

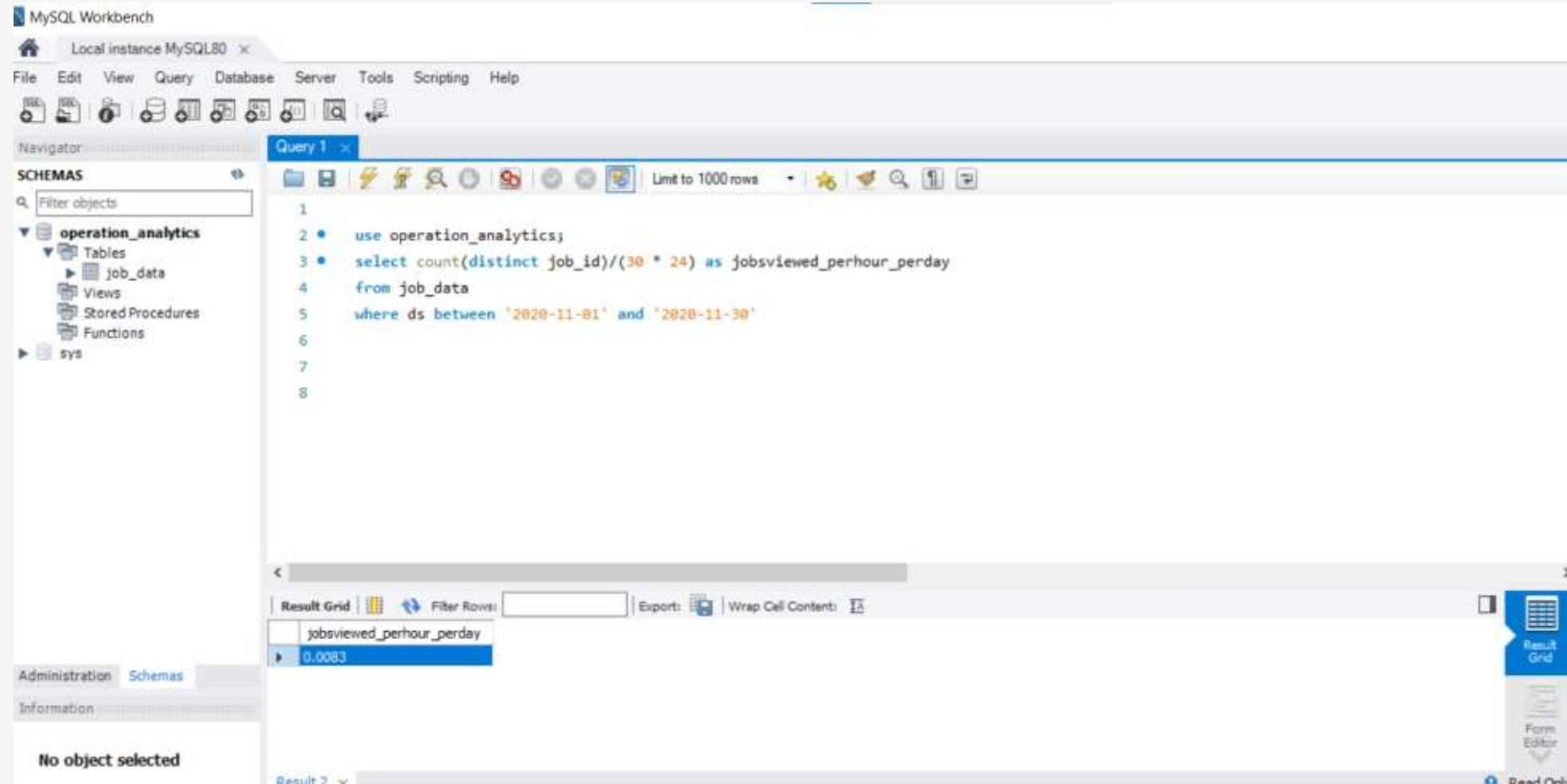
OPERATION ANALYTICS AND INVESTIGATING  
METRIC SPIKE.



## CASE STUDY 1:

1A. Number of jobs reviewed : Amount of jobs reviewed over time.

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



The screenshot shows the MySQL Workbench interface. On the left, the 'SCHEMAS' pane displays the 'operation\_analytics' database with tables 'job\_data' and 'views'. The main editor window shows a SQL query for 'Query 1'.

```
1  
2 • use operation_analytics;  
3 • select count(distinct job_id)/(30 * 24) as jobsviewed_perhour_perday  
4   from job_data  
5   where ds between '2020-11-01' and '2020-11-30'  
6  
7  
8
```

The result grid at the bottom shows one row with the value 0.0083 for the column 'jobsviewed\_perhour\_perday'.

jobsviewed_perhour_perday
0.0083

**1B.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?

The screenshot shows the MySQL Workbench interface. The left sidebar displays the 'SCHEMAS' tree with 'operation\_analytics' expanded, showing 'job\_data' under 'Tables'. The main editor window, titled 'Query 1', contains the following SQL query:

```
3
4 * select
5   ds,jobs_reviewed,
6   avg(jobs_reviewed) over(order by ds rows between 6 preceding and current row)
7   as throughput
8 from
9 (
10  select ds,count(distinct job_id) as jobs_reviewed
11  from job_data
12  where ds between '2020-11-01' and '2020-11-30'
13  group by ds
14  order by ds
15 )b
16
```

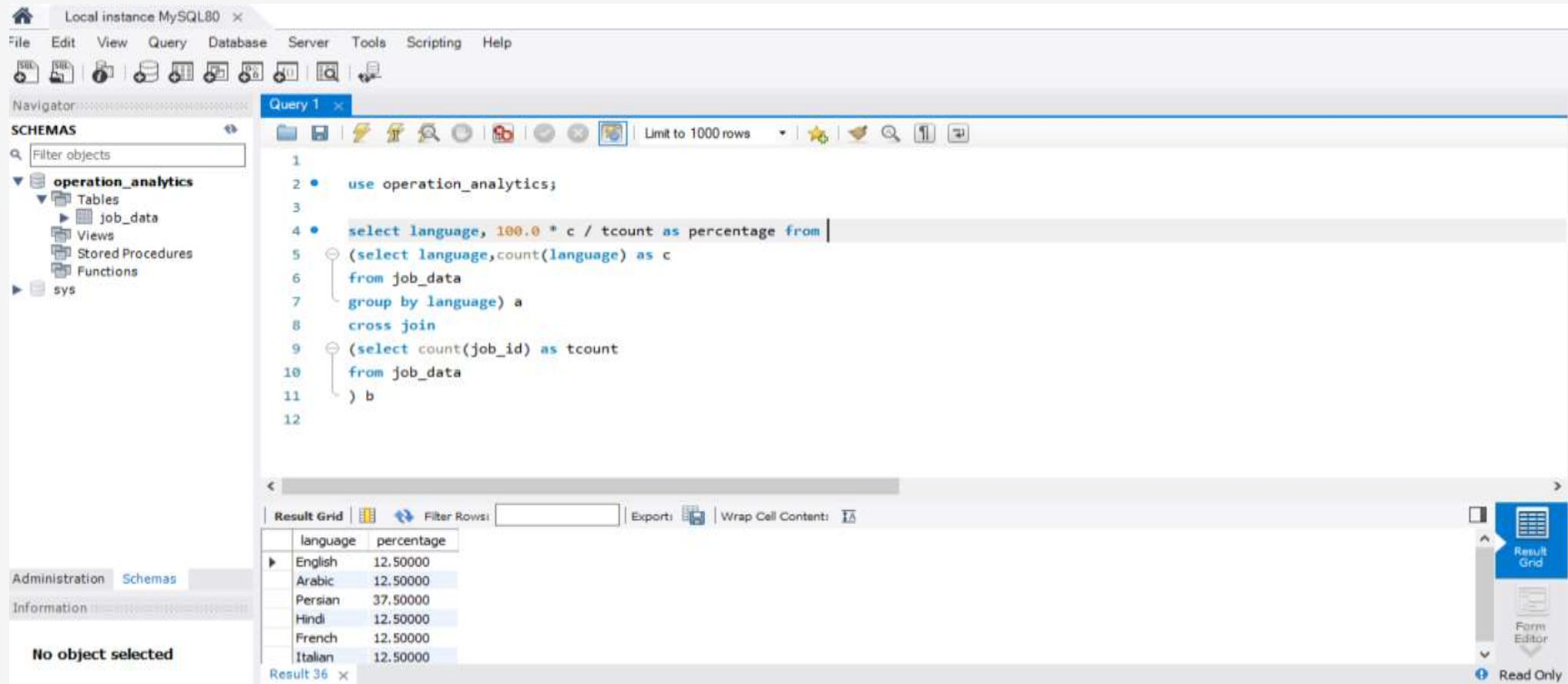
Below the query editor, the 'Result Grid' tab is active, displaying the following data:

ds	jobs_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

The bottom status bar indicates 'No object selected' and 'Read Only'.

## 1C. 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?



The screenshot shows the MySQL Workbench interface. The left sidebar displays the 'SCHEMAS' tree with 'operation\_analytics' expanded, showing 'job\_data' under 'Tables'. The main editor window shows a SQL query in 'Query 1':

```
1
2 use operation_analytics;
3
4 select language, 100.0 * c / tcount as percentage from
5 (select language, count(language) as c
6  from job_data
7  group by language) a
8 cross join
9 (select count(job_id) as tcount
10  from job_data
11  ) b
12
```

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

language	percentage
English	12.50000
Arabic	12.50000
Persian	37.50000
Hindi	12.50000
French	12.50000
Italian	12.50000

The bottom status bar indicates 'No object selected' and 'Result 36 x'. The right sidebar shows 'Result Grid' and 'Form Editor' buttons, with a 'Read Only' indicator at the bottom.

**1D.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?

The screenshot shows the MySQL Workbench interface. On the left, the 'SCHEMAS' pane shows the 'operation\_analytics' database selected, with 'job\_data' table visible under 'Tables'. The main editor window, titled 'Query 1', contains the following SQL query:

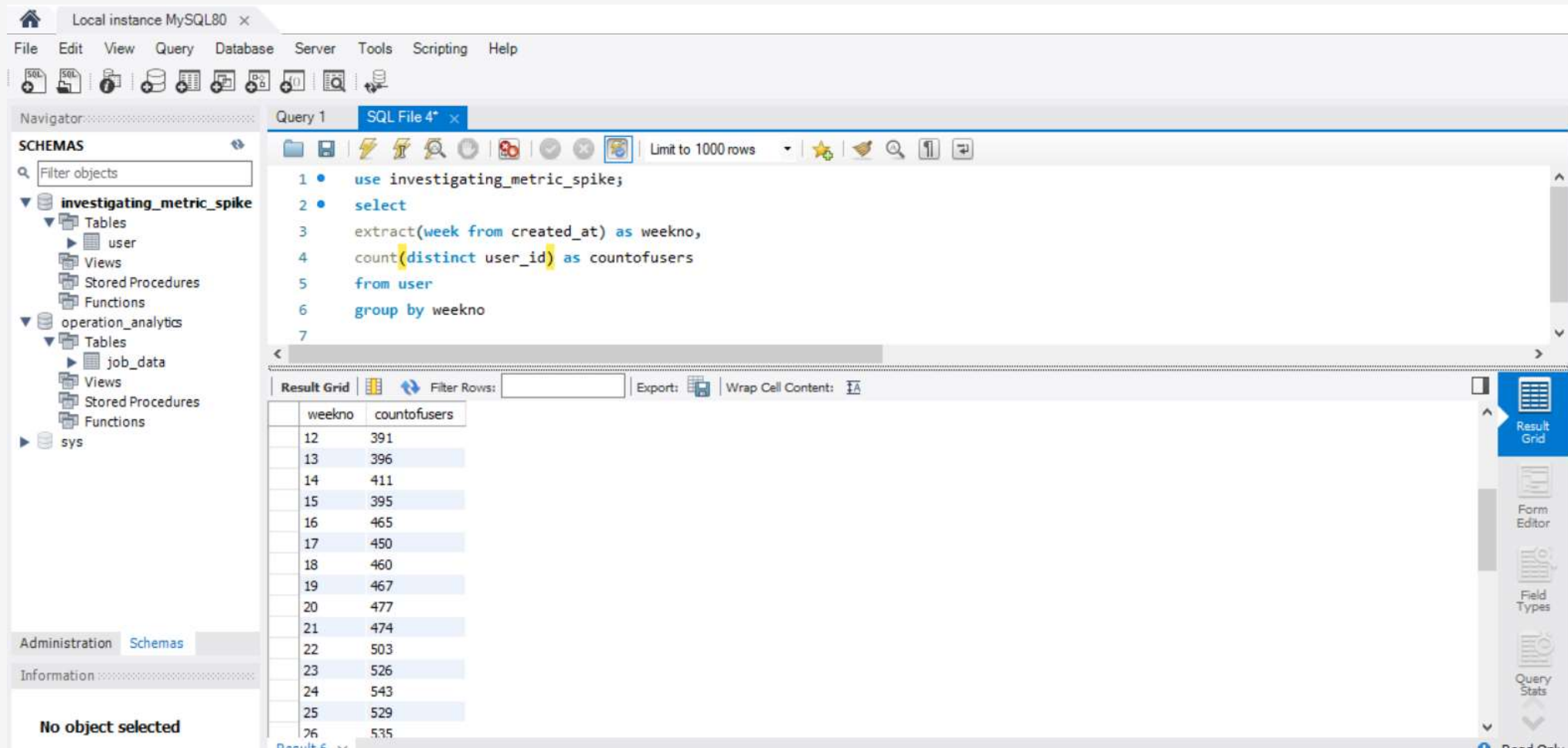
```
1
2 • use operation_analytics;
3
4 • select count(*) as allcount, ds, job_id, actor_id, event, language, time_spent, org
5   from job_data
6   group by ds, job_id, actor_id, event, language, time_spent, org
7   having allcount>1
8
```

Below the query editor, the 'Result Grid' tab is active, showing a table with the following columns: allcount, ds, job\_id, actor\_id, event, language, time\_spent, org. The table is currently empty. The bottom status bar indicates 'Result 43' and 'Read Only'.

## CASE STUDY-2:

**2A.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?



The screenshot shows the MySQL Workbench interface. The left sidebar displays the 'SCHEMAS' tree with 'investigating\_metric\_spike' expanded, showing 'Tables' (user, job\_data) and 'Views'. The main editor shows a SQL query in 'Query 1' that uses the 'investigating\_metric\_spike' database and selects the count of distinct users by week. The 'Result Grid' at the bottom displays the query results.

```
1 use investigating_metric_spike;
2 select
3   extract(week from created_at) as weekno,
4   count(distinct user_id) as countofusers
5 from user
6 group by weekno
```

weekno	countofusers
12	391
13	396
14	411
15	395
16	465
17	450
18	460
19	467
20	477
21	474
22	503
23	526
24	543
25	529
26	535

**2B.User Growth:** Amount of users growing over time for a product.  
**Your task:** Calculate the user growth for product?

The screenshot shows the MySQL Workbench interface. The left sidebar displays the 'SCHEMAS' tree with the following structure:

- investigating\_metric\_spike
  - Tables
    - user
  - Views
  - Stored Procedures
  - Functions
- operation\_analytics
  - Tables
    - job\_data
  - Views
  - Stored Procedures
  - Functions
- sys

The main editor window shows the following SQL query:

```
2 select yearno, weekno, countofusers,  
3 sum(countofusers) over(order by yearno, weekno rows between unbounded preceding and current row)  
4 as usergrowth  
5 from  
6 (  
7 select  
8 extract(week from activated_at) as weekno,  
9 extract(year from activated_at) as yearno,  
10 count(distinct user_id) as countofusers  
11 from user  
12 where state='active'  
13 group by yearno, weekno  
14 order by yearno, weekno) a  
15
```

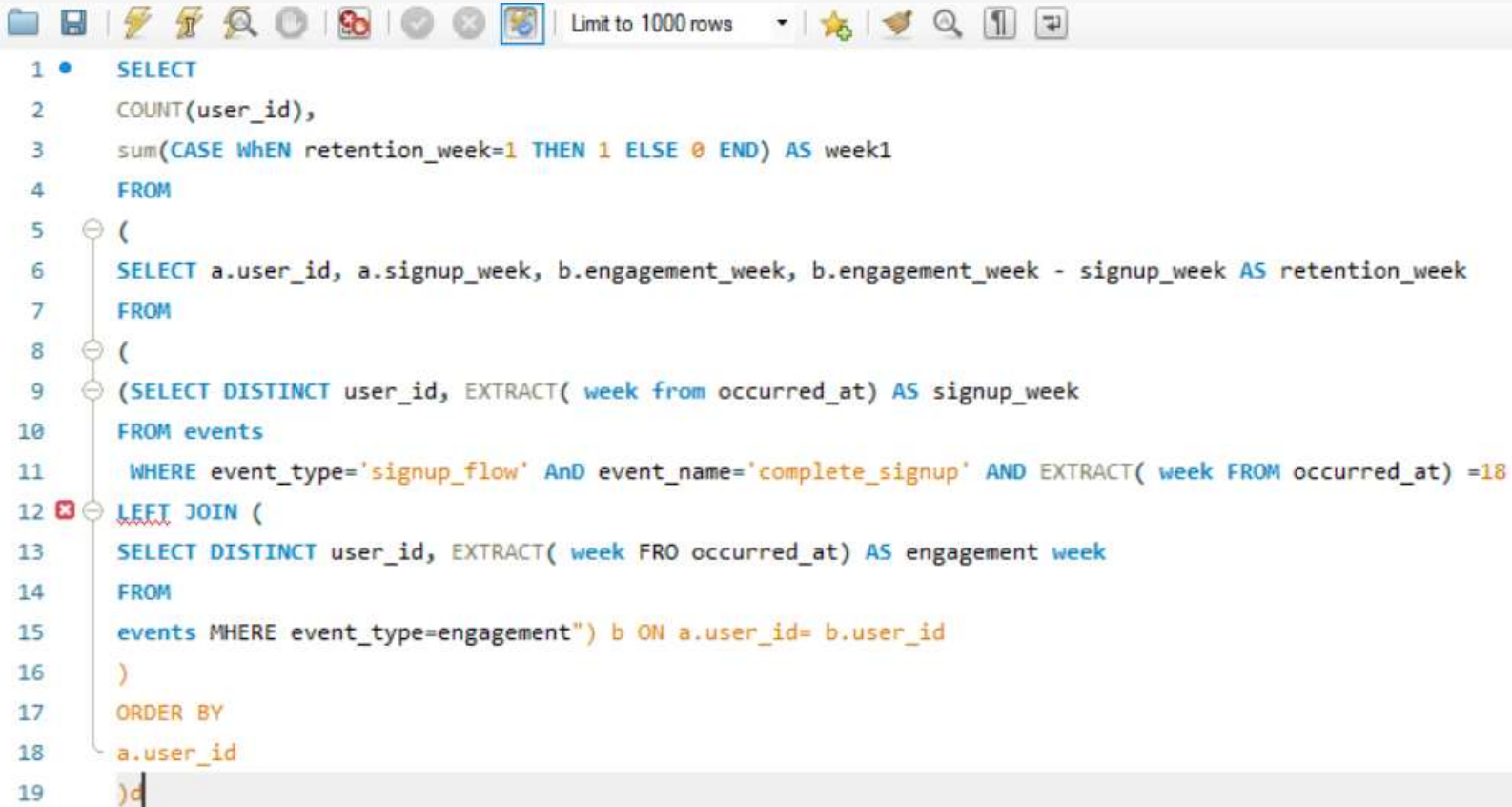
The 'Result Grid' at the bottom displays the following data:

yearno	weekno	countofusers	usergrowth
2013	0	23	23
2013	1	30	53
2013	2	48	101
2013	3	36	137
2013	4	30	167



**2C.Weekly Retention:** Users getting retained weekly after signing-up for a product.

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



The screenshot shows a SQL editor interface with a toolbar at the top containing icons for file operations, search, and execution. A status bar indicates "Limit to 1000 rows". The SQL query is as follows:

```
1 • SELECT
2   COUNT(user_id),
3   sum(CASE WHEN retention_week=1 THEN 1 ELSE 0 END) AS week1
4 FROM
5 (
6   SELECT a.user_id, a.signup_week, b.engagement_week, b.engagement_week - signup_week AS retention_week
7   FROM
8   (
9     (SELECT DISTINCT user_id, EXTRACT( week from occurred_at) AS signup_week
10    FROM events
11     WHERE event_type='signup_flow' AND event_name='complete_signup' AND EXTRACT( week FROM occurred_at) =18
12  LEFT JOIN (
13    SELECT DISTINCT user_id, EXTRACT( week FRO occurred_at) AS engagement week
14    FROM
15    events MHERE event_type=engagement") b ON a.user_id= b.user_id
16  )
17  ORDER BY
18  a.user_id
19 )d
```



**2D.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?

---

se Server Tools Scripting Help



Query 1

SQL File 4\*

SQL File 5\*

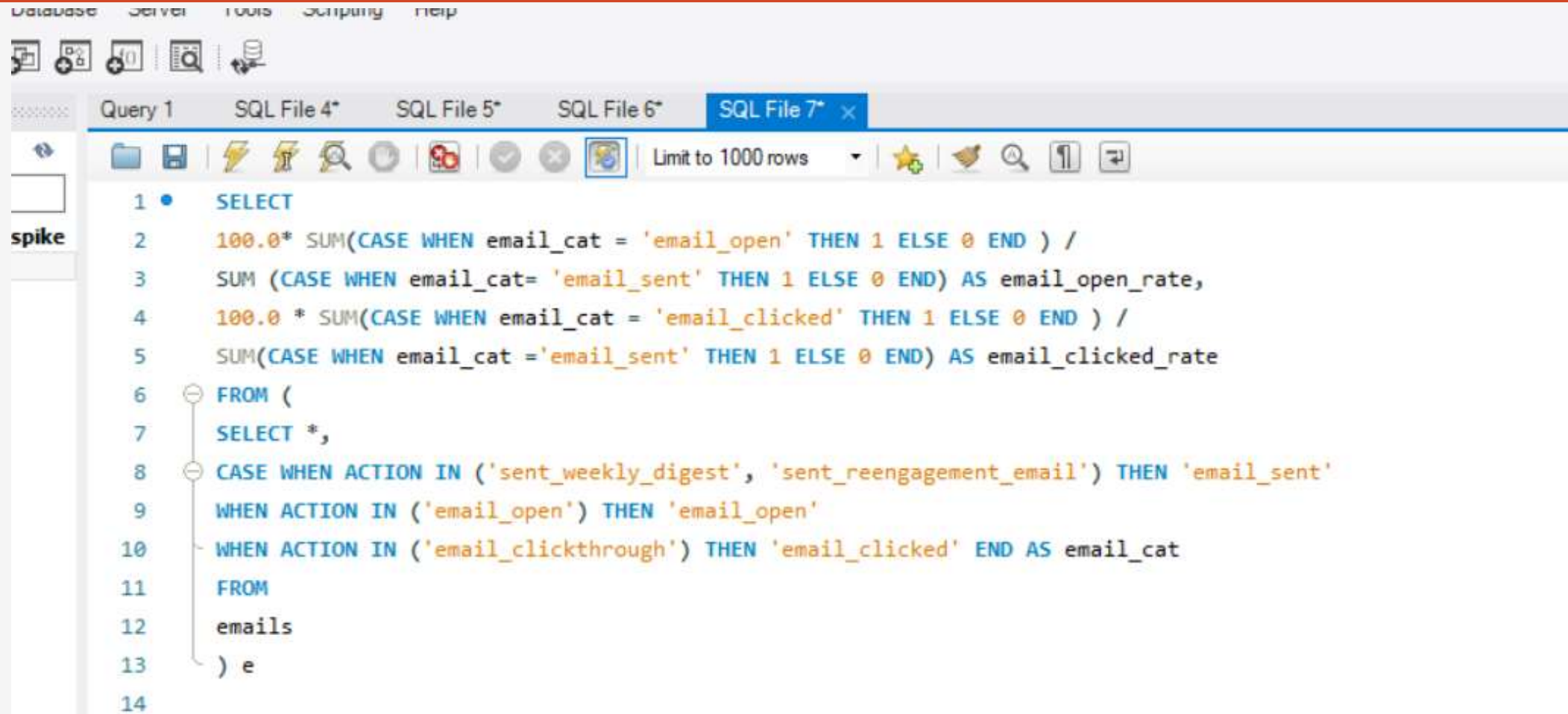
SQL File 6\* x



```
1 • SELECT extract(year FROM occurred_at) AS year,  
2 ❌ extract (week FROM occurred_at) AS week, device, COUNT(distinct user_id) FROM events  
3 WHERE event_type='engagement'  
4 GROUP BY 1,2,3  
5 ORDER BY 1,2,3
```

**2E.Email Engagement:** Users engaging with the email service.

**Your task:** Calculate the email engagement metrics?



The screenshot shows a SQL IDE interface with a menu bar (Database, Server, Tools, Scripting, Help) and a toolbar. The active window is 'SQL File 7\*'. The query editor displays the following SQL code:

```
1 • SELECT
2 100.0* SUM(CASE WHEN email_cat = 'email_open' THEN 1 ELSE 0 END ) /
3 SUM (CASE WHEN email_cat= 'email_sent' THEN 1 ELSE 0 END) AS email_open_rate,
4 100.0 * SUM(CASE WHEN email_cat = 'email_clicked' THEN 1 ELSE 0 END ) /
5 SUM(CASE WHEN email_cat ='email_sent' THEN 1 ELSE 0 END) AS email_clicked_rate
6 FROM (
7 SELECT *,
8 CASE WHEN ACTION IN ('sent_weekly_digest', 'sent_reengagement_email') THEN 'email_sent'
9 WHEN ACTION IN ('email_open') THEN 'email_open'
10 WHEN ACTION IN ('email_clickthrough') THEN 'email_clicked' END AS email_cat
11 FROM
12 emails
13 ) e
14
```

On the left side of the IDE, there is a sidebar with a 'spike' icon and a tree view showing a folder structure.