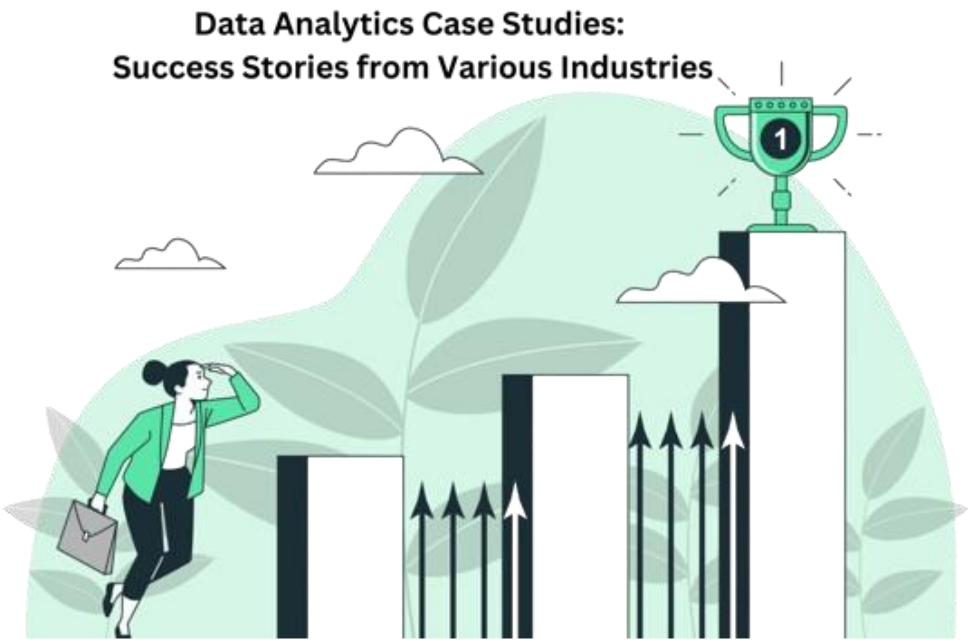

Data Analytics Case Studies:
Success Stories from Various Industries
1

Operation Analytics and Investigating Metric Spike

BY

SHNGAIN KUPAR SHULLAI

Ngain

17042024

Overview

Operational Analytics is a crucial process that involves analysing a company's end-to-end operations. This analysis helps identify areas for improvement within the company.

As a Data Analyst, I'll work closely with various teams, such as operations, support, and marketing, helping them derive valuable insights from the data We collect.

PURPOSE

One of the key aspects of Operational Analytics is investigating metric spikes. This involves understanding and explaining sudden changes in key metrics, such as a dip in daily user engagement or a drop in sales.

As a Data Analyst, I’ll need to answer these questions daily, making it crucial to understand how to investigate these metric spikes. In this project, I'll take on the role of a Lead Data Analyst at a company like HDFC bank, Tesla.

PLAN

Various datasets and tables, and My task will be to derive insights from this data to answer questions posed by different departments within the company.

My goal is to use your advanced SQL skills to analyse the data and provide valuable insights that can help improve the company's operations and understand sudden changes in key metrics.

Tech-Stack Used

- ❖ I use MS Word 2021, Google Drive, MySQL Workbench 8.0 CE to do this analysis.

- ❖ My goal is to find useful information from the data that can help improve the Company in the future.

- ❖ SQL is like a common language for computers, making it easy to use and learn.

WHY I CHOSE SQL FOR THIS CASE STUDY?

- ✓ With SQL, I can quickly find exactly what I’m looking for in my database.

- ✓ Company has tons of data, but SQL it can handle it.

- ✓ SQL helps me find how different parts of Company data are related, showing us interesting patterns.

- ✓ SQL databases have built-in ways to keep Company data safe from unauthorized access.

- ✓ I am deeply passionate about honing my skills with SQL Workbench and other related software tools.

- ✓ My goal is to steadily progress and refine my abilities until I reach a professional level of proficiency.

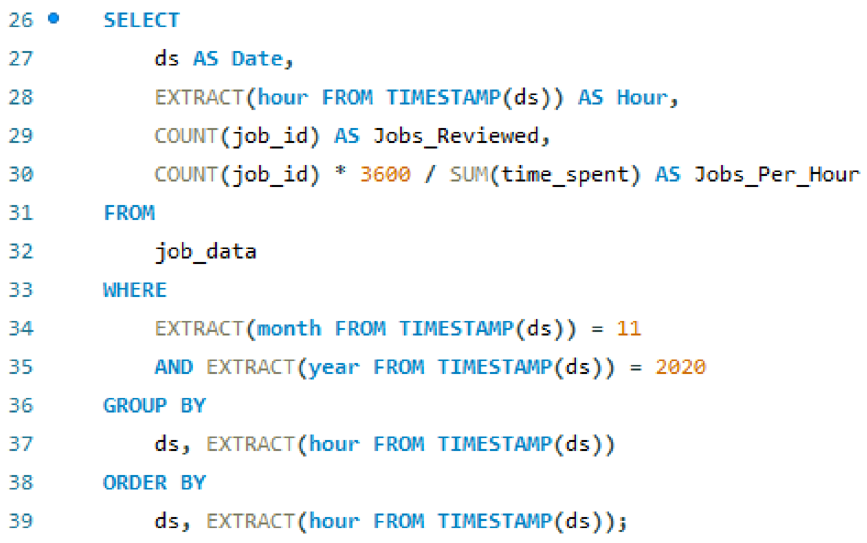
Case Study 1: Job Data Analysis

1. JOBS REVIEWED OVER TIME

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

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

Approach ***Screenshots Attached



```
SELECT
    ds AS Date,
    EXTRACT(hour FROM TIMESTAMP(ds)) AS Hour,
    COUNT(job_id) AS Jobs_Reviewed,
    COUNT(job_id) * 3600 / SUM(time_spent) AS Jobs_Per_Hour
FROM
    job_data
WHERE
    EXTRACT(month FROM TIMESTAMP(ds)) = 11
    AND EXTRACT(year FROM TIMESTAMP(ds)) = 2020
GROUP BY
    ds, EXTRACT(hour FROM TIMESTAMP(ds))
ORDER BY
    ds, EXTRACT(hour FROM TIMESTAMP(ds));
```

Use alias as Date.

A TIMESTAMP in SQL is a data type used to store date and time values. It represents a specific point in time, including both the date (year, month, day) and the time (hour, minute, second).

‘EXTRACT’ helps us extract the hour from the date and time stored in the ds column. Use alias as Hour.

‘COUNT’ tells us how many jobs were reviewed based on the number of times each unique job ID appears. Use alias as Jobs_Reviewed.

COUNT(job_id) calculates how many jobs were reviewed in total. SUM(time_spent) calculates the total time spent reviewing those jobs. By multiplying the count of jobs by 3600 (to convert hours to seconds) and then dividing by the total time spent, we get the rate at which jobs were reviewed per hour. Use alias as Jobs_Per_Hour

‘FROM job_data’ refers table source of my data. ‘11’ means November. This filter ensures that only the rows with dates (‘ds’ column) falling in the month of November are included in the results.

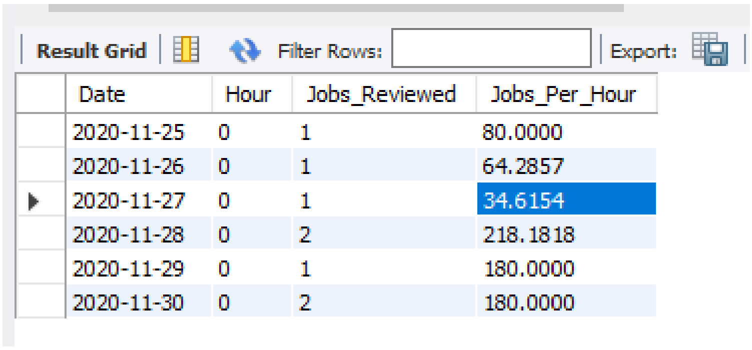
‘2020’ means year. This filter ensures that only the rows with dates (‘ds’ column) falling in the year 2020 are included in the results.

‘GROUP BY ds’ will group together rows that share the same date.

‘EXTRACT’ filter ensures only the rows with dates (‘ds’ column) and picks out just the hour part.

‘EXTRACT’ sorts the rows of the result set first by the date (ds column) in ascending order and then by the hour extracted from the timestamp (ds column).

RESULT



Date	Hour	Jobs_Reviewed	Jobs_Per_Hour
2020-11-25	0	1	80.0000
2020-11-26	0	1	64.2857
2020-11-27	0	1	34.6154
2020-11-28	0	2	218.1818
2020-11-29	0	1	180.0000
2020-11-30	0	2	180.0000

Jobs reviewed indicates how many jobs have been looked at, regardless of any specific details about each job.

On November 28th, 2 jobs were reviewed. The rate of jobs reviewed per hour was 218.18, indicating a high level of productivity in reviewing tasks during that day.

November 29th saw 1 job reviewed, with a rate of 180.00 jobs per hour. This indicates another day of productive reviewing activity, similar to November 28th.

Jobs per hour, on the other hand, represents the rate at which jobs are reviewed per hour.

This metric provides insight into the efficiency of the reviewing process by considering both the quantity of jobs and the time taken to review them.

In summary, I should decide to optimize manpower or technology resources, improve performance, and make more informed strategic decisions.

INSIGHTS

- ✓ I observed fluctuations in the number of jobs reviewed over time, with certain days or hours seeing higher activity levels compared to others.

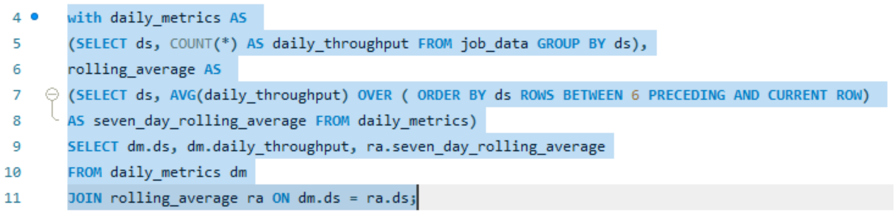
- ✓ While not explicitly analysed in this query, considering language and organization could provide additional insights into reviewing patterns, especially if there are preferences or biases towards certain languages or organizations.

2. THROUGHPUT ANALYSIS

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

My Task: 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.

Approach ***Screenshots Attached



```sql
with daily_metrics AS
(SELECT ds, COUNT(*) AS daily_throughput FROM job_data GROUP BY ds),
rolling_average AS
(SELECT ds, AVG(daily_throughput) OVER ( ORDER BY ds ROWS BETWEEN 6 PRECEDING AND CURRENT ROW)
AS seven_day_rolling_average FROM daily_metrics)
SELECT dm.ds, dm.daily_throughput, ra.seven_day_rolling_average
FROM daily_metrics dm
JOIN rolling_average ra ON dm.ds = ra.ds;
```

Common Table Expression (CTE) named daily_metrics: This part counts how many jobs were processed each day (ds) and gives that count a name, daily_throughput.

It groups the data by day (ds), so each row represents one day's worth of data and how many jobs were processed on that day.

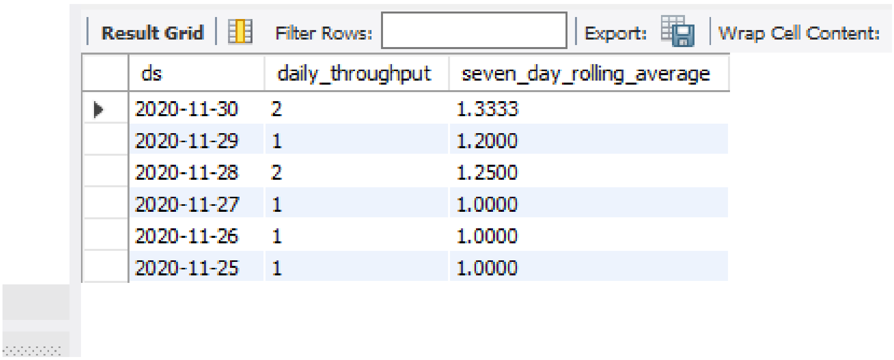
CTE named rolling_average: This section calculates the average daily throughput over the last 7 days for each day.

It uses a window function AVG() to calculate the average of daily_throughput over a sliding window of the previous 7 days, including the current day.

Main Query: It selects the date (ds), the daily throughput (daily_throughput) from the daily_metrics CTE, and the 7-day rolling average throughput (seven_day_rolling_average) from the rolling_average CTE.

It joins the two CTEs together based on the date (ds), so for each day, it shows both the daily throughput and the corresponding 7-day rolling average throughput.

RESULT



ds	daily_throughput	seven_day_rolling_average
2020-11-30	2	1.3333
2020-11-29	1	1.2000
2020-11-28	2	1.2500
2020-11-27	1	1.0000
2020-11-26	1	1.0000
2020-11-25	1	1.0000

These returns provide insights into the daily throughput and the 7-day rolling average of job processing over a specific period, in this case, from November 25th to November 30th, 2020.

Daily Throughput: On November 30th, there were 2 jobs processed.

Seven-day Rolling Average: On November 30th, the 7-day rolling average is 1.33, indicating an average of 1.33 jobs processed per day over the past week.

INSIGHTS

- ✓ In the provided data, we observe fluctuations in both daily throughput and rolling average, indicating some variability and steady workflow in job processing.

- ✓ If the rolling average consistently increases or decreases over time, it reveals trends in job processing efficiency. A steady increase may indicate improving efficiency, while a decline could signify decreasing productivity.

- ✓ If there's a sudden increase in daily throughput without a corresponding rise in the rolling average, it might indicate temporary workload spikes that require additional resources to maintain efficiency.

- ✓ Comparing actual daily throughput with the rolling average helps evaluate performance against expectations. Lower daily throughput compared to the rolling average may prompt investigations into workflow issues.

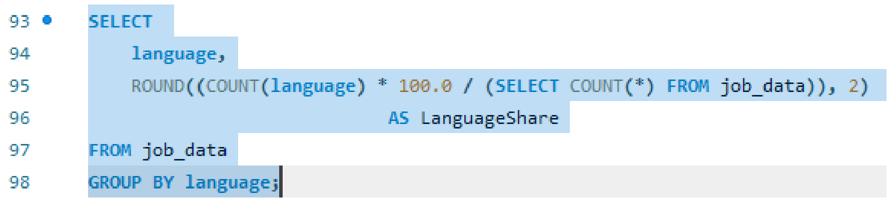
- ✓ In general, analysing daily throughput and the 7-day rolling average provides valuable insights for monitoring workflow effects, detecting trends, and making informed decisions regarding resource allocation and process optimization.

3. LANGUAGE SHARE ANALYSIS

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

My Task: Write an SQL query to calculate the percentage share of each language over the last 30 days.

Approach ***Screenshots Attached



```sql
SELECT
    language,
    ROUND((COUNT(language) * 100.0 / (SELECT COUNT(*) FROM job_data)), 2)
                                    AS LanguageShare
FROM job_data
GROUP BY language;
```

I work at the job_data table and grouping the data based on the different languages used in the language column.

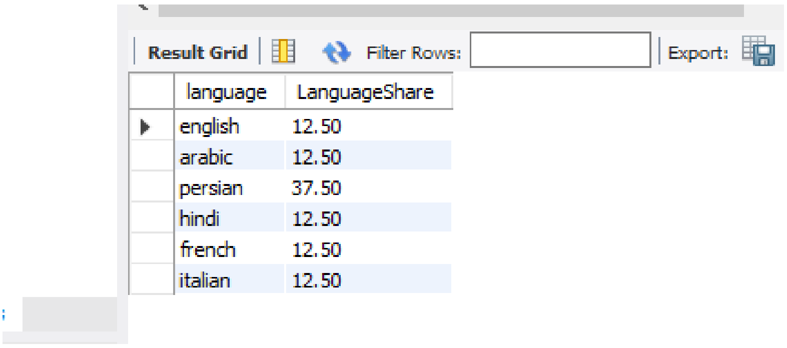
For each language group, I count how many times that language appears in the dataset using COUNT (language). This gives the number of jobs for each language.

To find the percentage share of each language, take the count of jobs for each language and divide it by the total number of jobs in the dataset.

The total number of jobs is obtained by a subquery (SELECT COUNT (*) FROM job_data), which counts all the rows in the job_data table.

The ROUND function is used to round the percentage (Results multiply by 100) share to two (2) decimal places.

RESULT



language	LanguageShare
english	12.50
arabic	12.50
persian	37.50
hindi	12.50
french	12.50
italian	12.50

The final result gives us each language along with its percentage share in the total number of jobs. In general, this query breaks down the job data by language, calculates the percentage share of each language in the total number of jobs.

Persian accounts for 37.50% of the job tasks, indicating it's the most frequently occurring language in the dataset.

INSIGHTS

- ✓ If Persian is the most common language, allocating more resources, such as translators or reviewers proficient in Persian, might improve overall efficiency.

- ✓ If Hindi, English, Arabic, French and Italian have low percentage share, but there's a growing market demand for content in their speaking regions, it might indicate an opportunity to expand services to meet that demand.

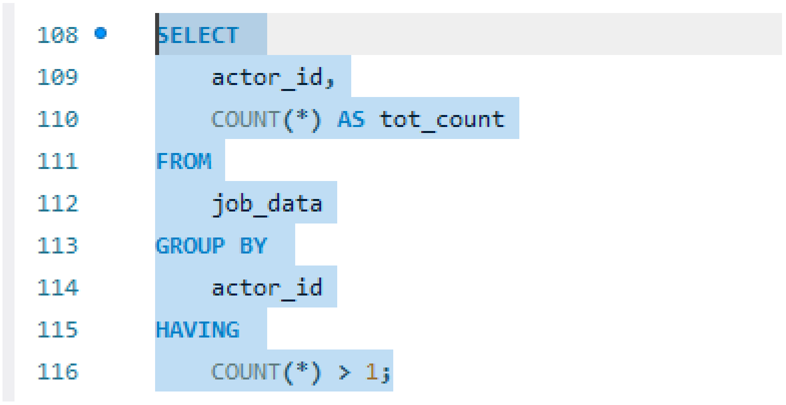
- ✓ This return enables businesses to make informed decisions about where to allocate resources, how to expand into new markets, and how to adapt their overall business direction to meet the needs and preferences of their target audience.

4. DUPLICATE ROWS DETECTION

Objective: Identify duplicate rows in the data.

My Task: Write an SQL query to display duplicate rows from the job_data table.

Approach ***Screenshots Attached



```sql
SELECT
    actor_id,
    COUNT(*) AS tot_count
FROM
    job_data
GROUP BY
    actor_id
HAVING
    COUNT(*) > 1;
```

Counting the appearance of each actor_id.

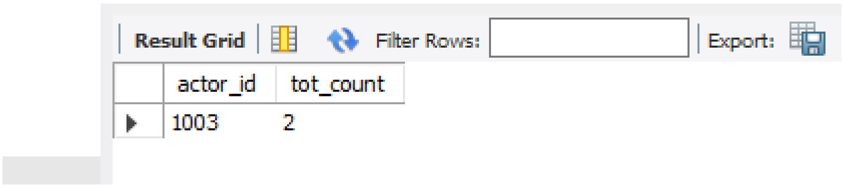
The GROUP BY clause groups the data by actor_id.

The HAVING COUNT (*) > 1 condition ensures that only actor_id values with more than one appearance (i.e., duplicates) are selected.

The COUNT (*) AS tot_count calculates the total count of each actor_id.

This query will provide the actor_id values along with their total count, only including those actor_id values that have duplicates in the job_data table.

RESULT



actor_id	tot_count
1003	2

Multiple appearance of the same ‘actor_id’ 1003 appears twice in the dataset.

INSIGHTS

- ✓ Data integrity can be enhanced, ensuring accurate analysis and decision-making.
- ✓ By addressing duplicate entries for actor 1003, workflow efficiency can be improved, potentially optimizing resource allocation.
- ✓ Monitoring this frequency requires adjustment to meet performance targets.
- ✓ Appropriate adjustments can be made to ensure optimal resource distribution across tasks and actors.
- ✓ In general, recognizing the presence of duplicate entries enhances data quality and drives process optimization efforts for improved overall performance and decision-making.

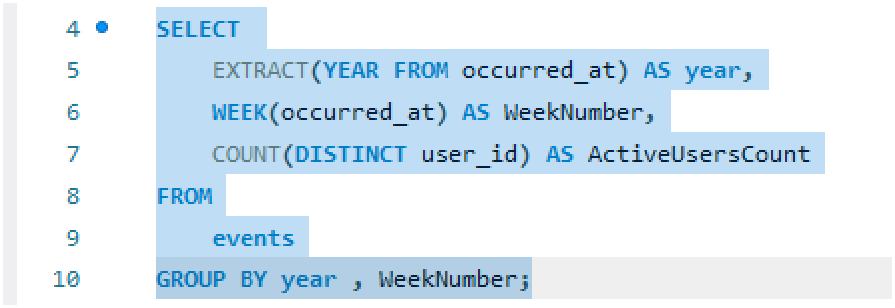
Case Study 2: Investigating Metric Spike

1. WEEKLY USER ENGAGEMENT

Objective: Measure the activeness of users on a weekly basis.

My Task: Write an SQL query to calculate the weekly user engagement.

Approach ***Screenshots Attached



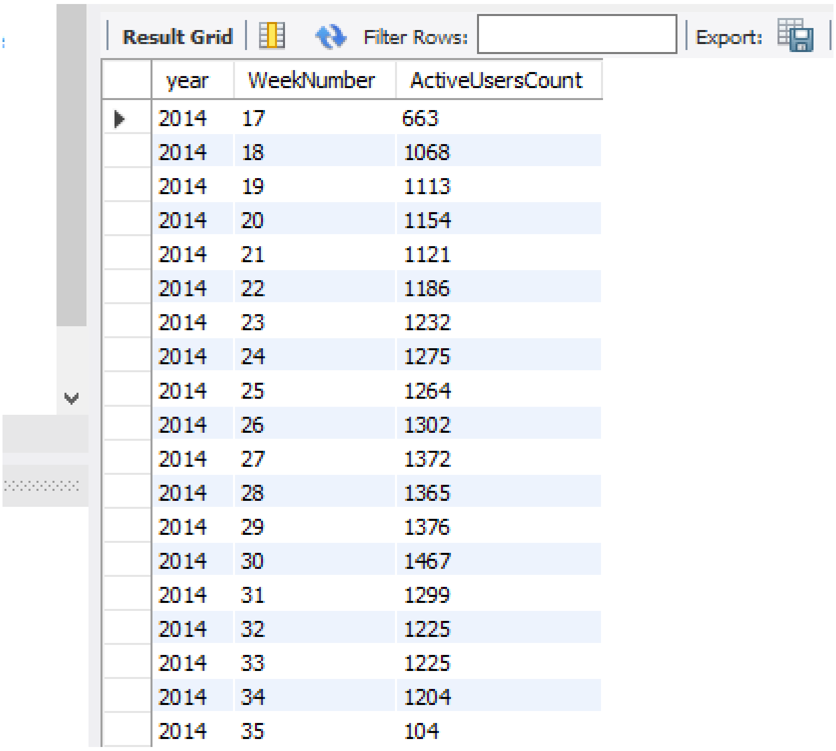
```sql
SELECT
    EXTRACT(YEAR FROM occurred_at) AS year,
    WEEK(occurred_at) AS WeekNumber,
    COUNT(DISTINCT user_id) AS ActiveUsersCount
FROM
    events
GROUP BY year , WeekNumber;
```

Extraction of year part from the occurred_at column. It returns the year in which each event occurred.

Then the extraction of week number from the occurred_at column. It represents the week of the year in which each event occurred.

Then counting the number of unique (Distinct) users for each week.
The results are grouped by them, ensuring that events occurring in the same week of different years are not grouped together.

RESULT



year	WeekNumber	ActiveUsersCount
2014	17	663
2014	18	1068
2014	19	1113
2014	20	1154
2014	21	1121
2014	22	1186
2014	23	1232
2014	24	1275
2014	25	1264
2014	26	1302
2014	27	1372
2014	28	1365
2014	29	1376
2014	30	1467
2014	31	1299
2014	32	1225
2014	33	1225
2014	34	1204
2014	35	104

The return displays the user engagement metrics on a weekly basis throughout the year 2014.

During Week 30, there were 1467 active users, while during Week 35, there were only 104 active users. There are varying levels of user activity over time.

There is a noticeable increase in user activity from Week 30 to Week 31, followed by a slight decrease in Week 32.

INSIGHTS

- ✓ Analysing this data allows for understanding trends in user behaviour, identifying periods of high and low activity, and potentially informing decisions regarding resource allocation, marketing strategies, or product enhancements to better serve user needs.

- ✓ In general, this return provides a snapshot of weekly user engagement metrics over the course of the year, enabling stakeholders to understand and assess user activity patterns and make data-driven decisions accordingly.

- ✓ Higher or lower levels of activity are also due to external factors such as holidays, events, or industry-specific trends.

- ✓ Investigate the causes behind this change and look for correlations with marketing campaigns, product updates, seasonal trends, or external events that may have influenced user behaviour

- ✓ Allocate additional resources for customer care, server strength and advertising during peak activity periods.

- ✓ Moreover, regularly reviewing the effectiveness of implemented changes and adjust tactics as needed to maintain or improve user engagement over time.

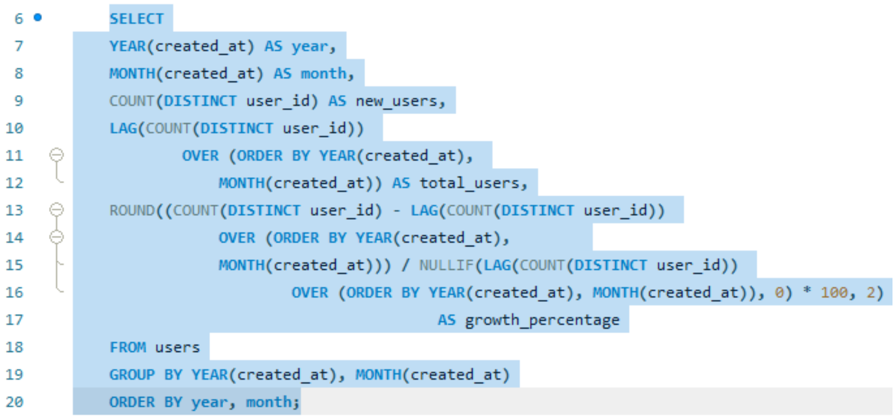
- ✓ By following these steps, stakeholders can effectively leverage the insights and drive continuous improvement in overall business performance.

2. USER GROWTH ANALYSIS

Objective: Analyse the growth of users over time for a product.

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

Approach ***Screenshots Attached



```sql
SELECT
YEAR(created_at) AS year,
MONTH(created_at) AS month,
COUNT(DISTINCT user_id) AS new_users,
LAG(COUNT(DISTINCT user_id))
        OVER (ORDER BY YEAR(created_at),
            MONTH(created_at)) AS total_users,
ROUND((COUNT(DISTINCT user_id) - LAG(COUNT(DISTINCT user_id))
            OVER (ORDER BY YEAR(created_at),
            MONTH(created_at))) / NULLIF(LAG(COUNT(DISTINCT user_id))
                    OVER (ORDER BY YEAR(created_at), MONTH(created_at)), 0) * 100, 2)
                            AS growth_percentage
FROM users
GROUP BY YEAR(created_at), MONTH(created_at)
ORDER BY year, month;
```

Selecting the year and month from the created_at column, which tells us when each user account was created.

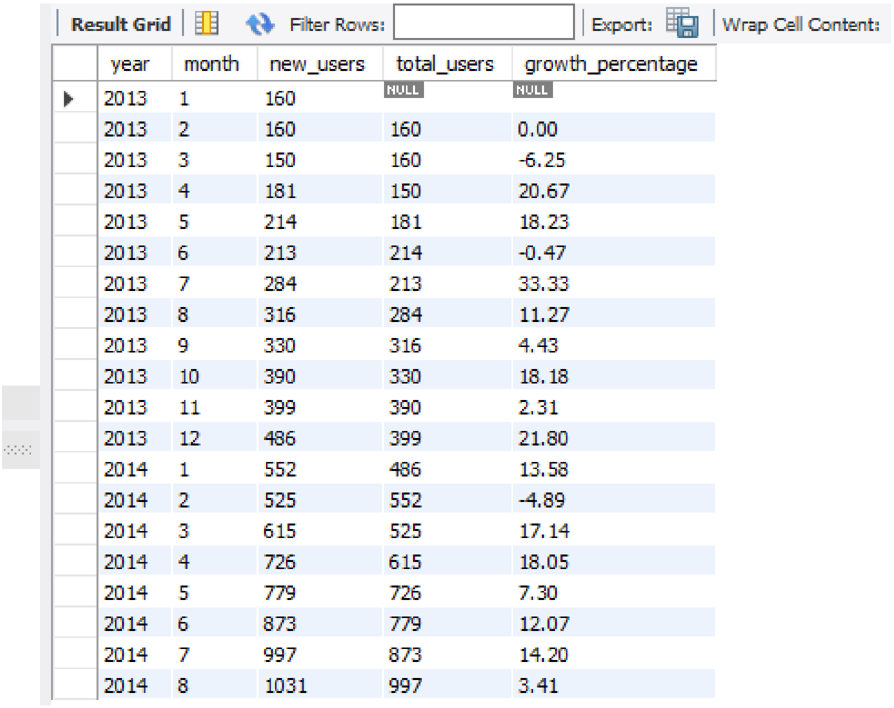
Count the number of new users (new_users) for each month using the COUNT (DISTINCT user_id) function. This gives us the total number of new users in each month.

To find the total number of users up to each month, LAG window function looks at the previous month's count of distinct users and provides it as total_users. This tells us how many users there were before the current month.
The growth percentage is calculated by comparing the difference in new users between the current month and the previous month (new_users - total_users), then dividing it by the total users from the previous month.

I use NULLIF when there were no users in the previous month to avoid division by zero. The result is multiplied by 100 and rounded to two decimal places.

Grouping the results by year and month to organize them, and then ordering them in ascending order by year and month to show the growth progression over time.

RESULT



year	month	new_users	total_users	growth_percentage
2013	1	160	NULL	NULL
2013	2	160	160	0.00
2013	3	150	160	-6.25
2013	4	181	150	20.67
2013	5	214	181	18.23
2013	6	213	214	-0.47
2013	7	284	213	33.33
2013	8	316	284	11.27
2013	9	330	316	4.43
2013	10	390	330	18.18
2013	11	399	390	2.31
2013	12	486	399	21.80
2014	1	552	486	13.58
2014	2	525	552	-4.89
2014	3	615	525	17.14
2014	4	726	615	18.05
2014	5	779	726	7.30
2014	6	873	779	12.07
2014	7	997	873	14.20
2014	8	1031	997	3.41

In January 2013, 160 new users were added and there were 160 total users.

In March 2013, there were 150 new users, resulting in a 6.25% (negative) decrease compared to the previous month's total users.

In July 2013, there was a significant increase, with 284 new users, representing a 33.33% growth compared to the previous month.

August 2014 saw 1,031 new users, resulting in a 3.41% growth compared to the previous month's total users.

INSIGHTS

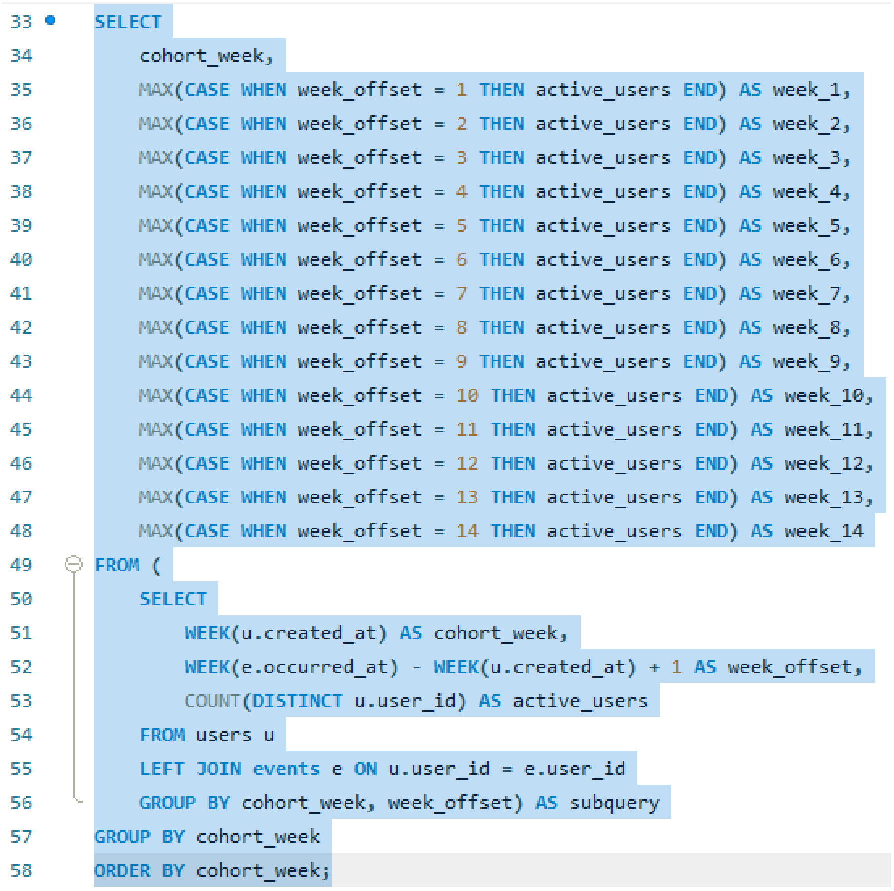
- ✓ The percentage growth of new users compared to the previous month's total users, provides valuable insights into the product's user acquisition and growth trends.
- ✓ It helps stakeholders understand how user numbers change over time and can guide decision-making related to marketing strategies, product development, and resource allocation.
- ✓ Allocate resources towards campaigns that drive the highest user acquisition rates, while adjusting or discontinuing underperforming initiatives.
- ✓ Understand growth correlates with product updates prioritize future development efforts.
- ✓ Allocate budgets, manpower, and other resources towards areas that contribute most effectively.
- ✓ Moreover, regularly review user growth trends to adapt strategies as needed and ensure alignment with long-term business objectives.

3. WEEKLY RETENTION ANALYSIS

Objective: Analyse the retention of users on a weekly basis after signing up for a product.

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

Approach ***Screenshots Attached



```
SELECT
    cohort_week,
    MAX(CASE WHEN week_offset = 1 THEN active_users END) AS week_1,
    MAX(CASE WHEN week_offset = 2 THEN active_users END) AS week_2,
    MAX(CASE WHEN week_offset = 3 THEN active_users END) AS week_3,
    MAX(CASE WHEN week_offset = 4 THEN active_users END) AS week_4,
    MAX(CASE WHEN week_offset = 5 THEN active_users END) AS week_5,
    MAX(CASE WHEN week_offset = 6 THEN active_users END) AS week_6,
    MAX(CASE WHEN week_offset = 7 THEN active_users END) AS week_7,
    MAX(CASE WHEN week_offset = 8 THEN active_users END) AS week_8,
    MAX(CASE WHEN week_offset = 9 THEN active_users END) AS week_9,
    MAX(CASE WHEN week_offset = 10 THEN active_users END) AS week_10,
    MAX(CASE WHEN week_offset = 11 THEN active_users END) AS week_11,
    MAX(CASE WHEN week_offset = 12 THEN active_users END) AS week_12,
    MAX(CASE WHEN week_offset = 13 THEN active_users END) AS week_13,
    MAX(CASE WHEN week_offset = 14 THEN active_users END) AS week_14
FROM (
    SELECT
        WEEK(u.created_at) AS cohort_week,
        WEEK(e.occurred_at) - WEEK(u.created_at) + 1 AS week_offset,
        COUNT(DISTINCT u.user_id) AS active_users
    FROM users u
    LEFT JOIN events e ON u.user_id = e.user_id
    GROUP BY cohort_week, week_offset) AS subquery
GROUP BY cohort_week
ORDER BY cohort_week;
```

The inner query first calculates the week number of each user's sign-up date (cohort_week) and the week offset between the sign-up week and subsequent activity weeks (week_offset).

It counts the number of distinct active users for each combination of cohort_week and week_offset.

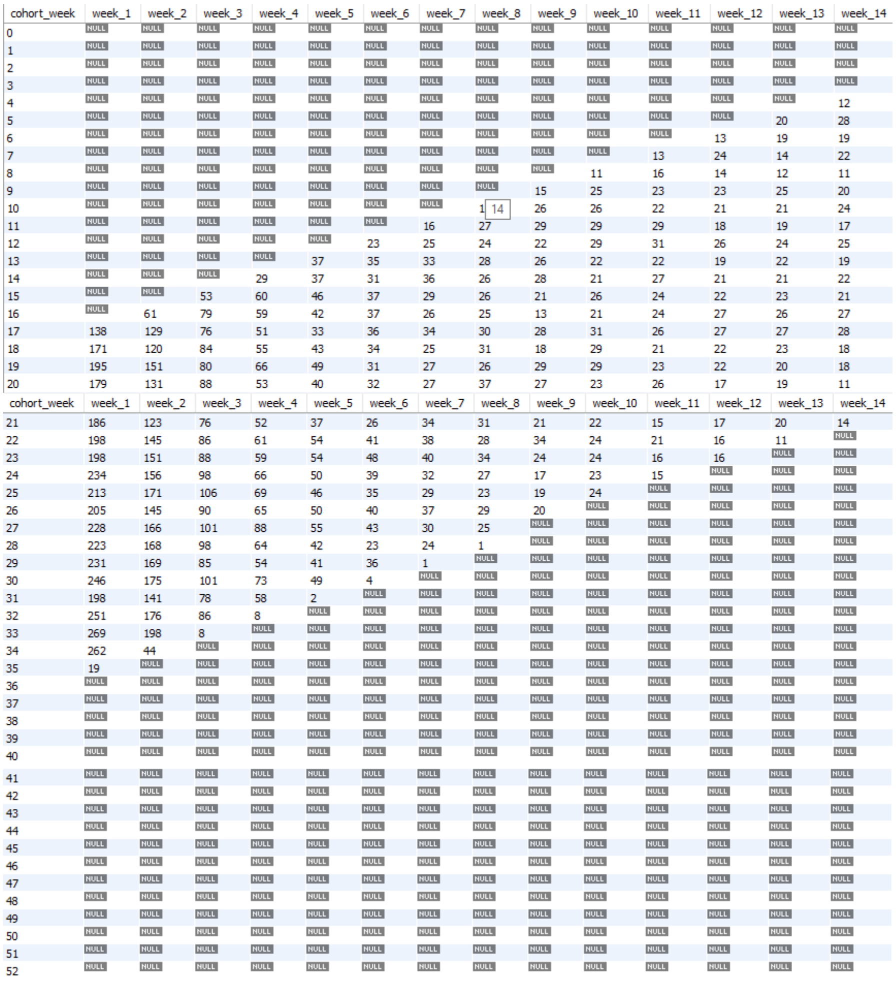
The outer query then pivots the results of the subquery, creating columns for each week offset (week_1, week_2, ..., week_14). It uses conditional aggregation to populate these columns with the maximum count of active users for each respective week offset.

The results are grouped by cohort_week, ensuring that each row represents a unique sign-up cohort. Finally, the results are ordered by cohort_week to present the data in chronological order.

RESULT

In the first row (cohort_week = 0), all subsequent weeks are empty, indicating that there were no active users beyond the first week for this cohort. This suggests that users from this cohort did not retain engagement beyond the first week.

Cohorts exhibit steady retention over time, with active users decreasing gradually week by week. Others show fluctuations in retention, with some weeks having higher or lower engagement compared to others.



cohort_week	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
0	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL
1	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL
2	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL
3	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL
4	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	12
5	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	20	28
6	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	13	19	19
7	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	13	24	14	22
8	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	11	16	14	12	11
9	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	15	25	23	23	25	20
10	NULL	NULL	NULL	NULL	NULL	NULL	NULL	1 14	26	26	22	21	21	24
11	NULL	NULL	NULL	NULL	NULL	NULL	16	27	29	29	29	18	19	17
12	NULL	NULL	NULL	NULL	NULL	23	25	24	22	29	31	26	24	25
13	NULL	NULL	NULL	NULL	37	35	33	28	26	22	22	19	22	19
14	NULL	NULL	NULL	29	37	31	36	26	28	21	27	21	21	22
15	NULL	NULL	53	60	46	37	29	26	21	26	24	22	23	21
16	NULL	61	79	59	42	37	26	25	13	21	24	27	26	27
17	138	129	76	51	33	36	34	30	28	31	26	27	27	28
18	171	120	84	55	43	34	25	31	18	29	21	22	23	18
19	195	151	80	66	49	31	27	26	29	29	23	22	20	18
20	179	131	88	53	40	32	27	37	27	23	26	17	19	11
21	186	123	76	52	37	26	34	31	21	22	15	17	20	14
22	198	145	86	61	54	41	38	28	34	24	21	16	11	NULL
23	198	151	88	59	54	48	40	34	24	24	16	16	NULL	NULL
24	234	156	98	66	50	39	32	27	17	23	15	NULL	NULL	NULL
25	213	171	106	69	46	35	29	23	19	24	NULL	NULL	NULL	NULL
26	205	145	90	65	50	40	37	29	20	NULL	NULL	NULL	NULL	NULL
27	228	166	101	88	55	43	30	25	NULL	NULL	NULL	NULL	NULL	NULL
28	223	168	98	64	42	23	24	1	NULL	NULL	NULL	NULL	NULL	NULL
29	231	169	85	54	41	36	1	NULL	NULL	NULL	NULL	NULL	NULL	NULL
30	246	175	101	73	49	4	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL
31	198	141	78	58	2	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL
32	251	176	86	8	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL
33	269	198	8	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL
34	262	44	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL
35	19	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL
36	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL
37	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL
38	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL
39	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL
40	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL
41	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL
42	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL
43	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL
44	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL
45	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL
46	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL
47	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL
48	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL
49	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL
50	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL
51	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL
52	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL

This matrix like structure allows us to visually assess user retention trends over time, helping stakeholders understand how user engagement evolves after sign-up. It can inform decisions related to user engagement strategies, product improvements, and resource allocation to optimize user retention and maximize long-term user value.

INSIGHTS

- ✓ This help understand how user engagement evolves after sign-up. It can inform decisions related to user engagement strategies, product improvements, and resource allocation to optimize user retention and maximize long-term user value.

- ✓ By examining the values across the different weeks for each cohort, I can identify trends in user retention over time.

- ✓ The retention rates of cohorts vary between groups of users who signed up at different times helps identify factors influencing retention, such as changes in product features or marketing strategies.

- ✓ These cohorts may indicate successful onboarding processes or product improvements that encourage ongoing user activity.

- ✓ Cohorts with steep declines suggests potential issues with user onboarding, product usability, or initial value delivery that may need to be addressed.

- ✓ These implemented changes aimed at improving retention and to assess their effectiveness.

- ✓ Cohorts signing up during holiday seasons or special events may exhibit different retention patterns compared to those signing up during regular periods.

- ✓ Identifying trends and compare retention rates between weekdays and weekends or between users from different geographic regions.

- ✓ Use historical retention data to forecast future user engagement and anticipate potential challenges or opportunities.

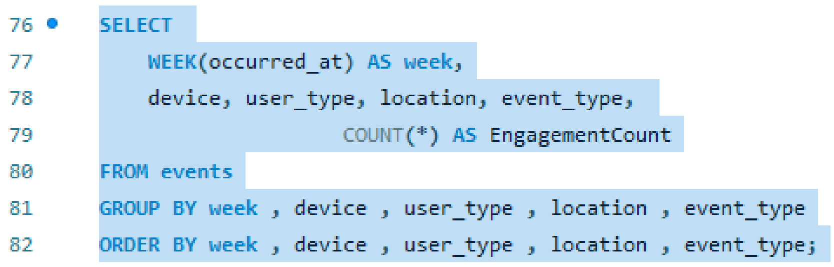
- ✓ Thus, this analysis can gain valuable insights into user retention dynamics and make informed decisions to optimize user engagement and drive business growth.

4. WEEKLY ENGAGEMENT PER DEVICE

Objective: Measure the activeness of users on a weekly basis per device.

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

Approach *****Screenshots Attached**



```
SELECT
    WEEK(occurred_at) AS week,
    device, user_type, location, event_type,
                COUNT(*) AS EngagementCount
FROM events
GROUP BY week , device , user_type , location , event_type
ORDER BY week , device , user_type , location , event_type;
```

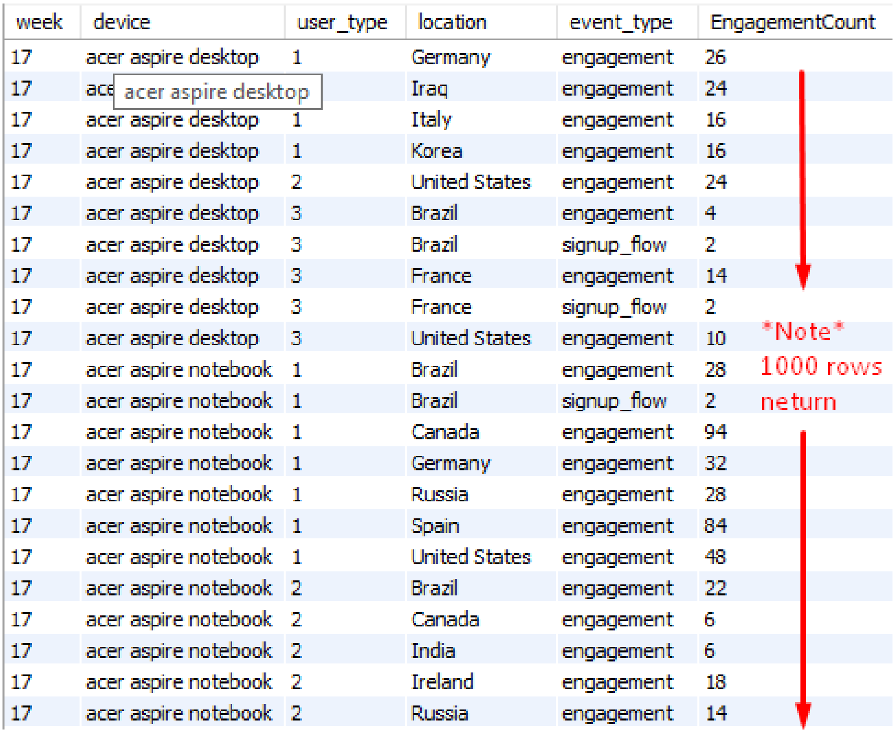
'device', 'user_type', 'location', and 'event_type' are columns included to provide more details about the events.

The GROUP BY clause now includes all these columns along with the week based on their attributes.

The ORDER BY clause includes columns to sort the results in any formal order.

RESULT

Here I can observe trends in engagement behaviour based on the type of device used, user type, geographical location, and the specific types of events performed.



week	device	user_type	location	event_type	EngagementCount
17	acer aspire desktop	1	Germany	engagement	26
17	acer aspire desktop	[illegible]	Iraq	engagement	24
17	acer aspire desktop	1	Italy	engagement	16
17	acer aspire desktop	1	Korea	engagement	16
17	acer aspire desktop	2	United States	engagement	24
17	acer aspire desktop	3	Brazil	engagement	4
17	acer aspire desktop	3	Brazil	signup_flow	2
17	acer aspire desktop	3	France	engagement	14
17	acer aspire desktop	3	France	signup_flow	2
17	acer aspire desktop	3	United States	engagement	10
17	acer aspire notebook	1	Brazil	engagement	28
17	acer aspire notebook	1	Brazil	signup_flow	2
17	acer aspire notebook	1	Canada	engagement	94
17	acer aspire notebook	1	Germany	engagement	32
17	acer aspire notebook	1	Russia	engagement	28
17	acer aspire notebook	1	Spain	engagement	84
17	acer aspire notebook	1	United States	engagement	48
17	acer aspire notebook	2	Brazil	engagement	22
17	acer aspire notebook	2	Canada	engagement	6
17	acer aspire notebook	2	India	engagement	6
17	acer aspire notebook	2	Ireland	engagement	18
17	acer aspire notebook	2	Russia	engagement	14

Note
1000 rows
neturn

Samsung Galaxy S4: In week 17, users of Samsung Galaxy S4 in different locations (Netherlands, Pakistan, United Kingdom, United States) engaged with the product performing various events showing engagement count ranged from 6 to 34. Additionally, in week 17, there were 2 engagements (signup flows) specifically from users in the United States.

Windows Devices: In week 17, users from Korea, Norway, Sweden, and the United States performed various engagement events, with engagement counts ranging from 4 to 60. Different user types (1, 2, 3) on Windows devices participated in engagements, indicating diverse user demographics.

Acer Aspire Desktop: In week 18, users with Acer Aspire Desktop devices engaged from various locations (Argentina, Germany, Iraq, Italy, Japan, Korea, Taiwan, United Arab Emirates, United Kingdom, United States, Brazil, Canada).

Engagement having higher engagement counts like United States with 104 engagements compared to others.

INSIGHTS

- ✓ This breakdown by device helps stakeholders understand how users on different devices interact with the product, allowing for targeted optimizations and improvements tailored to each device type.

- ✓ This granular level of analysis enables stakeholders to make informed decisions about email marketing strategies, user experience optimization, and targeting specific user segments based on their engagement patterns.

- ✓ Various devices, such as smartphones, tablets, and desktops reveal which devices are more popular among users for engaging with the platform.

- ✓ Analysing engagement by location can provide insights into regional differences in user behaviour and preferences.

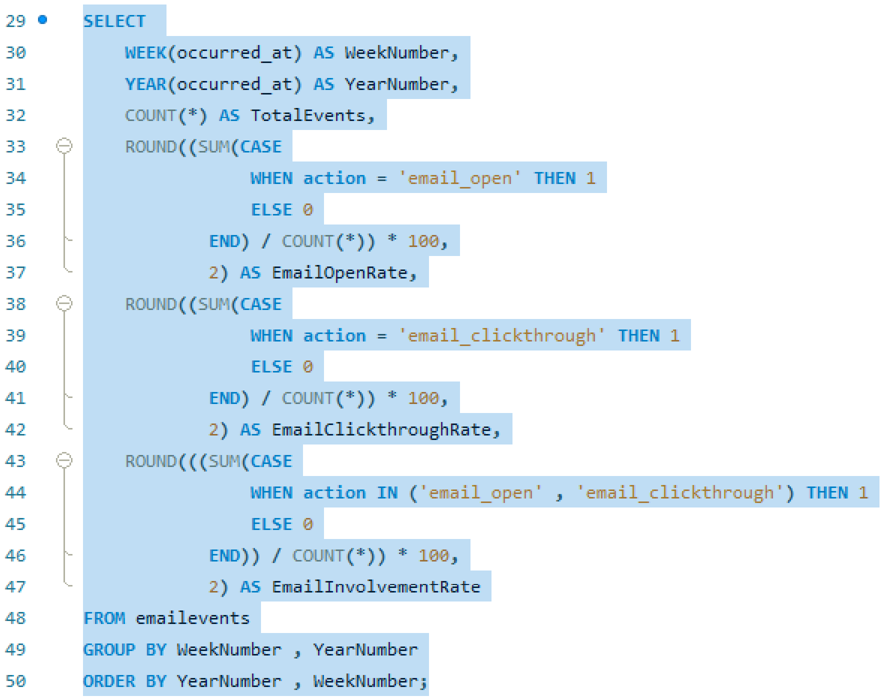
- ✓ Analysing engagement counts by signup flows can help prioritize efforts to optimize user experiences.

5. EMAIL ENGAGEMENT ANALYSIS

Objective: Analyse how users are engaging with the email service.

My Task: Write an SQL query to calculate the email engagement metrics.

Approach *****Screenshots Attached**



```sql
SELECT
    WEEK(occurred_at) AS WeekNumber,
    YEAR(occurred_at) AS YearNumber,
    COUNT(*) AS TotalEvents,
    ROUND((SUM(CASE
                WHEN action = 'email_open' THEN 1
                ELSE 0
            END) / COUNT(*)) * 100,
            2) AS EmailOpenRate,
    ROUND((SUM(CASE
                WHEN action = 'email_clickthrough' THEN 1
                ELSE 0
            END) / COUNT(*)) * 100,
            2) AS EmailClickthroughRate,
    ROUND(((SUM(CASE
                WHEN action IN ('email_open' , 'email_clickthrough') THEN 1
                ELSE 0
            END)) / COUNT(*)) * 100,
            2) AS EmailInvolvementRate
FROM emailevents
GROUP BY WeekNumber , YearNumber
ORDER BY YearNumber , WeekNumber;
```

This query calculates email engagement metrics on a weekly basis from the 'emailevents' table.

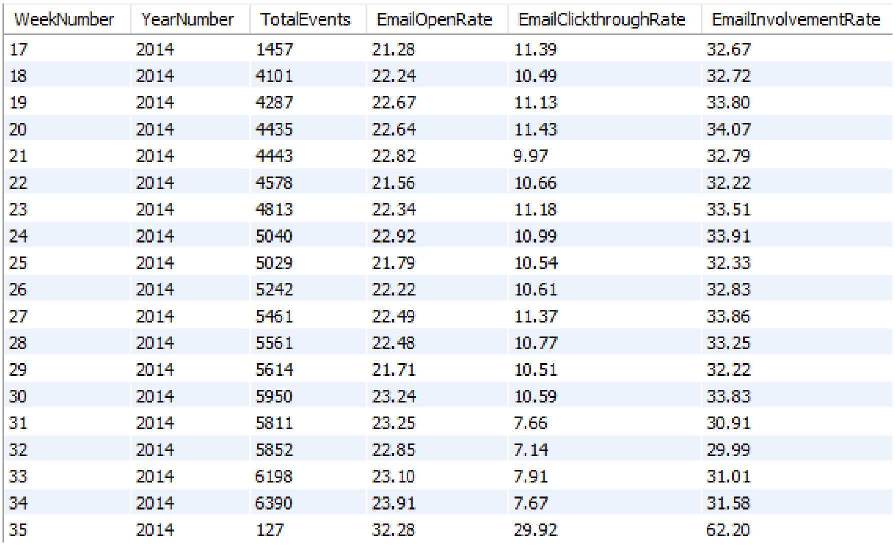
Email Open Rate: It Calculates the email open rate by dividing the count of 'email_open' events by the total events and multiplying by 100 to get the percentage. It rounds the result to two decimal places.

It Calculates the email clickthrough rate using a similar approach as the email open rate.

Email Involvement Rate: It Calculates the email involvement rate by summing up the counts of both 'email_open' and 'email_clickthrough' events and dividing by the total events. It also rounds the result to two decimal places.

Then, groups the data by week and year and order the results by year and week number. This query monitors email campaign performance over time.

RESULT



WeekNumber	YearNumber	TotalEvents	EmailOpenRate	EmailClickthroughRate	EmailInvolvementRate
17	2014	1457	21.28	11.39	32.67
18	2014	4101	22.24	10.49	32.72
19	2014	4287	22.67	11.13	33.80
20	2014	4435	22.64	11.43	34.07
21	2014	4443	22.82	9.97	32.79
22	2014	4578	21.56	10.66	32.22
23	2014	4813	22.34	11.18	33.51
24	2014	5040	22.92	10.99	33.91
25	2014	5029	21.79	10.54	32.33
26	2014	5242	22.22	10.61	32.83
27	2014	5461	22.49	11.37	33.86
28	2014	5561	22.48	10.77	33.25
29	2014	5614	21.71	10.51	32.22
30	2014	5950	23.24	10.59	33.83
31	2014	5811	23.25	7.66	30.91
32	2014	5852	22.85	7.14	29.99
33	2014	6198	23.10	7.91	31.01
34	2014	6390	23.91	7.67	31.58
35	2014	127	32.28	29.92	62.20

Total events gradually increased from week 17 to week 34, indicating a consistent volume of email activities.

Email open rate fluctuated between approximately 21% and 24%, with slight variations over time.

Email clickthrough rate remained relatively stable, ranging from around 7% to 12%.

Email involvement rate showed minor fluctuations, hovering around 30% to 34% for most weeks.

INSIGHTS

- ✓ Focus on crafting compelling and relevant subject lines and pre-header text to increase email open rates. A/B testing can help identify the most effective messaging.

- ✓ Ensure that email content is engaging, informative, and tailored to the interests and preferences of the target audience.

- ✓ Use personalized recommendations, visually appealing designs, and clear calls-to-action to improve clickthrough rates.

- ✓ Implement audience segmentation based on demographics, behaviour, and preferences to deliver more targeted and relevant email campaigns.

- ✓ Tailoring content to specific audience segments can enhance engagement and drive higher conversion rates.

- ✓ Evaluate the frequency and timing of email sends to avoid overwhelming subscribers and optimize engagement.

- ✓ Continuously monitor email engagement metrics and analyse performance data to identify trends, patterns, and areas for improvement.

- ✓ By implementing these steps, the company can enhance the effectiveness of its email marketing campaigns, improve engagement rates, and ultimately drive better results and ROI.

TheEnd