Operation Analytics and Investigating Metric Spike

Project Description: -

•The given project consists of 2 case studies: -

•**First** is regarding Operation Analytics where job data is provided and number of jobs reviewed, 7day rolling average of throughput, percentage share of language used and duplicates are found out.

•**Second** is Investigating Metric Spike were user engagement, user growth, weekly retention, weekly engagement, and email engagement is determined.

•The following information is found with the help of SQL queries.

Approach: -

The required information was determined via SQL queries where the data base was created first in SQL and moreover for the second case study due to the size of the data excel was used to make charts for better visualisation.

Tech Stack Used: -

•MySQL was used to run the queries.

•The language was selected because of comfort and experience in the same.

•MS Excel was used in the second case study for better visualisation.

•As I am currently learning this tool, it was utilised to get more hands on experience.

Case Study 1: Job Data Analysis -

 # job_data1: -

```
create table job_data1(ds date, job_id int not null,actor_id int not null, event varchar(20)not null, language varchar(20)not null, time_spend int not null, org char(2));


select * from job_data1;


INSERT INTO job_data1(ds, job_id, actor_id, event, language, time_spend,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');
```

Output: -

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

ds	job_id	actor_id	event	language	time_spend	org
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

Tasks:-

1.Jobs Reviewed Over Time: -

Calculate the number of jobs reviewed per hour for each day in November 2020.

```
select * from job_data1;
select avg(t) as 'avg jobs reviewed per day per hour',
avg(p) as 'avg jobs reviewed per day per second'
from
(Select
ds,
((count(job_id) *3600)/sum(time_spend)) as t,
((count(job_id))/sum(time_spend)) as p
from
job_data1 where month(ds)=11 group by ds) a;
```

Output: -

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

avg jobs reviewed per day per hour	avg jobs reviewed per day per second
126.18048333	0.03505000

2.Throughput Analysis-

Calculate the 7-day rolling average of throughput (number of events per second).

select round(count(event)/sum(time_spend), 2) as "weekly throughput" from job_data1;

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

	weekly throughput
▶	0.03

select ds as dates, round(count(event)/sum(time_spend), 2) as "daily throughput" from job_data1 group by ds order by ds;

Output: -

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

	dates	daily throughput
▶	2020-11-25	0.02
	2020-11-26	0.02
	2020-11-27	0.01
	2020-11-28	0.06
	2020-11-29	0.05
	2020-11-30	0.05

➔ I would prefer 7- day rolling throughput (weekly throughput) as the result or output will show faster without any syntax error occurred in it and it is easy to understand.

3.Language Share Analysis-

Calculate the percentage share of each language in the last 30 days.

select language as language,round(100 * count(*)/total,2) as percentage, sub.total from job_data1 cross join (select count(*) as total from job_data1) as sub group by language, sub.total;

Output: -

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

language	percentage	total
English	12.50	8
Arabic	12.50	8
Persian	37.50	8
Hindi	12.50	8
French	12.50	8
Italian	12.50	8

4.Duplicate rows detection-

To display duplicate rows from the job_data table.

select actor_id, count (*) as duplicates from job_data1 group by actor_id having count (*) > 1;

Output: -

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

actor_id	duplicates
1003	2

Case Study 2: Investigating Metric Spike –

<u>Working with Three Tables</u>: -

#Table:1. users -

create table users (user_id int, created_at varchar (100), company_id int, language varchar (50), activated_at varchar (100), state varchar (50));

SHOW VARIABLES LIKE 'secure_file_priv';

LOAD DATA INFILE "C:/ProgramData/MySQL/MySQL Server 8.0/Uploads/users.csv"

INTO TABLE users

FIELDS TERMINATED BY ','

ENCLOSED BY ""

LINES TERMINATED BY '\n'

IGNORE 1 ROWS;

select * from users;

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

user_id	created_at	company_id	language	activated_at	state
0	01-01-2013 20:59	5737	english	01-01-2013 21:01	active
3	01-01-2013 18:40	2800	german	01-01-2013 18:42	active
4	01-01-2013 14:37	5110	indian	01-01-2013 14:39	active
6	01-01-2013 18:37	11699	english	01-01-2013 18:38	active
7	01-01-2013 16:19	4765	french	01-01-2013 16:20	active
8	01-01-2013 04:38	2698	french	01-01-2013 04:40	active
11	01-01-2013 08:07	3745	english	01-01-2013 08:09	active
13	02-01-2013 12:27	4025	english	02-01-2013 12:29	active
15	02-01-2013 15:39	4259	english	02-01-2013 15:41	active

#Table: 2. Events-

```
SHOW VARIABLES LIKE 'secure_file_priv';


LOAD DATA INFILE "C:/ProgramData/MySQL/MySQL Server 8.0/Uploads/events.csv"

INTO TABLE events

FIELDS TERMINATED BY ','

ENCLOSED BY ""

LINES TERMINATED BY '\n'

IGNORE 1 ROWS;


desc events;


select * from events;
```

Output: -

Result Grid | Filter Rows: | Export: | Wrap Cell Content: | Fetch rows:

user_id	occurred_at	event_type	event_name	location	device	user_type
10522	02-05-2014 11:02	engagement	login	Japan	dell inspiron notebook	3
10522	02-05-2014 11:02	engagement	home_page	Japan	dell inspiron notebook	3
10522	02-05-2014 11:03	engagement	like_message	Japan	dell inspiron notebook	3
10522	02-05-2014 11:04	engagement	view_inbox	Japan	dell inspiron notebook	3
10522	02-05-2014 11:03	engagement	search_run	Japan	dell inspiron notebook	3
10522	02-05-2014 11:03	engagement	search_run	Japan	dell inspiron notebook	3
10612	01-05-2014 09:59	engagement	login	Netherlands	iphone 5	1
10612	01-05-2014 10:00	engagement	like_message	Netherlands	iphone 5	1

#Table:3. emailEvents-

```
create table emailEvents(user_id int, occured_at varchar(100), action
varchar(100), user_type int);


SHOW VARIABLES LIKE 'secure_file_priv';


LOAD DATA INFILE "C:/ProgramData/MySQL/MySQL Server
8.0/Uploads/email_events.csv"

INTO TABLE emailEvents

FIELDS TERMINATED BY ','

ENCLOSED BY ""

LINES TERMINATED BY '\n'

IGNORE 1 ROWS;

select * from emailEvents;
```

Output: -

Result Grid | Filter Rows: | Export:

user_id	occured_at	action	user_type
0	06-05-2014 09:30	sent_weekly_digest	1
0	13-05-2014 09:30	sent_weekly_digest	1
0	20-05-2014 09:30	sent_weekly_digest	1
0	27-05-2014 09:30	sent_weekly_digest	1
0	03-06-2014 09:30	sent_weekly_digest	1
0	03-06-2014 09:30	email_open	1
0	10-06-2014 09:30	sent_weekly_digest	1
0	10-06-2014 09:30	email_open	1

Tasks: -

1.Weekly User Engagement-

calculate the weekly user engagement.

```
select extract(week from occurred_at) as weeknum,
count(distinct user_id)
from project.events
where event_type='engagement‘
group by weeknum;
```

Output: -

Result Grid | Filter Rows:

	weeknum	count(distinct user_id)
▶	NULL	6142

2.User Growth Analysis-

calculate the user growth for the product.

```
select year,weeknum,num_active_users,
sum(num_active_users) over
(order by year,weeknum rows between
unbounded preceding and current row) cum_active_users
from(select extract(year from a.activated_at)as year,
extract(week from a.activated_at) as weeknum,
count(distinct user_id)num_active_users from users a
where state='active‘ group by weeknum,year)a;
```

Output: -

A	B	C	D
year	weeknum	num_active_use	cum_active_users
2013	1	67	67
2013	2	29	96
2013	3	47	143
2013	4	36	179
2013	5	30	209
2013	6	48	257
2013	7	41	298
2013	8	39	337
2013	9	33	370
2013	10	43	413
2013	11	33	446
2013	12	32	478
2013	13	33	511
2013	14	40	551
2013	15	35	586
2013	16	42	628
2013	17	48	676
2013	18	48	724
2013	19	45	769
2013	20	55	824
2013	21	41	865
2013	22	49	914

3. Weekly Retention Analysis-

calculate the weekly retention of users based on their sign-up cohort.

select count(user_id),sum(case when retention_week-1 then 1 else 0 end) as week_1 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 = 'complate 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)a;

Output: -

	count	week_1
1	317	64

4. Weekly Engagement Per Device-

calculate the weekly engagement per device.

select extract(week from occurred_at) as week,

extract(year from occurred_at) as year,device, count(distinct user_id) as count

from events where event_type='engagement‘

group by 1,2,3

order by 1,2,3;

Output: -

Result Grid | Filter Rows:

week	year	device	count
NULL	NULL	acer aspire desktop	198
NULL	NULL	acer aspire notebook	338
NULL	NULL	amazon fire phone	89
NULL	NULL	asus chromebook	355
NULL	NULL	dell inspiron desktop	360
NULL	NULL	dell inspiron notebook	677
NULL	NULL	hp pavilion desktop	339
NULL	NULL	htc one	196
NULL	NULL	ipad air	478
NULL	NULL	ipad mini	292
NULL	NULL	iphone 4s	409
NULL	NULL	iphone 5	1025
NULL	NULL	iphone 5s	626
NULL	NULL	kindle fire	205

Result Grid | Filter Rows:

week	year	device	count
NULL	NULL	kindle fire	205
NULL	NULL	lenovo thinkpad	1309
NULL	NULL	mac mini	150
NULL	NULL	macbook air	950
NULL	NULL	macbook pro	1952
NULL	NULL	nexus 10	273
NULL	NULL	nexus 5	621
NULL	NULL	nexus 7	355
NULL	NULL	nokia lumia 635	211
NULL	NULL	samsumg galaxy tablet	107
NULL	NULL	samsung galaxy note	119
NULL	NULL	samsung galaxy s4	803
NULL	NULL	windows surface	182

5. Email Engagement Analysis-

calculate the email engagement metrics.

select 100 * 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 * 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_click_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 emailevents)sub

Output: -

email_open_rate	email_click_rate
31.1921	10.4745

Result: -

Operational analytics is the process of using data analysis and business intelligence to improve efficiency and streamline everyday operations in real time.

A subset of business analytics, operational analytics is supported by data mining, artificial intelligence, and machine learning.