Operation Analytics and Investigating Metric Spike

CASE STUDY 1 – Job Data

```
CREATE TABLE job_data
(
   ds DATE,
job_id INT,
actor_id INT,
   event VARCHAR(15) NOT NULL,
language VARCHAR(15) NOT
NULL,    time_spent INT NOT
NULL,    org CHAR(2),
   PRIMARY KEY(job_id,actor_id)
);
```

```
INSERT INTO job_data (ds, job_id, actor_id, event, language, time_spent, org)
VALUES ('2020-11-30', 21, 1001, 'skip', 'English', 15, 'A'),
   ('2020-11-30', 22, 1006, 'transfer', 'Arabic', 25, 'B'),
   ('2020-11-29', 23, 1003, 'decision', 'Persian', 20, 'C'),
   ('2020-11-28', 23, 1005,'transfer', 'Persian', 22, 'D'),
   ('2020-11-28', 25, 1002, 'decision', 'Hindi', 11, 'B'),
   ('2020-11-27', 11, 1007, 'decision', 'French', 104, 'D'),
   ('2020-11-26', 23, 1004, 'skip', 'Persian', 56, 'A'),
   ('2020-11-25', 20, 1003, 'transfer', 'Italian', 45, 'C');
```

A) **Number of Jobs Reviewed:** Amount of jobs reviewed over time.

```
SELECT COUNT (DISTINCT job_id)/(30*24) FROM job_data
WHERE ds BETWEEN '2020-11-01' AND '2020-11-30'
```

From the above query, it is observed that on 0.0111 jobs were reviewed per hour per day

B) **Throughput:** It is the no. of events happening per second.

```
SELECT ds,jobs_reviewed, AVG(jobs_reviewed) OVER (ORDER BY ds ROWS BETWEEN  6 PRECEDING AND CURRENT ROW) AS throughput
FROM ( SELECT ds, COUNT(distinct job_id) as jobs_reviewed from job_data
WHERE ds BETWEEN '2020-11-01' AND '2020-11-30' GROUP BY ds
ORDER BY ds ) a
```

Ds	Job_reviewed	Throughput
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
2020-11-30	2	1.3333

C) **Percentage share of each language:**

```
SELECT language, num_jobs, 100.0*num_jobs/tot_jobs AS per_jobs FROM (SELECT language, COUNT (DISTINCT job_id) AS num_jobs FROM job_data WHERE ds BETWEEN '2020-11-01' AND '2020-11-30' GROUP BY language) a CROSS JOIN ( SELECT COUNT(job_id) AS tot_jobs FROM job_data WHERE ds BETWEEN '2020-11-01' AND '2020-11-30' ) b
```

Output:

Language	Num_jobs	Per_jobs
Arabic	1	11.1111
English	1	11.1111
French	1	11.1111
Hindi	1	11.1111
Italian	1	11.1111
Italian	1	11.1111
Persian	3	33.3333

D) **Duplicate rows:**

```
SELECT * FROM (SELECT * , ROW_NUMBER() OVER (PARTITION BY ds, job_id, actor_id) AS rownum FROM job_data) a  WHERE job_id=23
```

Output:

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

CASE STUDY 2 – Investigating metric spike

Table-1 users create

table users(user_id

int primary key,

created_at

timestamp, state

varchar(10),

activated_at

timestamp,

company_id int,

langauge

varchar(10));

Table-2 Events create

table events (user_id int,

occured_at timestamp,

event_type varchar(10),

event_name varchar(20),

location varchar(10),

device varchar(10),

user_type int,

foreign key (user_id) references users1(user_id));

Table 3 – email-events

```
create table email_events(
user_id int, occured_at
timestamp, actions
varchar(20), user_type int,
    foreign key (user_id) references users1(user_id));
```

A) **User Engagement:**

```
SELECT EXTRACT (week, occurred_at) AS weeknum, COUNT
(DISTINCT user_id) FROM GROUP BY 1;
```

	Weeknum	Count
1	18	791
2	19	1244
3	20	1270
4	21	1341
5	22	1293
6	23	1366
7	24	1434
8	25	1462
9	26	1443
10	27	1477
11	28	1556
12	29	1556
13	30	1593
14	31	1685
15	32	1483
16	33	1438
17	34	1412
18	35	1442

B) **User Growth:**

```
SELECT year, weeknum, num_active_user, SUM
(num_active_user) OVER (ORDER BY year, weeknum ROWS BETWEEN
UNBOUNDED PRECEDING AND CURRENT ROW) AS cum_active_user
FROM ( SELECT EXTRACT (year FROM a.activated_at) AS year,
EXTRACT ( week FROM a.activated_at) AS weeknum, COUNT (
DISTINCT user_id) AS num_active_user FROM users a WHERE state =
'active' GROUP BY year, weeknum ORDER BY year, weeknum) a
```

	Year	weeknum	Num_active_user	Cum_active_user
1	2013	1	67	67
2	2013	2	29	96
3	2013	3	47	143
4	2013	4	36	179
5	2013	5	30	209
6	2013	6	48	257
7	2013	7	41	298
8	2013	8	39	337
9	2013	9	33	370
10	2013	10	43	413
11	2013	11	33	446
12	2013	12	32	478
13	2013	13	33	511
14	2013	14	40	551
15	2013	15	35	586
16	2013	16	42	628
17	2013	17	48	676
18	2013	18	48	724
19	2013	19	45	769
20	2013	20	55	824

C) **Weekly Retention:**

```
SELECT COUNT(user_id) , SUM( CASE WHEN retention_week = 1 THEN 1 ELSE 0 END) AS week1 FROM ( SELECT a.user_id, a.signup_week, b.engagement_week, b.engagement_week – a.signup_week AS retention_week FROM ( SELECT DISTINCT user_id, EXTRACT (week FROM occurred_at) AS signup_week FROM events WHERE event_type = 'signup_flow' AND event_name =
' complete_signup' AND EXTRACT (week FROM occurred_at) = 18 ) a LEFT JOIN ( SELECT DISTINCT user_id, EXTRACT (week FROM occurred_at) AS engagement_week FROM events WHERE event_type = 'engagement' ) b ON a.user_id = b.user_id ) ORDER BY a.user_id) c GROUP BY user_id LIMIT 100;
```

Count	Week1
1	0
1	0
2	1
3	0
5	1

2	1
1	0
3	1
2	1
6	1
2	1
6	1
10	1
2	1
2	1
1	0
1	0
2	1
6	0
3	0
2	1

D) **Weekly Engagement:**

```
SELECT EXTRACT (year FROM occurred_at) AS year,
EXTRACT(week FROM occurred_at) AS week, device, COUNT
(DISTINCT user_id) FROM events
WHERE event_type = 'engagement' GROUP BY 1,2,3 ORDER BY 1,2,3
```

Year	week	device
2014	18	Acer aspire desktop
2014	18	Acer aspire desktop
2014	18	Amazon fire phone
2014	18	Asus chromebook
2014	18	Dell inspiron desktop
2014	18	Dell inspiron notebook
2014	18	Hp pavilion desktop
2014	18	Htc one
2014	18	Ipad air
2014	18	Ipad mini
2014	18	Iphone 4s
2014	18	Iphone 5
2014	18	Iphone 5s
2014	18	Kindle fire
2014	18	Lenovo thinkpad
2014	18	Macbook air
2014	18	Macbook pro
2014	18	Mac mini
2014	18	Nexus 10
2014	18	Nexus 5
2014	18	Nexus 7

E) Email Engagement:

```
SELECT 100.0 * SUM ( CASE WHEN email_cat = 'email_open' THEN 1
ELSE 0 END) / SUM ( CASE WHEN email_cat = 'email_sent' THEN 1 ELSE 0 END) AS email_open_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_clicked_rate FROM ( SELECT
* , CASE WHEN action IN ('sent_weekly_digest', 'sent_reengagement_email') THEN 'email_sent' WHEN action IN ('email_open') THEN 'email_open' WHEN action in ('email_clickthrough')
THEN 'email_clicked' END AS email_cat FROM emails_event ) a LIMIT 100;
```

Email_open_rate	Email_clicked_rate
33.5834	14.7899

PROJECT DESCRIPTION:

Operation Analytics is the complete end to end operations done for the company. With the help of the areas on which it must improve upon. Operation Analyst work closely with the ops team, support team, marketing team, etc and help them derive insights out of the data they collect.
And further used to predict the overall growth or decline of a company's fortune. Which means better automation, better understanding between cross-functional teams, and more effective workflows.

Investigating metric spike is also an important part of operation analytics as being a Data Analyst we must be able to understand or make other teams understand questions like- Why is there a dip in daily engagement? Why have sales taken a dip? Etc. Questions like these must be answered daily and for that its very important to investigate metric spike.

APPROACH:

We plan to execute SQL queries on given database to create insights for the teams to make data driven decision. The SQL queries will give the following insights:

A) Case Study 1 (Job Data):

1.Number of jobs reviewed: We calculated the number of jobs reviewed per hour per day for November 2020.
2. Throughput: We calculated 7 day rolling average of throughput. Because, rolling averages are useful for finding long-term trends otherwise disguised by occasional fluctuations.
3. Percentage share of each language: We calculated the percentage share of each language in the last 30 days.
4. Duplicate Rows: We displayed duplicates from the table.

B) Case Study 2 (Investigating metric spike):

1. User Engagement: We calculated the weekly user engagement.
2. User Growth: We calculated the user growth for product.
3. Weekly Retention: We calculated the weekly retention of users-sign up cohort.
4. Weekly Engagement: We calculated the weekly engagement per device.
5. Email Engagement: We calculated the email engagement metrics.

TECH STACK USED

The software that's help to used to make this project

MySQL workbench 8.0 CE Version 8.0.31 build 2235049 CE (64 bits)