammer_events
)a;
```

Tech stack used

MySQL version 8.0

Office excel

Insights

- 1) The number of distinct jobs reviewed per hr per day for nov 2020 is 83%

- 2) We used the 7-day rolling average of throughput as it gives the average for all the days right
- 3) from day 1 to day 7 whereas, daily metric gives the average for only that particular day itself. The percentage share of Persian language is the most (37.5%).
- 4) There are two duplicate rows if we partition the data by job\_id. But if we look the overall columns, all the rows are unique.

#### Case Study 2 (Investigating metric spike):

- 5) The weekly user engagement increased from week 18th to week 31st and then started declining from then onwards. This means that some of the users do not find much quality in the product/service in the last of the weeks.
- 6) There are in total 9381 active users from 1 week of 2013 to the 35th week of 2014.
- 7) The overall count of weekly engagement per device used is the most for MacBook users and iPhone users.
- 8) The email opening rate is around 34% and email clicking rate is around 15%. The users are engaging with the email service which is good for the company to expand.