OPERATION ANALYTICS AND INVESTIGATING METRIC SPIKE

ADVANCED SQL

Submitted by:
Mahalakshmi Sreenivasan

A) Case Study 1 (Job Data)

Creation of Table: **job_data**

```
mysql> create table job_data(ds date, job_id integer, actor_id integer, event varchar(15), language varchar(20), time_spent integer, org char(1));
Query OK, 0 rows affected (0.27 sec)

mysql> insert into job_data values('2020-11-30',21,1001,'skip','English', 15, 'A');
Query OK, 1 row affected (0.04 sec)

mysql> insert into job_data values('2020-11-30',22,1006,'transfer','Arabic', 25, 'B');
Query OK, 1 row affected (0.01 sec)

mysql> insert into job_data values('2020-11-29',23,1003,'decision','Persian', 20, 'C');
Query OK, 1 row affected (0.01 sec)

mysql> insert into job_data values('2020-11-28',23,1005,'transfer','Persian', 22, 'D');
Query OK, 1 row affected (0.00 sec)

mysql> insert into job_data values('2020-11-28',25,1002,'decision','Hindi', 11, 'B');
Query OK, 1 row affected (0.01 sec)

mysql> insert into job_data values('2020-11-27',11,1007,'decision','French', 104, 'D');
Query OK, 1 row affected (0.01 sec)

mysql> insert into job_data values('2020-11-26',23,1004,'skip','Persian', 56, 'A');
Query OK, 1 row affected (0.01 sec)

mysql> insert into job_data values('2020-11-25',20,1003,'transfer','Italian', 45, 'C');
Query OK, 1 row affected (0.01 sec)
```

```
mysql> select*from job_data;
+------------+--------+----------+----------+----------+------------+------+
| ds         | job_id | actor_id | event    | language | time_spent | 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    |
+------------+--------+----------+----------+----------+------------+------+
8 rows in set (0.00 sec)
```

A. **Number of jobs reviewed:** Amount of jobs reviewed over time.

Your task: Calculate the number of jobs reviewed per hour per day for November 2020?

QUERY

select ds, count(job_id) JobsPerDay, sum(time_spent)/3600 HoursPerDay from job_data where ds>='2020-11-01' and ds<='2020-11-30' group by ds;

ds	JobsPerDay	HoursPerDay
2020-11-30	2	0.0111
2020-11-29	1	0.0056
2020-11-28	2	0.0092
2020-11-27	1	0.0289
2020-11-26	1	0.0156
2020-11-25	1	0.0125

6 rows in set (0.01 sec)

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

Your task: Let's say the above metric is called throughput. Calculate 7 day rolling average of throughput? For throughput, do you prefer daily metric or 7-day rolling and why?

QUERY

select ds,no_of_jobs,avg(no_of_jobs) over(order by ds ROWS BETWEEN UNBOUNDED PRECEDING AND CURRENT ROW) as throughput_rolling_avg from(select ds, count(distinct job_id) as no_of_jobs from job_data group by ds order by ds) a;

ds	no_of_jobs	throughput_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
2020-11-30	2	1.3333

6 rows in set (0.00 sec)

C. **Percentage share of each language:** Share of each language for different contents.

Your task: Calculate the percentage share of each language in the last 30 days?

QUERY

select language, 100*(count(language)/(select count(*) from job_data)) as percentage from job_data group by language;

language	percentage
English	12.5000
Arabic	12.5000
Persian	37.5000
Hindi	12.5000
French	12.5000
Italian	12.5000

6 rows in set (0.00 sec)

D. **Duplicate rows:** Rows that have the same value present in them.

Your task: Let's say you see some duplicate rows in the data. How will you display duplicates from the table?

QUERY

```
select* from(select *, ROW_NUMBER() OVER(partition by job_id) duplicate from job_data)a where duplicate>1;
```

```
+------------+--------+----------+----------+----------+------------+------+-----------+
| ds         | job_id | actor_id | event    | language | time_spent | org  | duplicate |
+------------+--------+----------+----------+----------+------------+------+-----------+
| 2020-11-28 |     23 |     1005 | transfer | Persian  |         22 | D    |         2 |
| 2020-11-26 |     23 |     1004 | skip     | Persian  |         56 | A    |         3 |
+------------+--------+----------+----------+----------+------------+------+-----------+
2 rows in set (0.00 sec)
```

B) Case Study 2 (Investigating Metric Spike)

A. **User Engagement:** To measure the activeness of a user. Measuring if the user finds quality in a product/service.

Your task: Calculate the weekly user engagement?

QUERY

select weekofyear(created_at) weeks, count(user_id) as no_of_users from users where state='active' group by weeks;

weeks	no_of_users
1	26
2	29
3	47
4	36
5	30
6	48
7	41
8	39
9	33
10	43
11	33
12	32
13	33
14	40
15	35
16	42
17	48
18	48
19	45
20	55
21	41
22	49
23	51
24	51
25	46

B. **User Growth:** Amount of users growing over time for a product.

Your task: Calculate the user growth for product?

QUERY

select date(created_at) day, count(distinct user_id) user_growth from users group by week(created_at);

day	user_growth
2013-01-01	52
2013-01-06	68
2013-01-13	76
2013-01-20	77
2013-01-27	75
2013-02-03	87
2013-02-10	80
2013-02-17	83
2013-02-24	81
2013-03-03	84
2013-03-10	88
2013-03-17	95
2013-03-24	92
2013-03-31	86
2013-04-07	96
2013-04-14	93
2013-04-21	100
2013-04-28	102
2013-05-05	105
2013-05-12	108
2013-05-19	104
2013-05-26	113
2013-06-02	112
2013-06-09	116
2013-06-16	118
2013-06-23	127
2013-06-30	127
2013-07-07	132
2013-07-14	141
2013-07-21	130
2013-07-28	141

C. **Weekly Retention:** Users getting retained weekly after signing-up for a product.

Your task: Calculate the weekly retention of users-sign up cohort?

QUERY

select weekofyear(u.created_at) week, count(e.user_id) retained_users from users u left join events e on u.user_id = e.user_id where u.state='active' group by week order by week;

week	retained_users
1	0
2	0
3	0
4	0
5	0
6	0
7	0
8	0
9	0
10	0
11	0
12	0
13	0
14	6
15	15
16	60
17	175
18	935
19	1516
20	1773

D. **Weekly Engagement:** To measure the activeness of a user. Measuring if the user finds quality in a product/service weekly.

Your task: Calculate the weekly engagement per device?

QUERY

select weekofyear(u.created_at) week, e.device, count(u.user_id) users from events e right join users u on e.user_id=u.user_id where e.event_type='engagement' group by week,e.device order by week;

week	device	users
14	dell inspiron notebook	6
15	iphone 4s	3
15	iphone 5	12
16	ipad mini	7
16	iphone 4s	9
16	iphone 5s	10
16	kindle fire	3
16	macbook air	8
16	macbook pro	11
16	nexus 5	2
16	nexus 7	8
16	windows surface	2
17	acer aspire notebook	15
17	asus chromebook	12
17	dell inspiron notebook	11
17	ipad mini	23
17	iphone 5	9
17	iphone 5s	4
17	lenovo thinkpad	14
17	macbook air	33
17	macbook pro	9
17	nexus 5	4
17	samsumg galaxy tablet	12
17	samsung galaxy s4	29
18	acer aspire desktop	9
18	acer aspire notebook	18
18	amazon fire phone	11
18	asus chromebook	11
18	dell inspiron desktop	17
18	dell inspiron notebook	27
18	hp pavilion desktop	17
18	htc one	4
18	ipad air	9

E. **Email Engagement:** Users engaging with the email service.

Your task: Calculate the email engagement metrics?

QUERY

select action, week(occurred_at) week, count(distinct user_id) engagement_count from email_events group by action, week order by action,week;

action	week	engagement_count
email_clickthrough	18	1
email_clickthrough	19	1
email_clickthrough	20	2
email_clickthrough	22	3
email_clickthrough	23	1
email_clickthrough	24	3
email_clickthrough	25	2
email_clickthrough	26	1
email_clickthrough	27	2
email_clickthrough	30	3
email_clickthrough	31	1
email_clickthrough	32	1
email_clickthrough	33	1
email_open	18	2
email_open	19	1
email_open	20	5
email_open	21	4
email_open	22	6
email_open	23	5
email_open	24	5
email_open	25	3
email_open	26	4
email_open	27	6
email_open	28	3
email_open	29	2
email_open	30	8
email_open	31	4
email_open	32	6
email_open	33	4
email_open	34	6
sent_weekly_digest	17	1
sent_weekly_digest	18	15
sent_weekly_digest	19	15
sent_weekly_digest	20	15