PROJECT 3: OPERATIONAL ANALYTICS AND INVESTIGATING METRIC SPIKE

PROJECT DESCRIPTION:

Operational Analytics is a crucial process that involves analyzing a company's end-to-end operations. This analysis helps identify areas for improvement within the company. As a Data Analyst, I collabroted with such as operations, support, and marketing, helping them derive valuable insights from the data they collect.
In this project I have focused on two main studies. The first is **job data analysis**, aim to analyse opterional related to job. The second one is **Investigating metric spike** , seeks to understand users engagement and activity pattern. In both cases I used advanced mysql skills to derived the knowledge, insight from dataset.

PROJECT APPROACH:

IN this I create database and table according to the given structure and I used a advanced mysql skills to solve the given query . In second project most we same kind query like group by, order by, aggregation function like count,sum etc. but I still think in this project biggest task is to import data in to mysql database while importing database I have been through lot of vedio but none of them couldn't help then I find the solution on quora. I had the server connectivity issue it took almost more than 2 days to import data because of that I'm never gonna forget this project but love to work in this project because I learned so many new things with this project.

TECH-STACK USED

the tech stack I used while making a project is MySQL community server with version 8.0.34 , for creating the database, also used the excel to import and export the file.

PROJECT INSIGHTS:

1) CASE STUDY 1 : JOB DATA ANALYSIS

Creating a table named job_data with the following columns:

- **job_id:** Unique identifier of jobs
- **actor_id:** Unique identifier of actor
- **event:** The type of event (decision/skip/transfer).
- **language:** The Language of the content
- **time_spent:** Time spent to review the job in seconds.
- **org:** The Organization of the actor
- **ds:** The date in the format yyyy/mm/dd (stored as text).

Query for creating a database;

```
Create database project_no_3;

Use project_no_3;
```

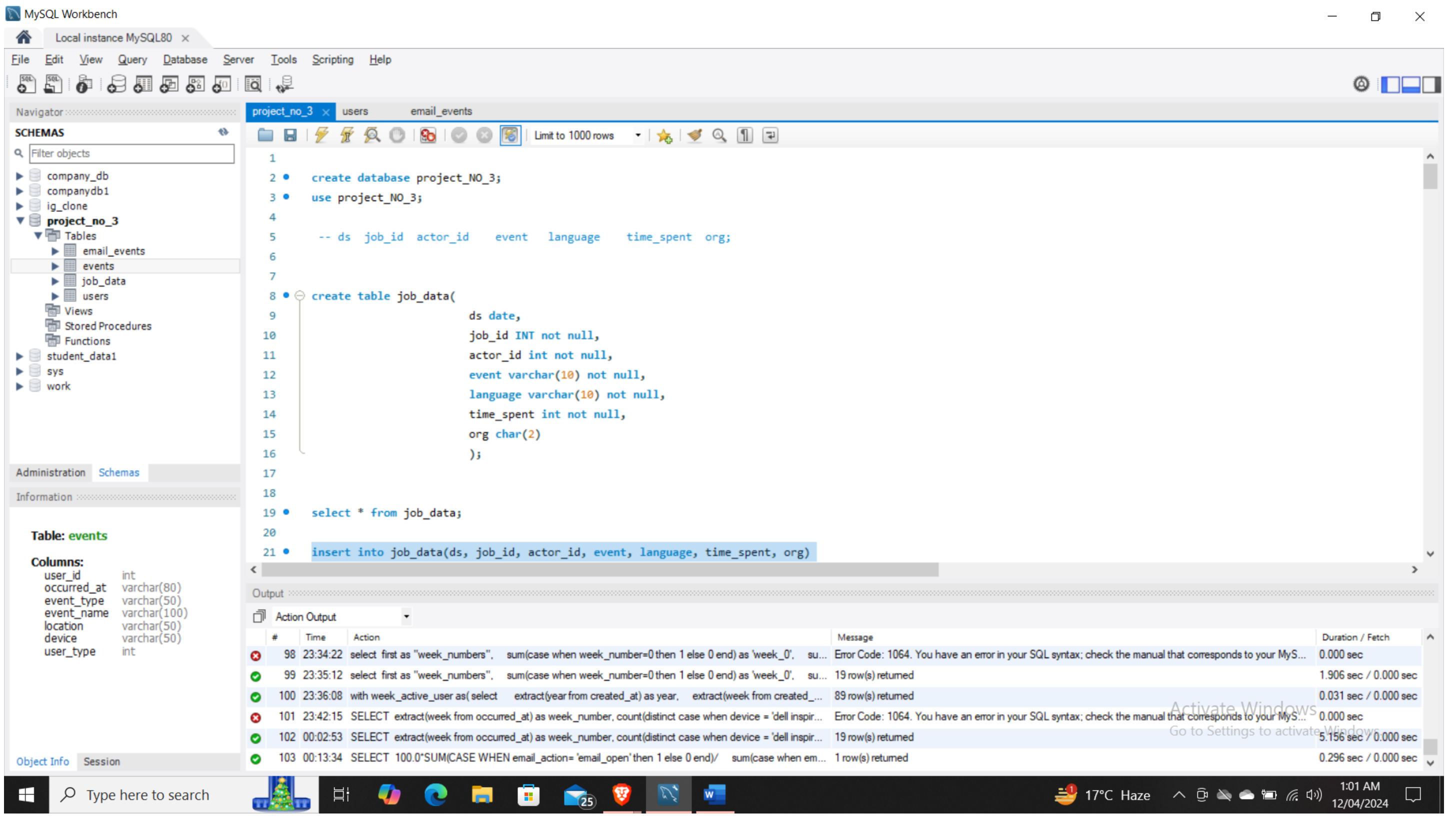
Query for creating a table;

```
create table job_data(
                          ds date,
                        job_id INT not null,
                        actor_id int not null,
                        event varchar(10) not null,
                        language varchar(10) not null,
                        time_spent int not null,
                        org char(2)
                        );
```

Query for inserting the data into table;

```
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');
```

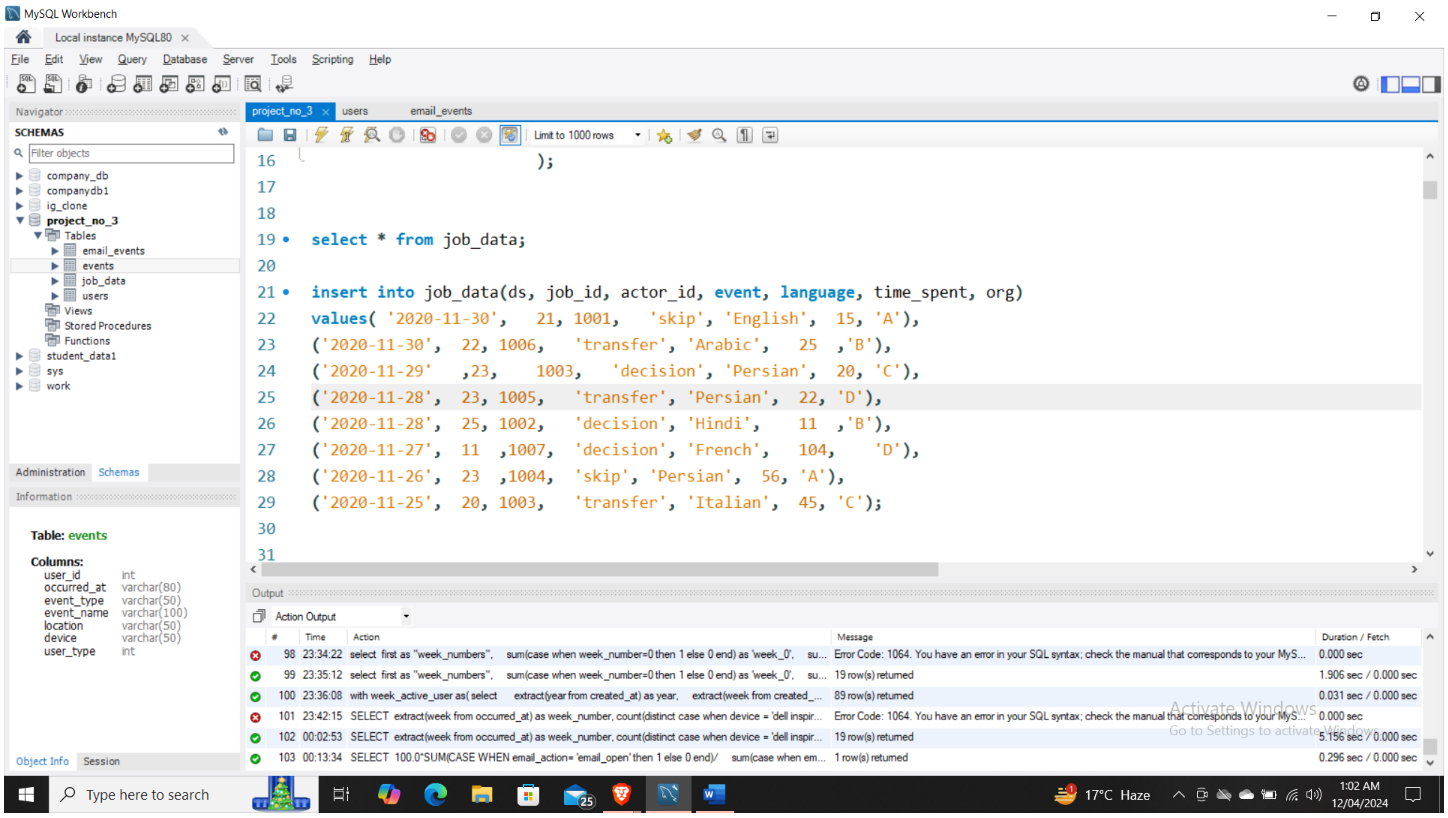
RESULT; for creating a new database and table:



```sql

create database project_NO_3;
use project_NO_3;

 -- ds  job_id  actor_id   event   language   time_spent  org;


create table job_data(
                        ds date,
                        job_id INT not null,
                        actor_id int not null,
                        event varchar(10) not null,
                        language varchar(10) not null,
                        time_spent int not null,
                        org char(2)
                        );


select * from job_data;

insert into job_data(ds, job_id, actor_id, event, language, time_spent, org)
```

```
#    Time      Action                                                                                   Message                                                                          Duration / Fetch
98   23:34:22  select first as "week_numbers",  sum(case when week_number=0 then 1 else 0 end) as 'week_0',  su...  Error Code: 1064. You have an error in your SQL syntax; check the manual that corresponds to your MyS...  0.000 sec
99   23:35:12  select first as "week_numbers",  sum(case when week_number=0 then 1 else 0 end) as 'week_0',  su...  19 row(s) returned                                                               1.906 sec / 0.000 sec
100  23:36:08  with week_active_user as( select   extract(year from created_at) as year,   extract(week from created_...  89 row(s) returned                                         0.031 sec / 0.000 sec
101  23:42:15  SELECT extract(week from occurred_at) as week_number, count(distinct case when device = 'dell inspir...  Error Code: 1064. You have an error in your SQL syntax; check the manual that corresponds to your MyS...  0.000 sec
102  00:02:53  SELECT extract(week from occurred_at) as week_number, count(distinct case when device = 'dell inspir...  19 row(s) returned                                          5.156 sec / 0.000 sec
103  00:13:34  SELECT 100.0*SUM(CASE WHEN email_action= 'email_open' then 1 else 0 end)/   sum(case when em...  1 row(s) returned                                                   0.296 sec / 0.000 sec
```

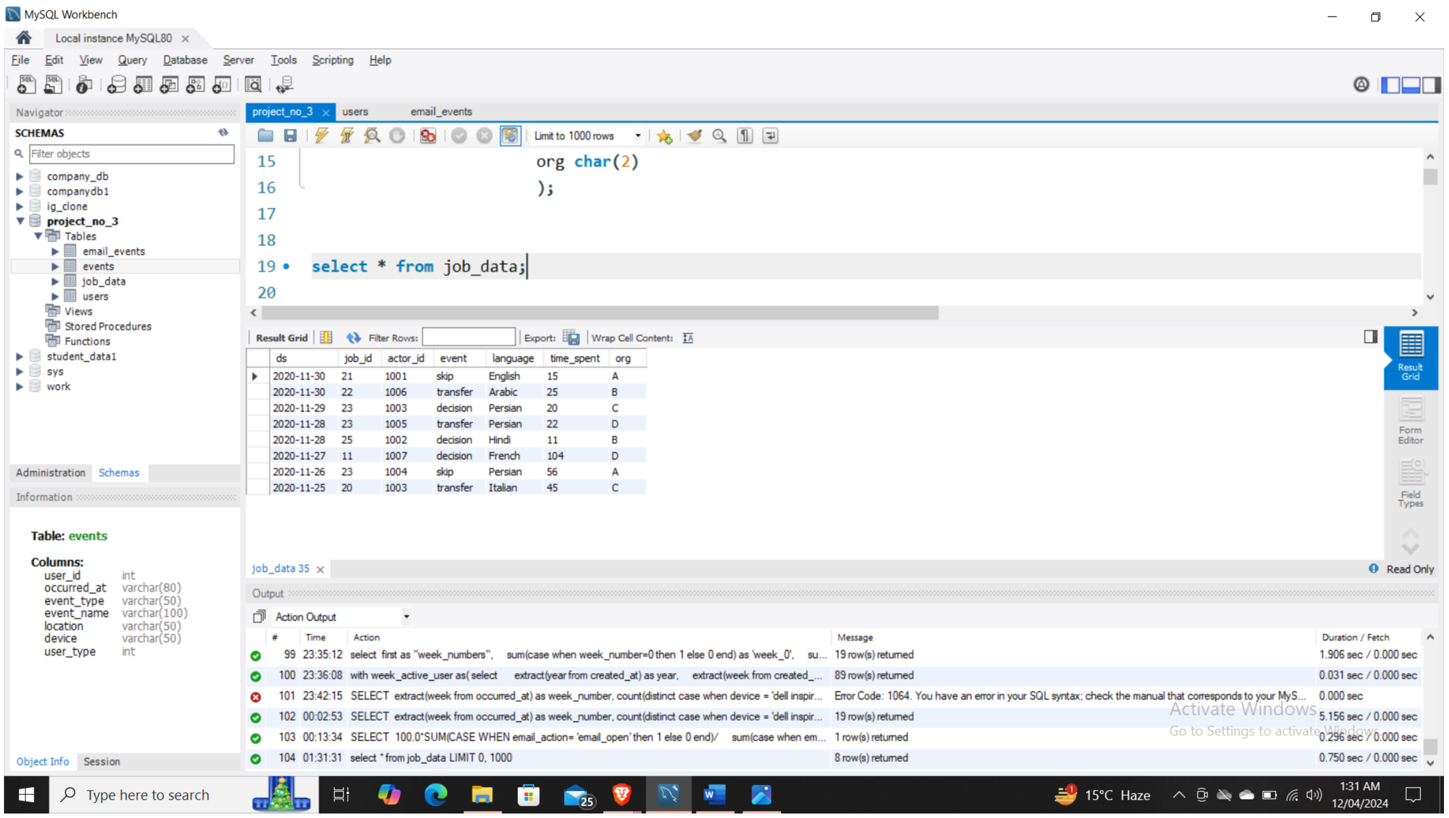
Result: query for inserting the data into job_table:



```sql
                        );


select * from job_data;

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');

```

```
#    Time      Action                                                                                   Message                                                                          Duration / Fetch
98   23:34:22  select first as "week_numbers",  sum(case when week_number=0 then 1 else 0 end) as 'week_0',  su...  Error Code: 1064. You have an error in your SQL syntax; check the manual that corresponds to your MyS...  0.000 sec
99   23:35:12  select first as "week_numbers",  sum(case when week_number=0 then 1 else 0 end) as 'week_0',  su...  19 row(s) returned                                                               1.906 sec / 0.000 sec
100  23:36:08  with week_active_user as( select   extract(year from created_at) as year,   extract(week from created_...  89 row(s) returned                                         0.031 sec / 0.000 sec
101  23:42:15  SELECT extract(week from occurred_at) as week_number, count(distinct case when device = 'dell inspir...  Error Code: 1064. You have an error in your SQL syntax; check the manual that corresponds to your MyS...  0.000 sec
102  00:02:53  SELECT extract(week from occurred_at) as week_number, count(distinct case when device = 'dell inspir...  19 row(s) returned                                          5.156 sec / 0.000 sec
103  00:13:34  SELECT 100.0*SUM(CASE WHEN email_action= 'email_open' then 1 else 0 end)/   sum(case when em...  1 row(s) returned                                                   0.296 sec / 0.000 sec
```

Table after insertion the data:



```
                    org char(2)
                    );

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

Table: events

Columns:

user_id	int
occurred_at	varchar(80)
event_type	varchar(50)
event_name	varchar(100)
location	varchar(50)
device	varchar(50)
user_type	int

```
#    Time      Action                                                                                   Message                                                                           Duration / Fetch
99   23:35:12  select first as "week_numbers",  sum(case when week_number=0 then 1 else 0 end) as 'week_0',  su...  19 row(s) returned                                                    1.906 sec / 0.000 sec
100  23:36:08  with week_active_user as( select  extract(year from created_at) as year,  extract(week from created_...  89 row(s) returned                                         0.031 sec / 0.000 sec
101  23:42:15  SELECT extract(week from occurred_at) as week_number, count(distinct case when device = 'dell inspir...  Error Code: 1064. You have an error in your SQL syntax; check the manual that corresponds to your MyS...  0.000 sec
102  00:02:53  SELECT extract(week from occurred_at) as week_number, count(distinct case when device = 'dell inspir...  19 row(s) returned                                         5.156 sec / 0.000 sec
103  00:13:34  SELECT 100.0*SUM(CASE WHEN email_action= 'email_open' then 1 else 0 end)/  sum(case when em...  1 row(s) returned                                         0.296 sec / 0.000 sec
104  01:31:31  select * from job_data LIMIT 0, 1000                                                          8 row(s) returned                                                     0.750 sec / 0.000 sec
```

Case study 1

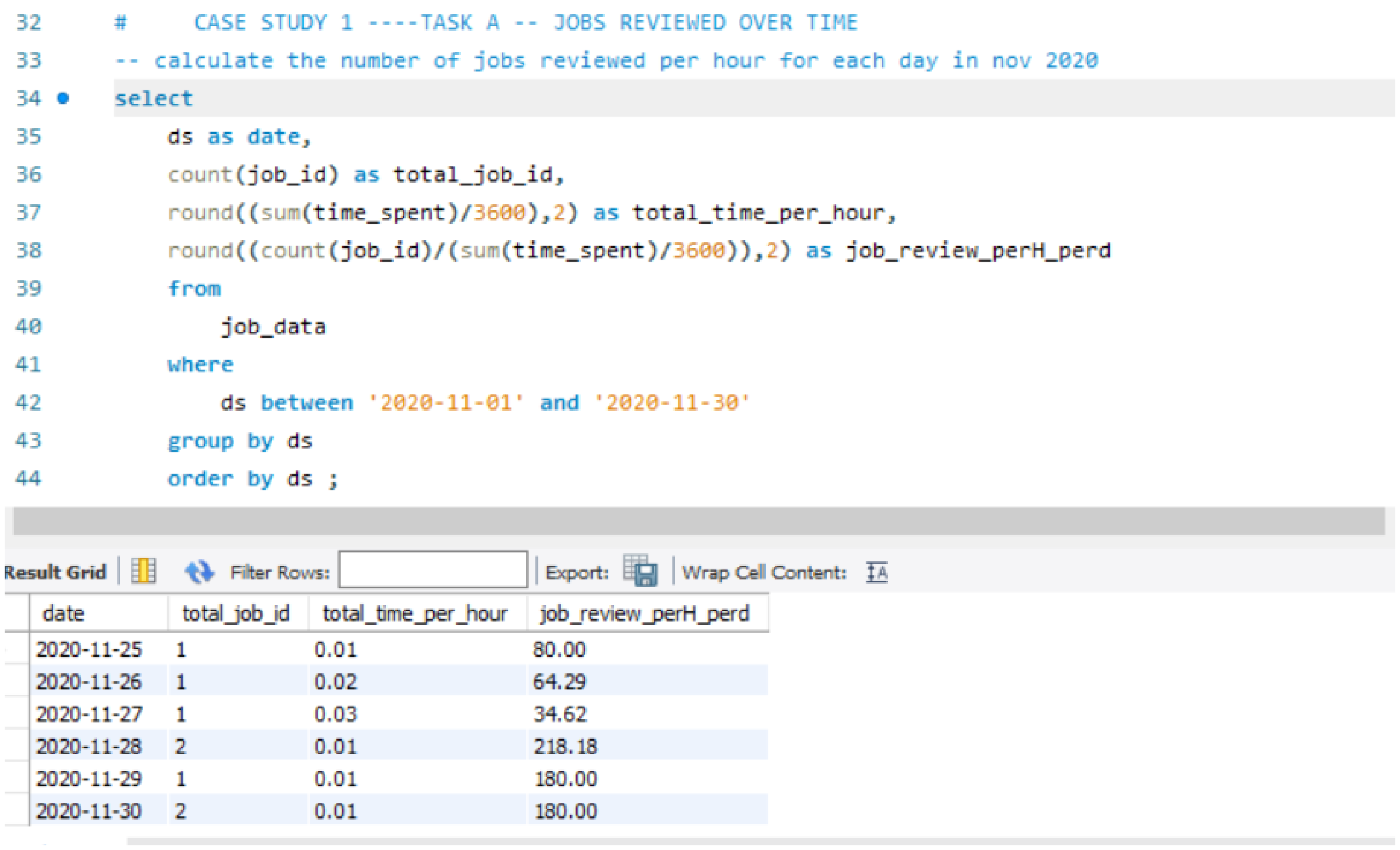
A. JOBS REVIEWED OVER TIME

Write an SQL query to calculate the number of jobs reviewed per hour for each day in November 2020.

QUERY:

```
        select
  ds as date,
  count(job_id) as total_job_id,
  round((sum(time_spent)/3600),2) as total_time_per_hour,
  round((count(job_id)/(sum(time_spent)/3600)),2) as job_review_perH_perd
  from
                job_data
        where
                ds between '2020-11-01' and '2020-11-30'
        group by ds
  order by ds ;
```

result: total number of jobs reviewed per hour for each day in November 2020 is given below in screen shot.



```
#     CASE STUDY 1 ----TASK A -- JOBS REVIEWED OVER TIME
-- calculate the number of jobs reviewed per hour for each day in nov 2020
select
    ds as date,
    count(job_id) as total_job_id,
    round((sum(time_spent)/3600),2) as total_time_per_hour,
    round((count(job_id)/(sum(time_spent)/3600)),2) as job_review_perH_perd
    from
        job_data
    where
        ds between '2020-11-01' and '2020-11-30'
    group by ds
    order by ds ;
```

date	total_job_id	total_time_per_hour	job_review_perH_perd
2020-11-25	1	0.01	80.00
2020-11-26	1	0.02	64.29
2020-11-27	1	0.03	34.62
2020-11-28	2	0.01	218.18
2020-11-29	1	0.01	180.00
2020-11-30	2	0.01	180.00

Insight: from the table we observe that 0.01 jobs reviwed per hour for each day in November 2020.

- The highest job reviewed on 28th November 218.18 per hour.

B) THROUGHTPUT ANALYSIS:

Write an SQL query to calculate the 7-day rolling average of throughput. Additionally, explain whether you prefer using the daily metric or the 7-day rolling average for throughput, and why.

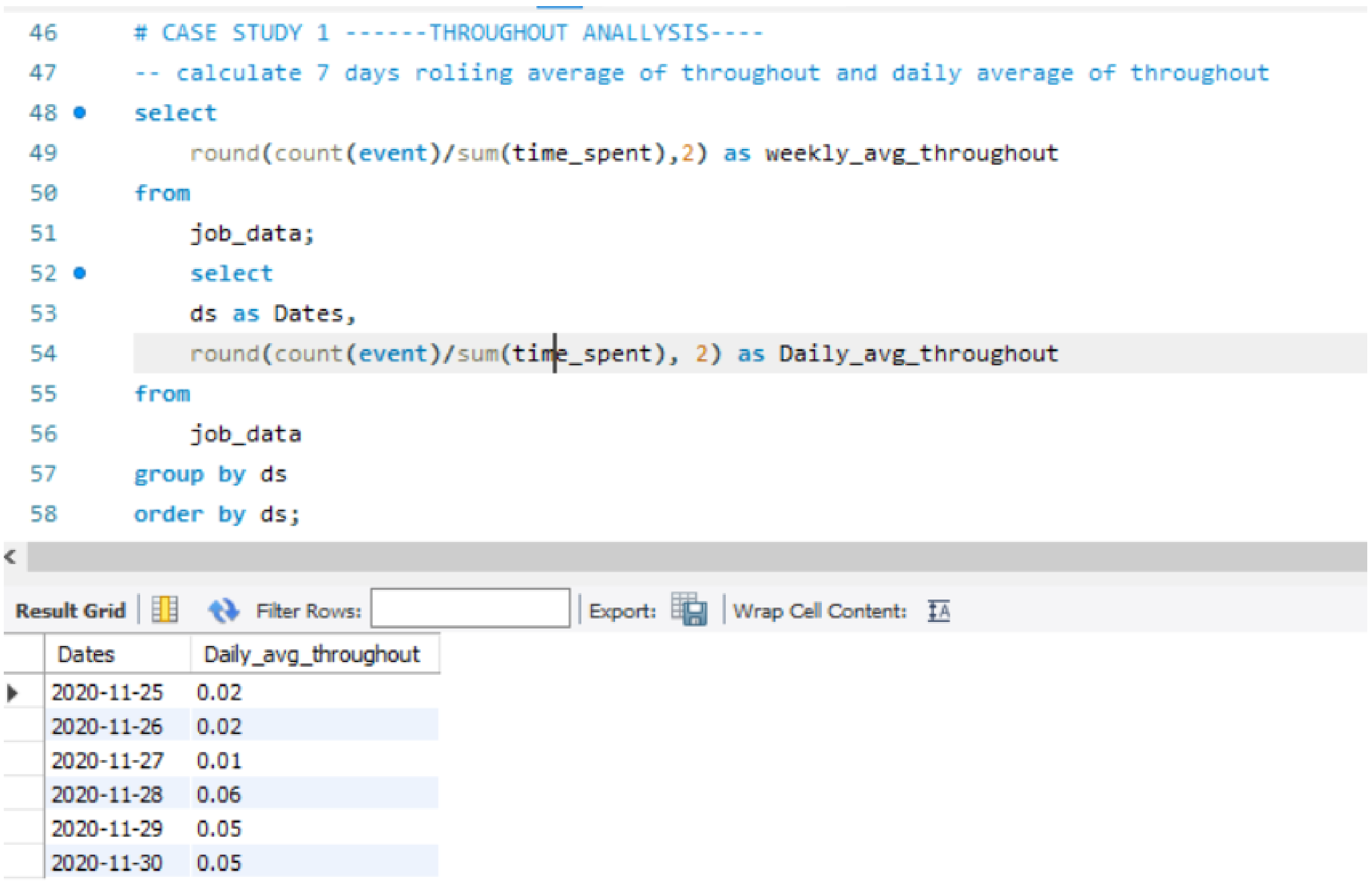
Query:

```
        select
        round(count(event)/sum(time_spent),2) as weekly_avg_throughout
from
        job_data;


select
        ds as Dates,
    round(count(event)/sum(time_spent), 2) as Daily_avg_throughout
```

```
from
        job_data
group by ds
order by ds;
```

insight for both weekly average throughout and daily out throughout:

- Result for weekly average throughout is 0.03 events per seconds
- The seven day rolling average is between 0.01 and 0.06.



```
# CASE STUDY 1 ------THROUGHOUT ANALLYSIS----
-- calculate 7 days roliing average of throughout and daily average of throughout
select
    round(count(event)/sum(time_spent),2) as weekly_avg_throughout
from
    job_data;
    select
    ds as Dates,
    round(count(event)/sum(time_spent), 2) as Daily_avg_throughout
from
    job_data
group by ds
order by ds;
```

Dates	Daily_avg_throughout
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

- Through my analysis I have come to the conclusion that weekly average throughout is more stable and comprehensive good pattern and and long term trend in the data.
- Daily average throughout is highly fluctuating and influenced by short events.

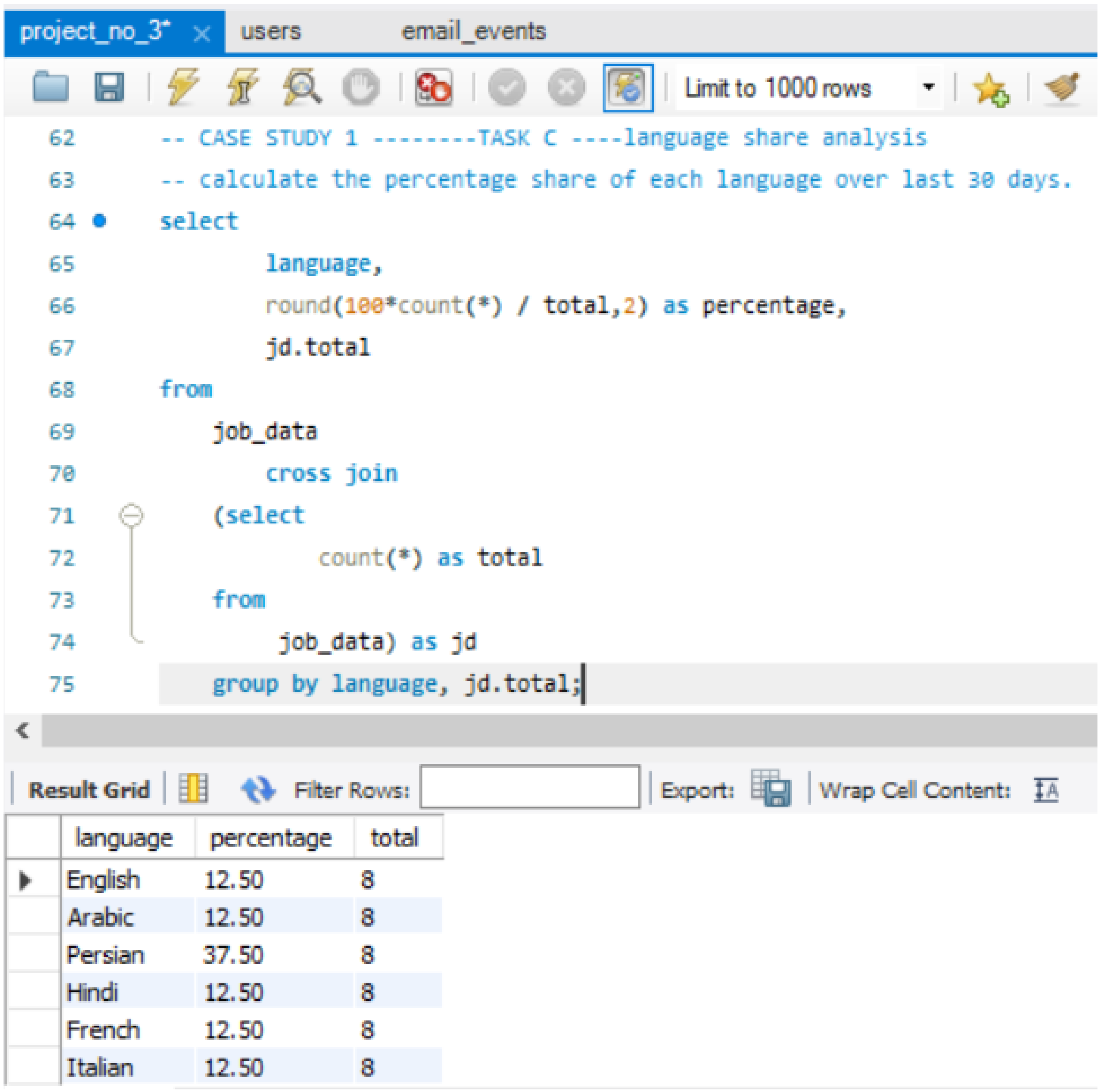
-- CASE STUDY 1 --------TASK C ----language share analysis

-- calculate the percentage share of each language over last 30 days.

Query:

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

result: persian language is the most used language with the percentage of **37.50** followed by other language having a equal share of **12.50** percent.



```
-- CASE STUDY 1 --------TASK C ----language share analysis
-- calculate the percentage share of each language over last 30 days.
select
        language,
        round(100*count(*) / total,2) as percentage,
        jd.total
from
    job_data
        cross join
    (select
            count(*) as total
    from
         job_data) as jd
    group by language, jd.total;
```

	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

TASK D . DUPLICATE ROWS DETECTION :

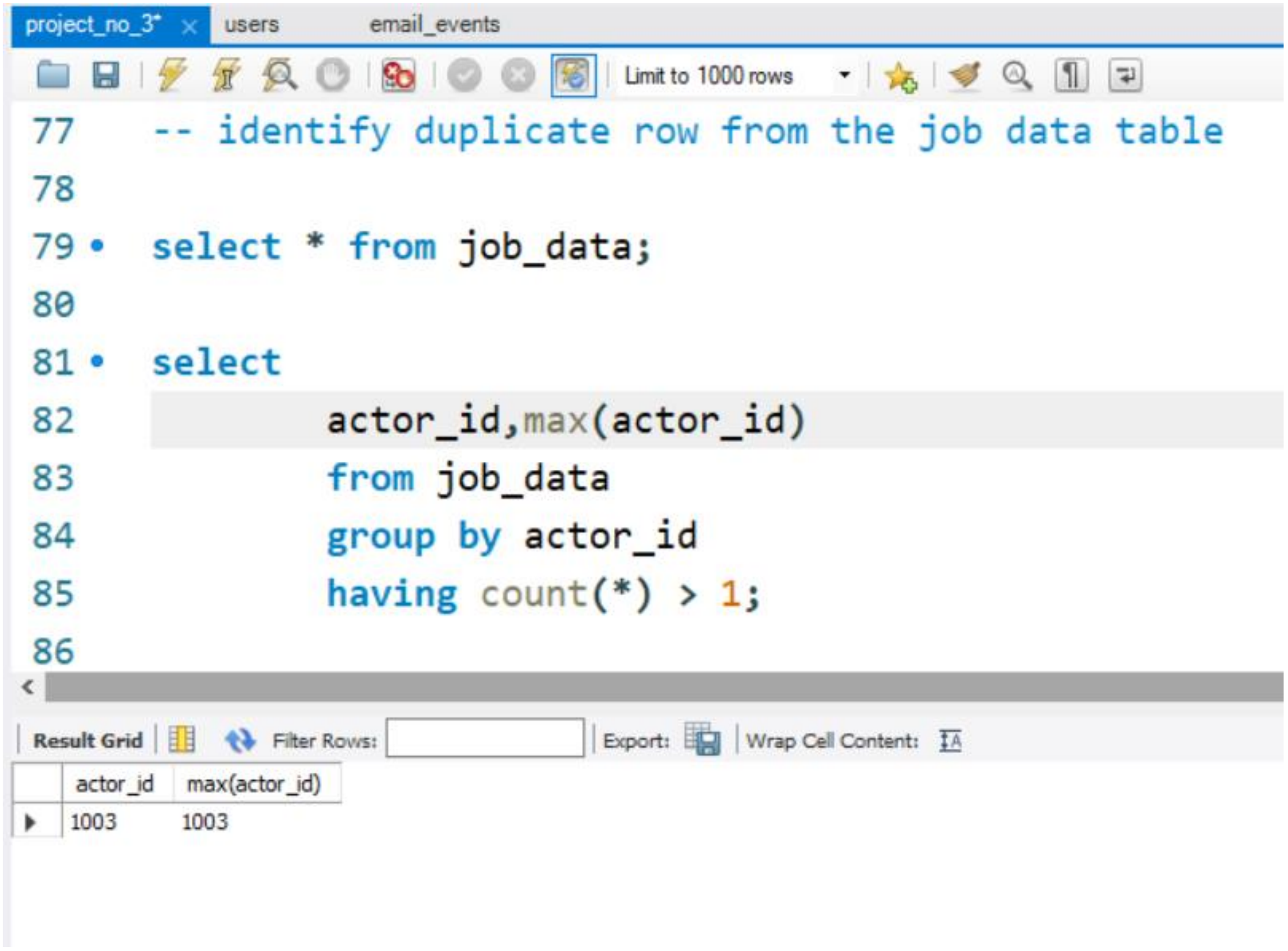
identify duplicate row from the job data table

QUERY:

select * from job_data;

```
select

        actor(id),  max(actor_id)

    from job_data

    group by actor_id
```

having count(*) > 1;



```
-- identify duplicate row from the job data table

select * from job_data;

select
        actor_id,max(actor_id)
        from job_data
        group by actor_id
        having count(*) > 1;
```

actor_id	max(actor_id)
1003	1003

Result:

- **we have total 8 rows in which 2 rows is duplicate.**
- **The actor_id 1003 is having duplicate data**.

1) CASE STUDY 2 : INVESTIGATING METRIC SPIKE

CREATING TABLE FOR INVESTIGATING METRIC SPIKE

1) CREATING FIRST TABLE USERS:-

STEP1: CREATE DATABASE

- **WHICH WE ALREADY CREATED COZ OF TASK 1.**

STEP 2: USE DATABASE

Query : use project_no_3

STEP 3: CRATE TABLE USERS

```
Query : create table users(
                User_id int,
                Created_at varchar(100),
                company_id int,
                language varhchar(100),
                activated_at varchar(100),
                state varchar(100);
```

STEP 4: AFTER CREATING TABLE WE UPOLOAD DATA INTO USERS

Query:

```
 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;
```

STEP 5: NOW WE CREATE TEMPORARY COLUMN TO CLEAN THE DATA

QUERY : alter table users add column temp_created_at datetime;

STEP 6: NOW WE STORE DATA IN THAT COLUMN BECAUSE IN DATA WE IMPORT DATE VALUE IN DIFFERENT FORMAT THAT MYSQL DOSENOT SUPPORT THAT IS WHY WE CHANGING THE DATE FORMAT IN THIS PROCESS:

QUERY:

```
update users  set temp_created_at = str_to_date(created_at, '%d-%m-%Y %H:%i');
```

STEP 7: NOW WE GONNA DELETE THE PREVIOUS COLUMN AND CHANGE THE NAME OF COLUMN THAT WE CREATE

QUERY:

```
alter table users drop column created_at;
 alter table users change column temp_created_at   created_at datetime;
```

) CREATING TABLE TWO EVENTS:-

STEP1: CREATE DATABASE

- **WHICH WE ALREADY CREATED COZ OF TASK 1.**

STEP 2: USE DATABASE

Query : use project_no_3

STEP 3: CRATE TABLE EVENTS

```
Query : create table events (
                        user_id int,
                        occurred_at varchar(80),
                        event_type varchar(50),
                        event_name varchar(100),
                        location varchar(50),
                        device varchar(50),
                        user_type int);
```

STEP 4: AFTER CREATING TABLE WE UPOLOAD DATA INTO EVENTS

Query:

```
 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;
```

STEP 5: NOW WE CREATE TEMPORARY COLUMN TO CLEAN THE DATA

QUERY : alter table events add column temp_occurred_at datetime;

STEP 6: NOW WE STORE DATA IN THAT COLUMN BECAUSE IN DATA WE IMPORT DATE VALUE IN DIFFERENT FORMAT THAT MYSQL DOSENOT SUPPORT THAT IS WHY WE CHANGING THE DATE FORMAT IN THIS PROCESS:

QUERY:

```
update events set temp_occurred_at = str_to_date(occurred_at, '%d-%m-%Y %H:%i');
```

STEP 7: NOW WE GONNA DELETE THE PREVIOUS COLUMN AND CHANGE THE NAME OF COLUMN THAT WE CREATE

QUERY:

```
alter table events drop column occurred_at;
 alter table events change column temp_occurred_at   occurred_at datetime;
```

1) CREATING TABLE 3 EMAIL EVENTS:-

STEP1: CREATE DATABASE

- **WHICH WE ALREADY CREATED COZ OF TASK 1.**

STEP 2: USE DATABASE

Query : use project_no_3

STEP 3: CRATE TABLE USERS

```
Query : create table email_events(
                user_id int,
                occurred_at varchar(100),
                action varchar(100),
                user_type int );
```

STEP 4: AFTER CREATING TABLE WE UPOLOAD DATA INTO email events

Query:

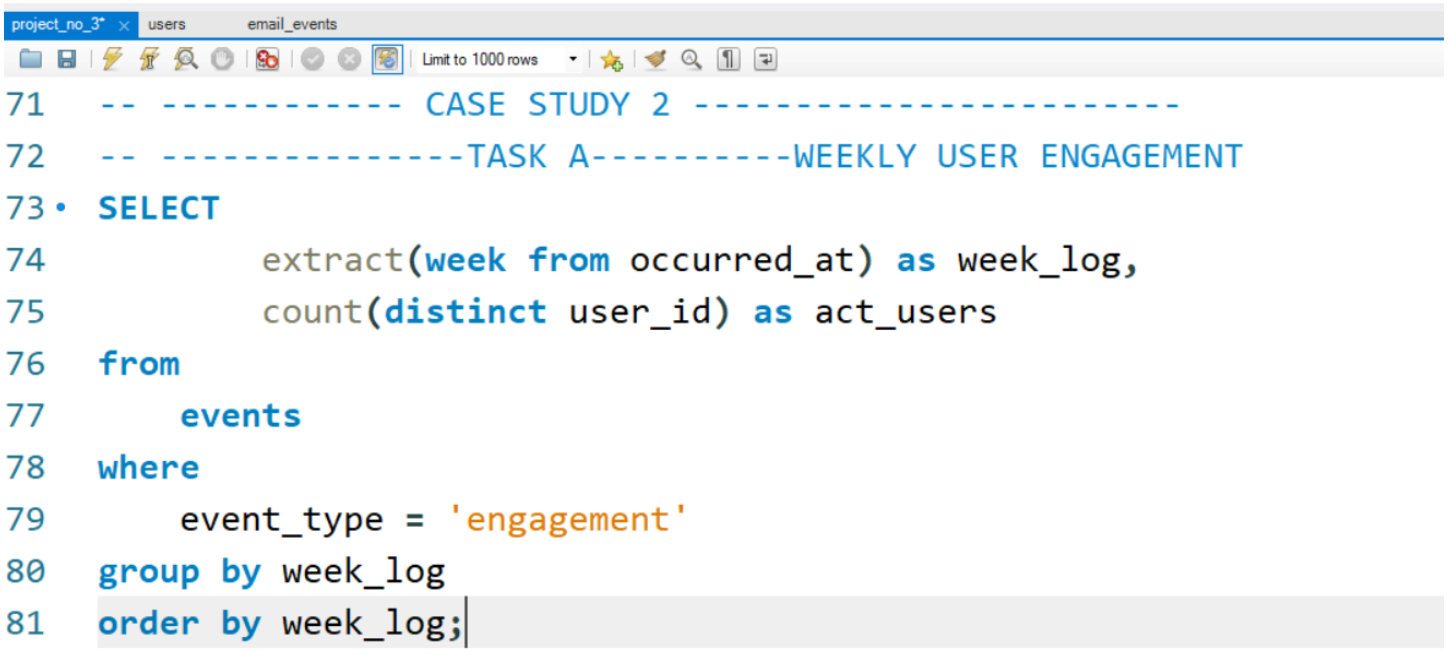
```
 load data infile "C:/ProgramData/MySQL/MySQL Server 8.0/Uploads/email_events.csv"
into table email_events
fields terminated by ','
enclosed by '"'
lines terminated by '\n'
ignore 1 rows;
```

CASE STUDY 2 :

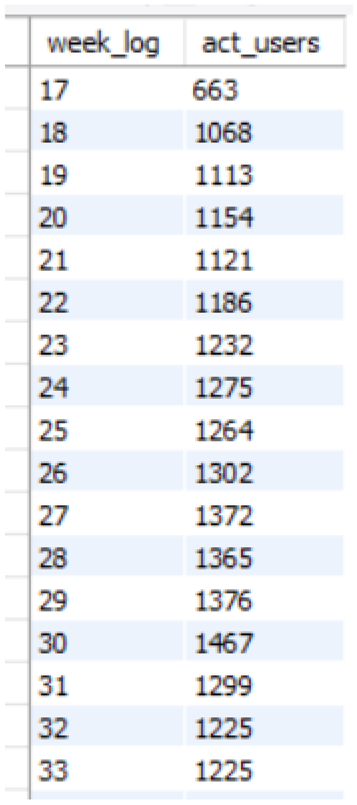
A. WEEKLY USER ENGAGEMENT

QUERY:

```
SELECT
                        extract(week from occurred_at) as week_log,
        count(distinct user_id) as act_users
from
            events
where
            event_type = 'engagement'
group by week_log
order by week_log;
```



```
-- ------------- CASE STUDY 2 ------------------------
-- ----------------TASK A----------WEEKLY USER ENGAGEMENT
SELECT
        extract(week from occurred_at) as week_log,
        count(distinct user_id) as act_users
from
    events
where
    event_type = 'engagement'
group by week_log
order by week_log;
```



week_log	act_users
17	663
18	1068
19	1113
20	1154
21	1121
22	1186
23	1232
24	1275
25	1264
26	1302
27	1372
28	1365
29	1376
30	1467
31	1299
32	1225
33	1225

INSIGHT:

- The highest user week 30th week 1467 users.
- The minimum user week 35th week 104 users.

TASK B) USER GROWTH ANALYSIS

Write an SQL query to calculate the user growth for the product.

QUERY:

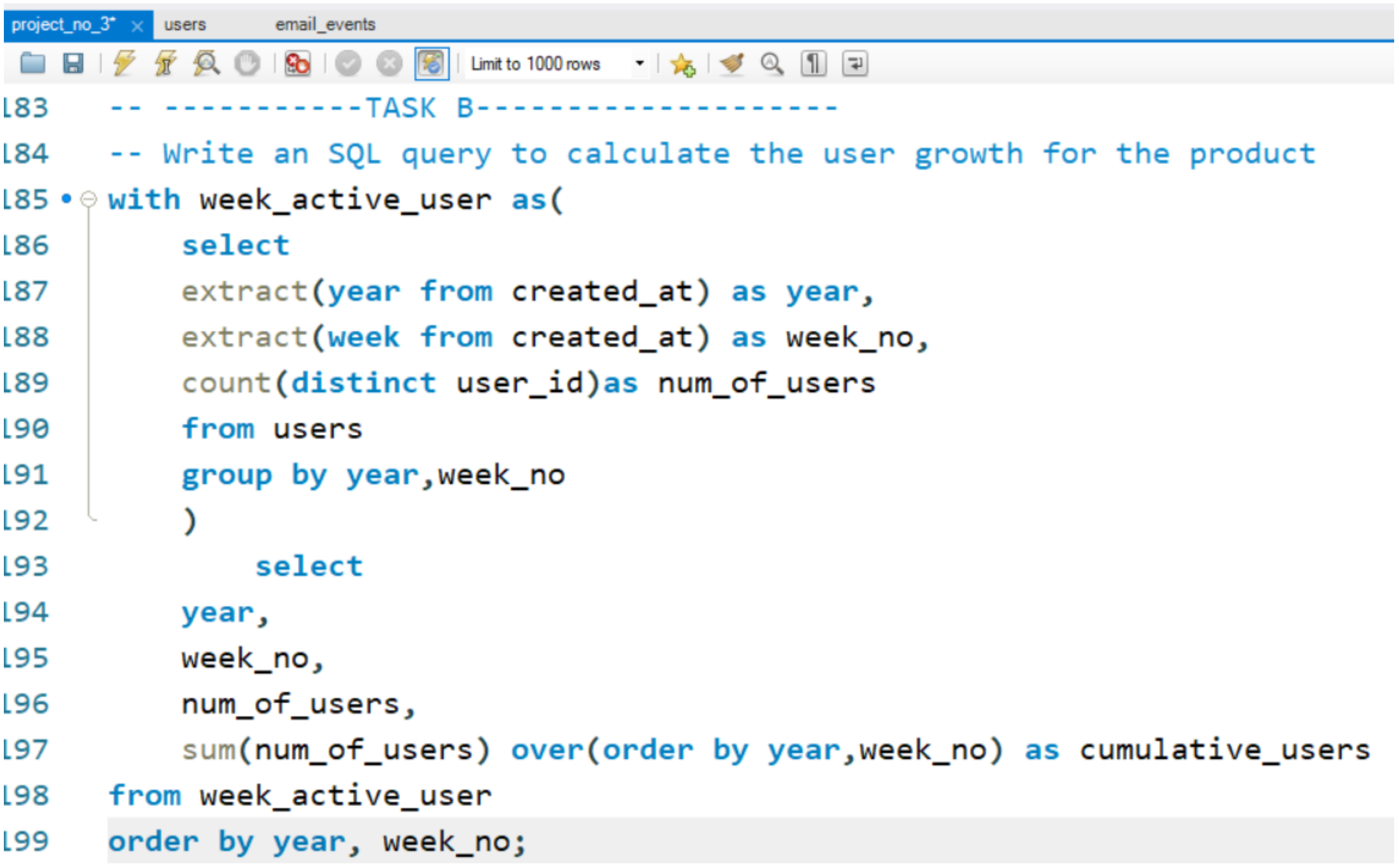
```
        with week_active_user as(
        select
   extract(year from created_at) as year,
   extract(week from created_at) as week_no,
   count(distinct user_id)as num_of_users
   from users
   group by year,week_no
   )
      select
   year,
   week_no,
```

num_of_users,

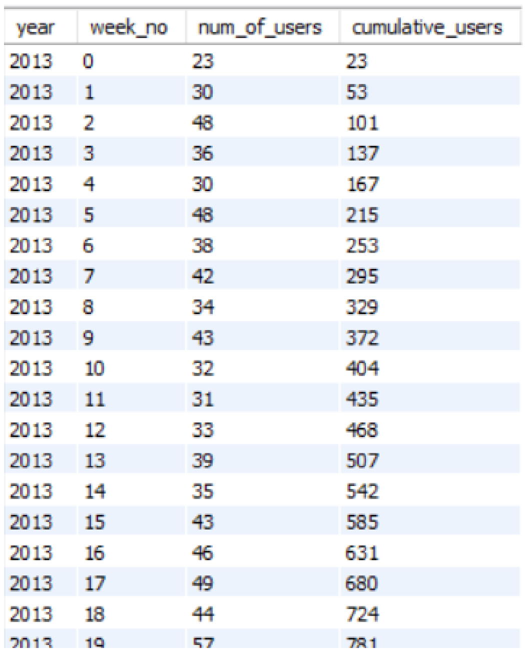
sum(num_of_users) over(order by year,week_no) as cumulative_users

from week_active_user

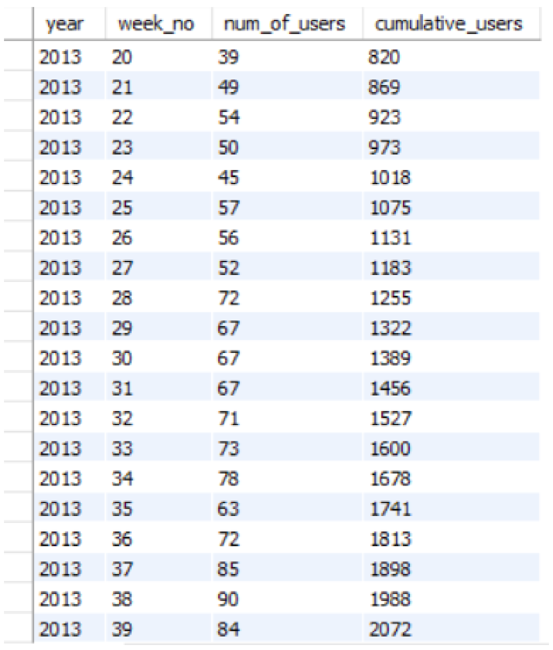
order by year, week_no;



```
-- -----------TASK B--------------------
-- Write an SQL query to calculate the user growth for the product
with week_active_user as(
    select
    extract(year from created_at) as year,
    extract(week from created_at) as week_no,
    count(distinct user_id)as num_of_users
    from users
    group by year,week_no
    )
        select
    year,
    week_no,
    num_of_users,
    sum(num_of_users) over(order by year,week_no) as cumulative_users
from week_active_user
order by year, week_no;
```



year	week_no	num_of_users	cumulative_users
2013	0	23	23
2013	1	30	53
2013	2	48	101
2013	3	36	137
2013	4	30	167
2013	5	48	215
2013	6	38	253
2013	7	42	295
2013	8	34	329
2013	9	43	372
2013	10	32	404
2013	11	31	435
2013	12	33	468
2013	13	39	507
2013	14	35	542
2013	15	43	585
2013	16	46	631
2013	17	49	680
2013	18	44	724
2013	19	57	781



year	week_no	num_of_users	cumulative_users
2013	20	39	820
2013	21	49	869
2013	22	54	923
2013	23	50	973
2013	24	45	1018
2013	25	57	1075
2013	26	56	1131
2013	27	52	1183
2013	28	72	1255
2013	29	67	1322
2013	30	67	1389
2013	31	67	1456
2013	32	71	1527
2013	33	73	1600
2013	34	78	1678
2013	35	63	1741
2013	36	72	1813
2013	37	85	1898
2013	38	90	1988
2013	39	84	2072

RESULT:

The highest number of new users is in the 33rd week of 2014 with 261 new users,

Reaching a total of 9014 users.

- Lowest number of users is in the 35th week of 2014 with 18 new users,

  Reaching a total of 9381 users.

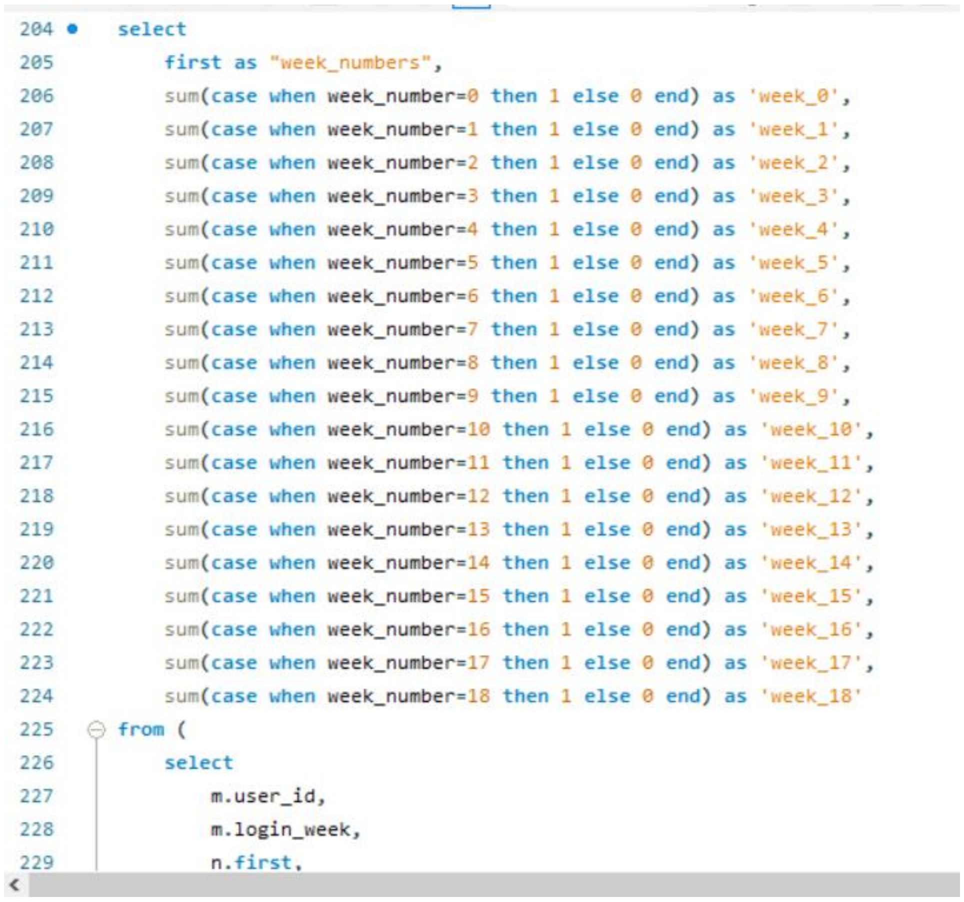
- According to the data company is showing the positive growth and its increase by the end of 35 week of 2014.

TASK C) WEEKLY RETENTION ANALYSIS

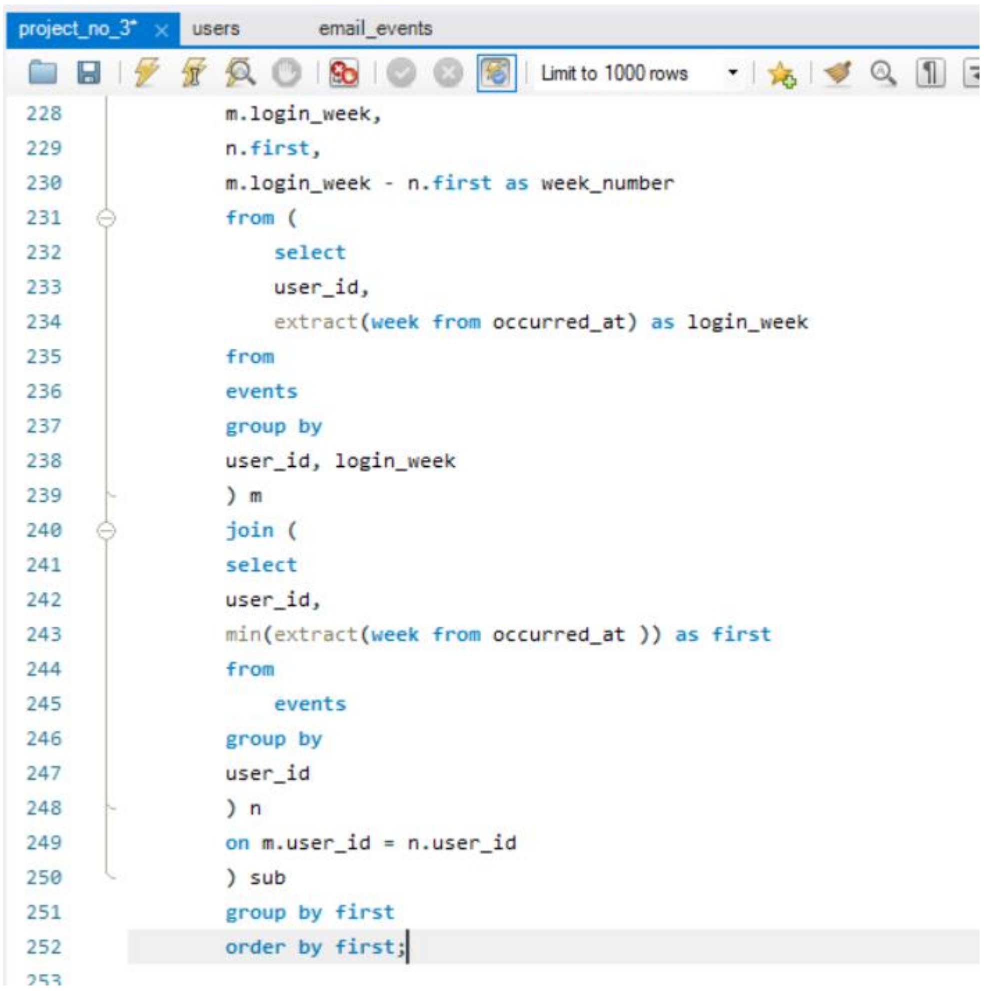
Write an SQL query to calculate the weekly retention of users based on their sign-up cohort.

QUERY:

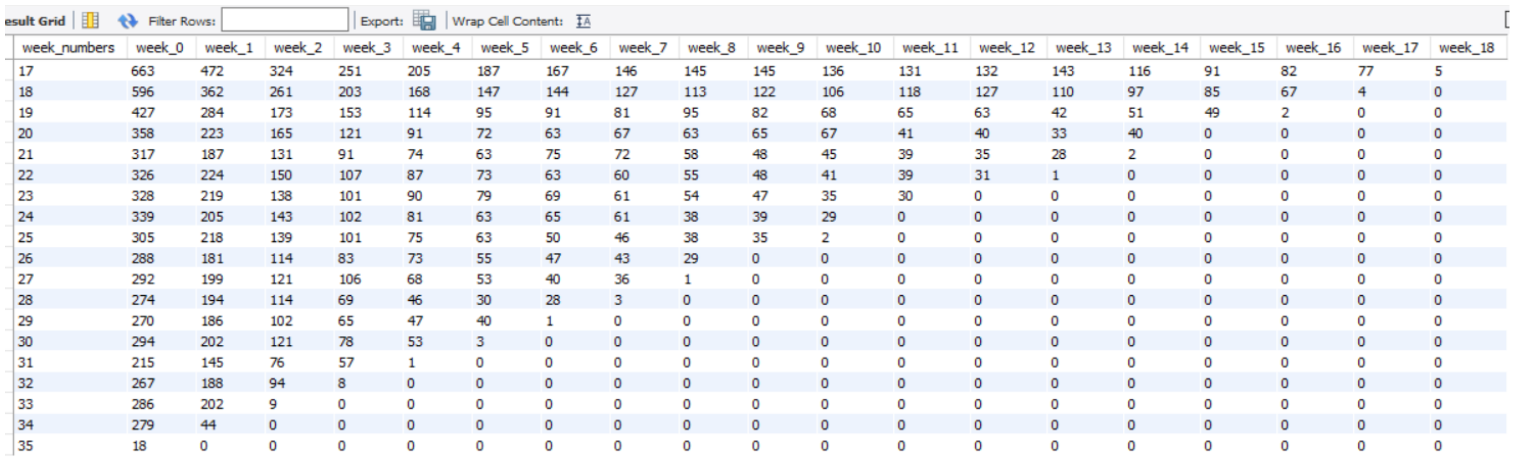
```
        select
        first as "week_numbers",
  sum(case when week_number=0 then 1 else 0 end) as 'week_0',
  sum(case when week_number=1 then 1 else 0 end) as 'week_1',
  sum(case when week_number=2 then 1 else 0 end) as 'week_2',
  sum(case when week_number=3 then 1 else 0 end) as 'week_3',
  sum(case when week_number=4 then 1 else 0 end) as 'week_4',
  sum(case when week_number=5 then 1 else 0 end) as 'week_5',
  sum(case when week_number=6 then 1 else 0 end) as 'week_6',
  sum(case when week_number=7 then 1 else 0 end) as 'week_7',
  sum(case when week_number=8 then 1 else 0 end) as 'week_8',
  sum(case when week_number=9 then 1 else 0 end) as 'week_9',
  sum(case when week_number=10 then 1 else 0 end) as 'week_10',
  sum(case when week_number=11 then 1 else 0 end) as 'week_11',
  sum(case when week_number=12 then 1 else 0 end) as 'week_12',
  sum(case when week_number=13 then 1 else 0 end) as 'week_13',
  sum(case when week_number=14 then 1 else 0 end) as 'week_14',
  sum(case when week_number=15 then 1 else 0 end) as 'week_15',
  sum(case when week_number=16 then 1 else 0 end) as 'week_16',
    sum(case when week_number=17 then 1 else 0 end) as 'week_17',
    sum(case when week_number=18 then 1 else 0 end) as 'week_18'
from (
    select
                    m.user_id,
        m.login_week,
        n.first,
        m.login_week - n.first as week_number
        from (
                                select
            user_id,
            extract(week from occurred_at) as login_week
                    from
        events
        group by
        user_id, login_week
        ) m
        join (
        select
        user_id,
        min(extract(week from occurred_at )) as first
        from
                                events
                    group by
        user_id
        ) n
        on m.user_id = n.user_id
        ) sub
        group by first
        order by first;
```



```
select
    first as "week_numbers",
    sum(case when week_number=0 then 1 else 0 end) as 'week_0',
    sum(case when week_number=1 then 1 else 0 end) as 'week_1',
    sum(case when week_number=2 then 1 else 0 end) as 'week_2',
    sum(case when week_number=3 then 1 else 0 end) as 'week_3',
    sum(case when week_number=4 then 1 else 0 end) as 'week_4',
    sum(case when week_number=5 then 1 else 0 end) as 'week_5',
    sum(case when week_number=6 then 1 else 0 end) as 'week_6',
    sum(case when week_number=7 then 1 else 0 end) as 'week_7',
    sum(case when week_number=8 then 1 else 0 end) as 'week_8',
    sum(case when week_number=9 then 1 else 0 end) as 'week_9',
    sum(case when week_number=10 then 1 else 0 end) as 'week_10',
    sum(case when week_number=11 then 1 else 0 end) as 'week_11',
    sum(case when week_number=12 then 1 else 0 end) as 'week_12',
    sum(case when week_number=13 then 1 else 0 end) as 'week_13',
    sum(case when week_number=14 then 1 else 0 end) as 'week_14',
    sum(case when week_number=15 then 1 else 0 end) as 'week_15',
    sum(case when week_number=16 then 1 else 0 end) as 'week_16',
    sum(case when week_number=17 then 1 else 0 end) as 'week_17',
    sum(case when week_number=18 then 1 else 0 end) as 'week_18'
from (
    select
        m.user_id,
        m.login_week,
        n.first,
```



```
        m.login_week,
        n.first,
        m.login_week - n.first as week_number
        from (
            select
            user_id,
            extract(week from occurred_at) as login_week
        from
        events
        group by
        user_id, login_week
        ) m
        join (
        select
        user_id,
        min(extract(week from occurred_at )) as first
        from
            events
        group by
        user_id
        ) n
        on m.user_id = n.user_id
        ) sub
        group by first
        order by first;
```


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

week_numbers	week_0	week_1	week_2	week_3	week_4	week_5	week_6	week_7	week_8	week_9	week_10	week_11	week_12	week_13	week_14	week_15	week_16	week_17	week_18
17	663	472	324	251	205	187	167	146	145	145	136	131	132	143	116	91	82	77	5
18	596	362	261	203	168	147	144	127	113	122	106	118	127	110	97	85	67	4	0
19	427	284	173	153	114	95	91	81	95	82	68	65	63	42	51	49	2	0	0
20	358	223	165	121	91	72	63	67	63	65	67	41	40	33	40	0	0	0	0
21	317	187	131	91	74	63	75	72	58	48	45	39	35	28	2	0	0	0	0
22	326	224	150	107	87	73	63	60	55	48	41	39	31	1	0	0	0	0	0
23	328	219	138	101	90	79	69	61	54	47	35	30	0	0	0	0	0	0	0
24	339	205	143	102	81	63	65	61	38	39	29	0	0	0	0	0	0	0	0
25	305	218	139	101	75	63	50	46	38	35	2	0	0	0	0	0	0	0	0
26	288	181	114	83	73	55	47	43	29	0	0	0	0	0	0	0	0	0	0
27	292	199	121	106	68	53	40	36	1	0	0	0	0	0	0	0	0	0	0
28	274	194	114	69	46	30	28	3	0	0	0	0	0	0	0	0	0	0	0
29	270	186	102	65	47	40	1	0	0	0	0	0	0	0	0	0	0	0	0
30	294	202	121	78	53	3	0	0	0	0	0	0	0	0	0	0	0	0	0
31	215	145	76	57	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0
32	267	188	94	8	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
33	286	202	9	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
34	279	44	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
35	18	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

RESULT:

The result of weekly retention users based on their sign up cohart

The lower user join in 35th week is 18 users

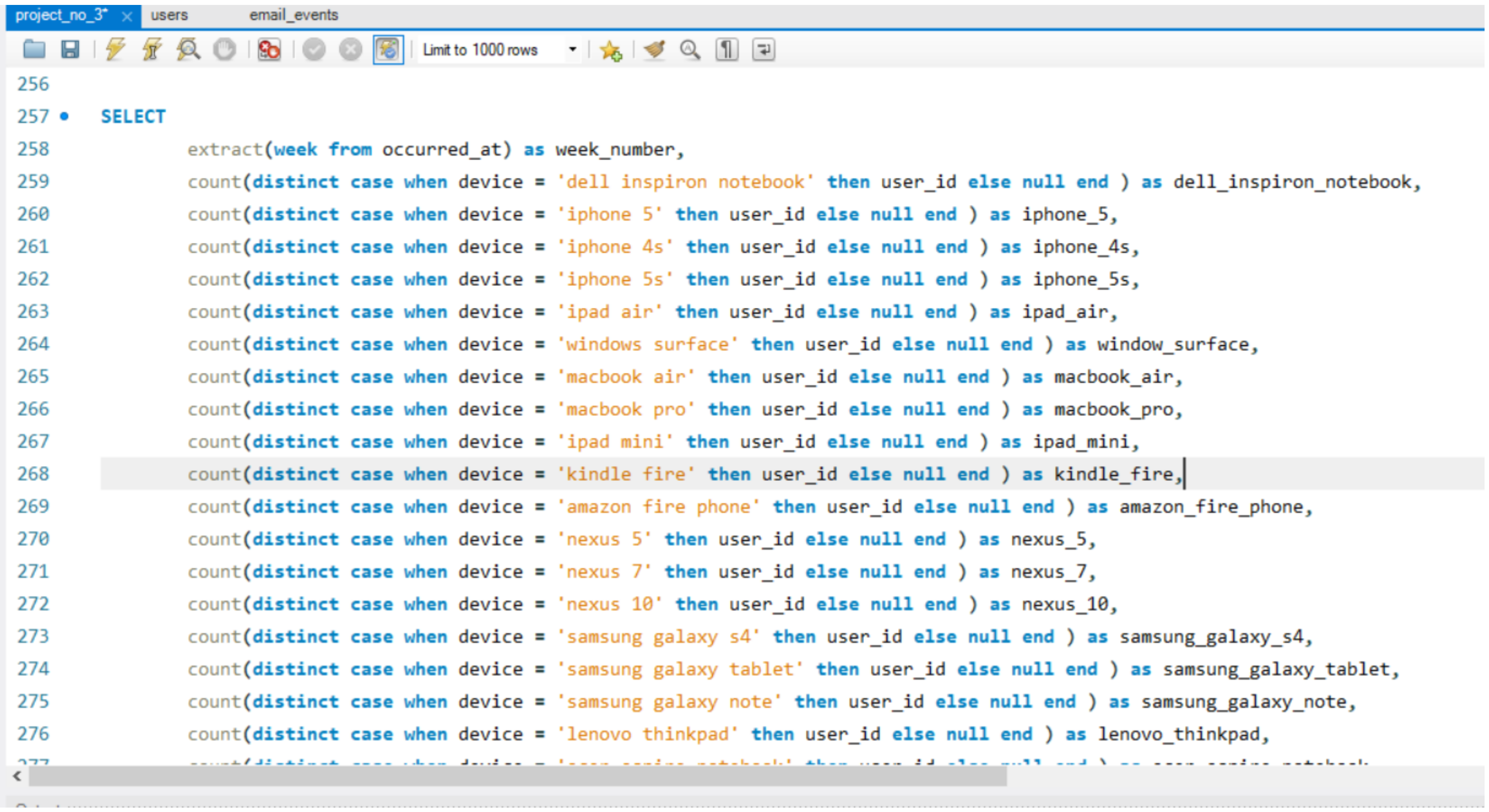
The 17th week is largest users joined week .which is retained by 18 week with 5 users.

TASK D) WEEKLY ENGAGEMENT PER DEVICE:

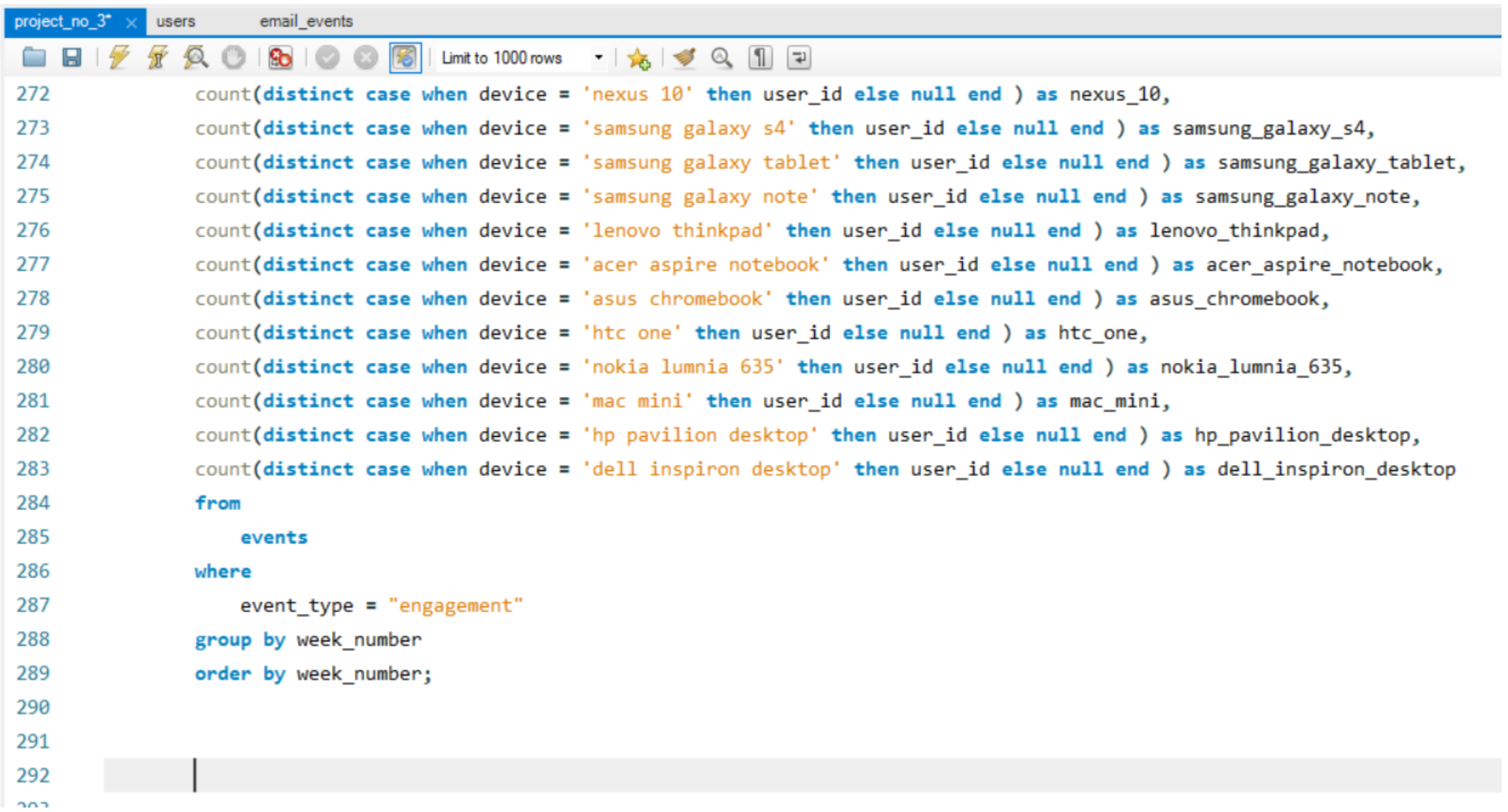
Write an SQL query to calculate the weekly engagement per device.

QUERY:

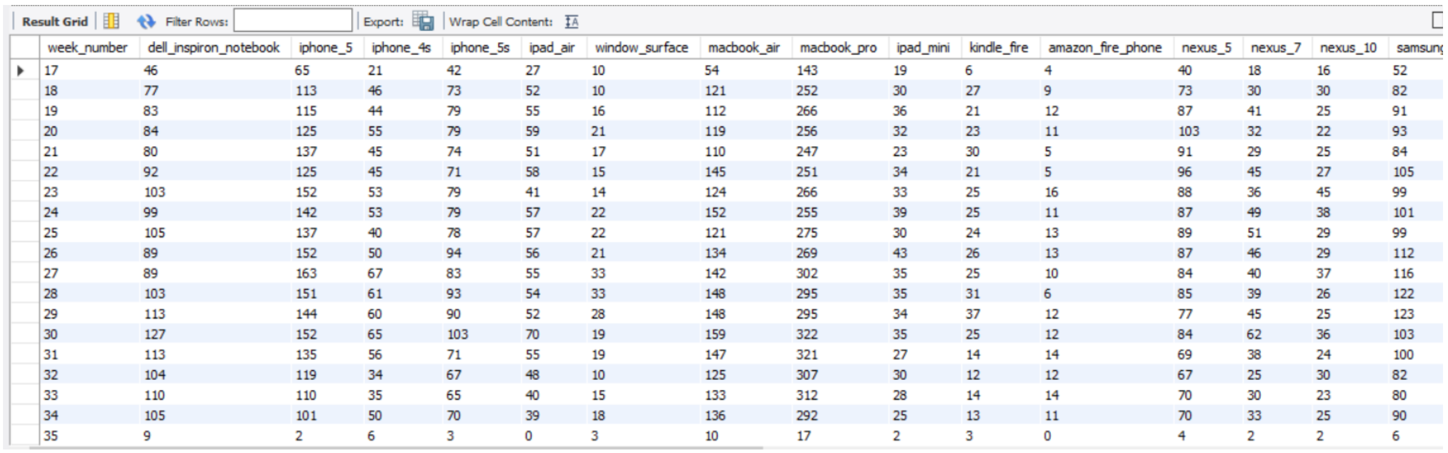
```
        SELECT
                extract(week from occurred_at) as week_number,
        count(distinct case when device = 'dell inspiron notebook' then user_id else null end ) as
dell_inspiron_notebook,
      count(distinct case when device = 'iphone 5' then user_id else null end ) as iphone_5,
      count(distinct case when device = 'iphone 4s' then user_id else null end ) as iphone_4s,
      count(distinct case when device = 'iphone 5s' then user_id else null end ) as iphone_5s,
      count(distinct case when device = 'ipad air' then user_id else null end ) as ipad_air,
      count(distinct case when device = 'windows surface' then user_id else null end ) as
window_surface,
      count(distinct case when device = 'macbook air' then user_id else null end ) as macbook_air,
      count(distinct case when device = 'macbook pro' then user_id else null end ) as macbook_pro,
      count(distinct case when device = 'ipad mini' then user_id else null end ) as ipad_mini,
      count(distinct case when device = 'kindle fire' then user_id else null end ) as kindle_fire,
      count(distinct case when device = 'amazon fire phone' then user_id else null end ) as
amazon_fire_phone,
      count(distinct case when device = 'nexus 5' then user_id else null end ) as nexus_5,
      count(distinct case when device = 'nexus 7' then user_id else null end ) as nexus_7,
      count(distinct case when device = 'nexus 10' then user_id else null end ) as nexus_10,
      count(distinct case when device = 'samsung galaxy s4' then user_id else null end ) as
samsung_galaxy_s4,
      count(distinct case when device = 'samsung galaxy tablet' then user_id else null end ) as
samsung_galaxy_tablet,
      count(distinct case when device = 'samsung galaxy note' then user_id else null end ) as
samsung_galaxy_note,
      count(distinct case when device = 'lenovo thinkpad' then user_id else null end ) as
lenovo_thinkpad,
      count(distinct case when device = 'acer aspire notebook' then user_id else null end ) as
acer_aspire_notebook,
      count(distinct case when device = 'asus chromebook' then user_id else null end ) as
asus_chromebook,
      count(distinct case when device = 'htc one' then user_id else null end ) as htc_one,
      count(distinct case when device = 'nokia lumnia 635' then user_id else null end ) as
nokia_lumnia_635,
      count(distinct case when device = 'mac mini' then user_id else null end ) as mac_mini,
      count(distinct case when device = 'hp pavilion desktop' then user_id else null end ) as
hp_pavilion_desktop,
      count(distinct case when device = 'dell inspiron desktop' then user_id else null end ) as
dell_inspiron_desktop
      from
                      events
                where
                      event_type = "engagement"
                group by week_number
      order by week_number;
```



```sql
SELECT
        extract(week from occurred_at) as week_number,
        count(distinct case when device = 'dell inspiron notebook' then user_id else null end ) as dell_inspiron_notebook,
        count(distinct case when device = 'iphone 5' then user_id else null end ) as iphone_5,
        count(distinct case when device = 'iphone 4s' then user_id else null end ) as iphone_4s,
        count(distinct case when device = 'iphone 5s' then user_id else null end ) as iphone_5s,
        count(distinct case when device = 'ipad air' then user_id else null end ) as ipad_air,
        count(distinct case when device = 'windows surface' then user_id else null end ) as window_surface,
        count(distinct case when device = 'macbook air' then user_id else null end ) as macbook_air,
        count(distinct case when device = 'macbook pro' then user_id else null end ) as macbook_pro,
        count(distinct case when device = 'ipad mini' then user_id else null end ) as ipad_mini,
        count(distinct case when device = 'kindle fire' then user_id else null end ) as kindle_fire,
        count(distinct case when device = 'amazon fire phone' then user_id else null end ) as amazon_fire_phone,
        count(distinct case when device = 'nexus 5' then user_id else null end ) as nexus_5,
        count(distinct case when device = 'nexus 7' then user_id else null end ) as nexus_7,
        count(distinct case when device = 'nexus 10' then user_id else null end ) as nexus_10,
        count(distinct case when device = 'samsung galaxy s4' then user_id else null end ) as samsung_galaxy_s4,
        count(distinct case when device = 'samsung galaxy tablet' then user_id else null end ) as samsung_galaxy_tablet,
        count(distinct case when device = 'samsung galaxy note' then user_id else null end ) as samsung_galaxy_note,
        count(distinct case when device = 'lenovo thinkpad' then user_id else null end ) as lenovo_thinkpad,
```

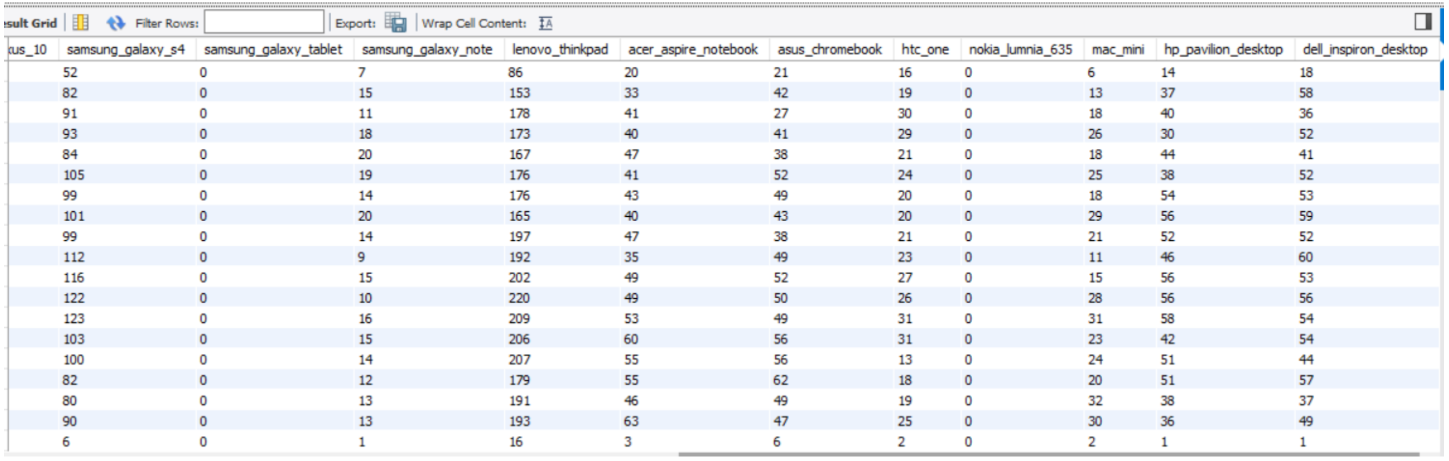


```sql
        count(distinct case when device = 'nexus 10' then user_id else null end ) as nexus_10,
        count(distinct case when device = 'samsung galaxy s4' then user_id else null end ) as samsung_galaxy_s4,
        count(distinct case when device = 'samsung galaxy tablet' then user_id else null end ) as samsung_galaxy_tablet,
        count(distinct case when device = 'samsung galaxy note' then user_id else null end ) as samsung_galaxy_note,
        count(distinct case when device = 'lenovo thinkpad' then user_id else null end ) as lenovo_thinkpad,
        count(distinct case when device = 'acer aspire notebook' then user_id else null end ) as acer_aspire_notebook,
        count(distinct case when device = 'asus chromebook' then user_id else null end ) as asus_chromebook,
        count(distinct case when device = 'htc one' then user_id else null end ) as htc_one,
        count(distinct case when device = 'nokia lumnia 635' then user_id else null end ) as nokia_lumnia_635,
        count(distinct case when device = 'mac mini' then user_id else null end ) as mac_mini,
        count(distinct case when device = 'hp pavilion desktop' then user_id else null end ) as hp_pavilion_desktop,
        count(distinct case when device = 'dell inspiron desktop' then user_id else null end ) as dell_inspiron_desktop
        from
            events
        where
            event_type = "engagement"
        group by week_number
        order by week_number;
```



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

week_number	dell_inspiron_notebook	iphone_5	iphone_4s	iphone_5s	ipad_air	window_surface	macbook_air	macbook_pro	ipad_mini	kindle_fire	amazon_fire_phone	nexus_5	nexus_7	nexus_10	samsun
17	46	65	21	42	27	10	54	143	19	6	4	40	18	16	52
18	77	113	46	73	52	10	121	252	30	27	9	73	30	30	82
19	83	115	44	79	55	16	112	266	36	21	12	87	41	25	91
20	84	125	55	79	59	21	119	256	32	23	11	103	32	22	93
21	80	137	45	74	51	17	110	247	23	30	5	91	29	25	84
22	92	125	45	71	58	15	145	251	34	21	5	96	45	27	105
23	103	152	53	79	41	14	124	266	33	25	16	88	36	45	99
24	99	142	53	79	57	22	152	255	39	25	11	87	49	38	101
25	105	137	40	78	57	22	121	275	30	24	13	89	51	29	99
26	89	152	50	94	56	21	134	269	43	26	13	87	46	29	112
27	89	163	67	83	55	33	142	302	35	25	10	84	40	37	116
28	103	151	61	93	54	33	148	295	35	31	6	85	39	26	122
29	113	144	60	90	52	28	148	295	34	37	12	77	45	25	123
30	127	152	65	103	70	19	159	322	35	25	12	84	62	36	103
31	113	135	56	71	55	19	147	321	27	14	14	69	38	24	100
32	104	119	34	67	48	10	125	307	30	12	12	67	25	30	82
33	110	110	35	65	40	15	133	312	28	14	14	70	30	23	80
34	105	101	50	70	39	18	136	292	25	13	11	70	33	25	90
35	9	2	6	3	0	3	10	17	2	3	0	4	2	2	6



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

us_10	samsung_galaxy_s4	samsung_galaxy_tablet	samsung_galaxy_note	lenovo_thinkpad	acer_aspire_notebook	asus_chromebook	htc_one	nokia_lumnia_635	mac_mini	hp_pavilion_desktop	dell_inspiron_desktop
	52	0	7	86	20	21	16	0	6	14	18
	82	0	15	153	33	42	19	0	13	37	58
	91	0	11	178	41	27	30	0	18	40	36
	93	0	18	173	40	41	29	0	26	30	52
	84	0	20	167	47	38	21	0	18	44	41
	105	0	19	176	41	52	24	0	25	38	52
	99	0	14	176	43	49	20	0	18	54	53
	101	0	20	165	40	43	20	0	29	56	59
	99	0	14	197	47	38	21	0	21	52	52
	112	0	9	192	35	49	23	0	11	46	60
	116	0	15	202	49	52	27	0	15	56	53
	122	0	10	220	49	50	26	0	28	56	56
	123	0	16	209	53	49	31	0	31	58	54
	103	0	15	206	60	56	31	0	23	42	54
	100	0	14	207	55	56	13	0	24	51	44
	82	0	12	179	55	62	18	0	20	51	57
	80	0	13	191	46	49	19	0	32	38	37
	90	0	13	193	63	47	25	0	30	36	49
	6	0	1	16	3	6	2	0	2	1	1

RESULT:

- From the table it is clear that most people uses macbook pro (322 users on 30th week) , followed By Lenovo thinkpad (220 users, 28th week) and iphone 5 (163 users on 27th week)

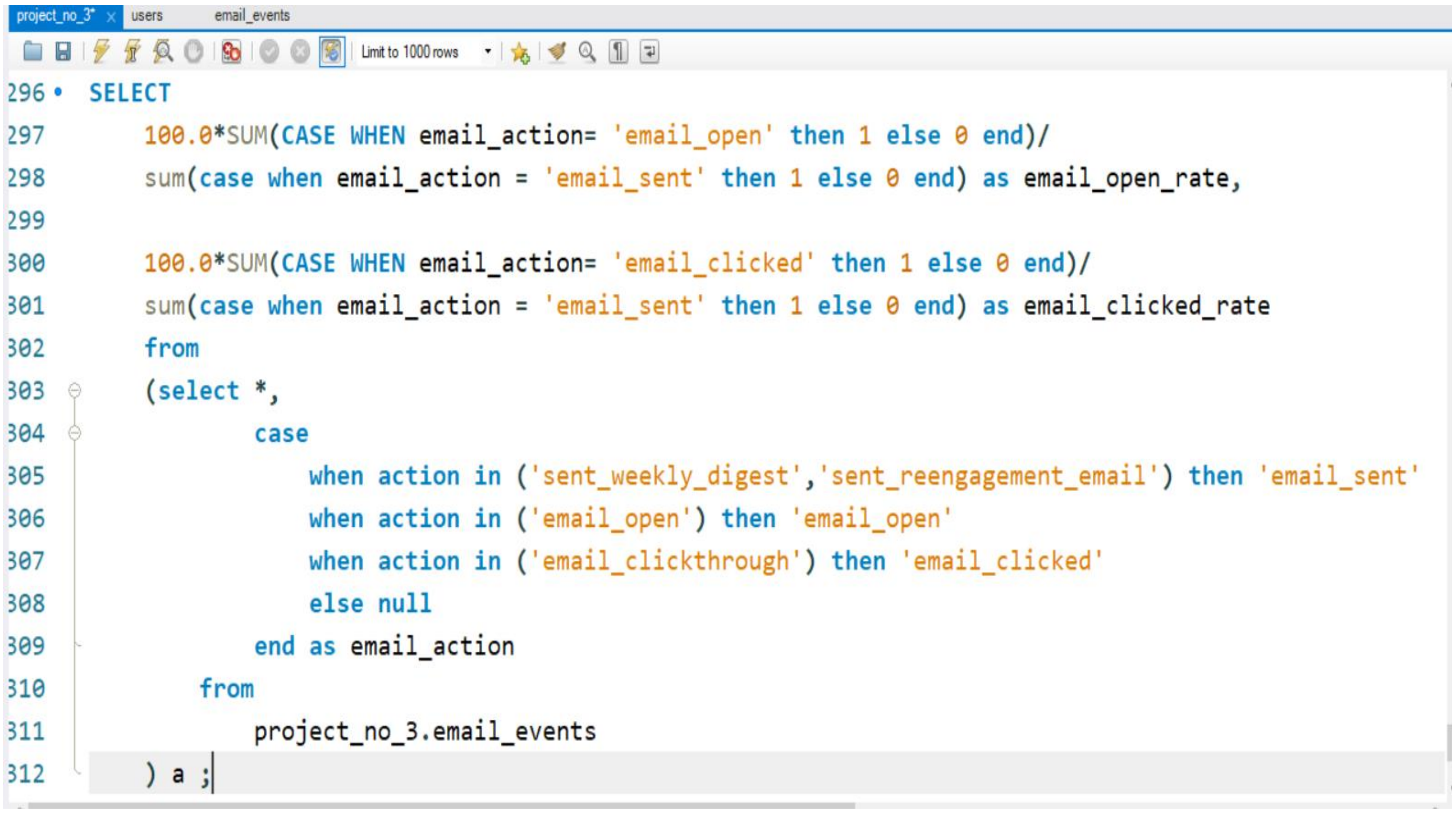
TASK E) EMAIL ENGAGEMENT ANALYSIS:

Write an SQL query to calculate the email engagement metrics.

QUERY:

```
        SELECT
        100.0*SUM(CASE WHEN email_action= 'email_open' then 1 else 0 end)/
   sum(case when email_action = 'email_sent' then 1 else 0 end) as email_open_rate,


   100.0*SUM(CASE WHEN email_action= 'email_clicked' then 1 else 0 end)/
   sum(case when email_action = 'email_sent' then 1 else 0 end) as email_clicked_rate
   from
   (select *,
                         case
```

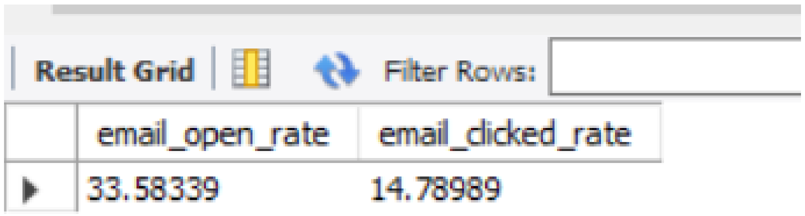
```
                                        when action in ('sent_weekly_digest','sent_reengagement_email') then
'email_sent'
            when action in ('email_open') then 'email_open'
            when action in ('email_clickthrough') then 'email_clicked'
            else null
                        end as email_action
                from
                        project_no_3.email_events
        ) a ;
```



project_no_3* users email_events

Limit to 1000 rows

```
SELECT
    100.0*SUM(CASE WHEN email_action= 'email_open' then 1 else 0 end)/
    sum(case when email_action = 'email_sent' then 1 else 0 end) as email_open_rate,

    100.0*SUM(CASE WHEN email_action= 'email_clicked' then 1 else 0 end)/
    sum(case when email_action = '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'
                else null
            end as email_action
        from
            project_no_3.email_events
    ) a ;
```



Result Grid Filter Rows:

email_open_rate	email_clicked_rate
33.58339	14.78989

RESULT: from this table we get the insight that out of all emails sent around 31.9% were opened and 10.47

Were only clicked.

Project result:

- Less than 0.01 jobs were reviwed every hour of each day of the month of November.
- 7 day rolling average is 0.03 events per second
- Persian language is the highest share among all .
- The maximum users using macbook pro.
- Only 31.9% emails are opened
- The weekly users engagement is higher on 30th week
- And lowest on 30th week

Conclusion:

This project help me gain knowledge about mysql how to import large data and improve my logical thinking in query.