**Project Description :**

This project involved using SQL to analyze operational data and investigate metric spikes. The goal was to generate insights and answer complex questions related to the organization's operations. I used a range of SQL techniques to answer complex questions, such as subqueries, joins, and aggregations. For example, I used subqueries to compare the performance of different teams or regions, and joins to combine data from multiple sources. I also calculated metrics such as rolling averages and percentiles to gain a deeper understanding of the data.

**Approach :**

* Database Creation: Use SQL to create a database that can store the data and support the queries you need to perform the analysis.
* Data Loading: Load the data into the database.
* Data Analysis: Use SQL queries to perform the analysis and retrieve the insights you want to extract.

**Tech-Stack Used :**

1. MySQL

**Insights :**

Below I have provided SQL queries and their respective outputs and with insights/outcomes for each of the questions.

**Case Study 1 (Job Data):**

A - **Number of jobs reviewed:**

SELECT DATE\_FORMAT(ds, '%Y-%m-%d %H:00:00') AS hour,

(sum(time\_spent)/count(\*)) as Time\_spent\_per\_hour\_per\_job

FROM jobs

WHERE MONTH(ds) = 11 AND YEAR(ds) = 2020

GROUP BY DATE\_FORMAT(ds, '%Y-%m-%d %H:00:00');

|  |  |
| --- | --- |
| hour | Time\_spent\_per\_hour\_per\_job |
| 30-11-2020 00:00 | 20 |
| 29-11-2020 00:00 | 20 |
| 28-11-2020 00:00 | 16.5 |
| 27-11-2020 00:00 | 104 |
| 26-11-2020 00:00 | 56 |
| 25-11-2020 00:00 | 45 |

B - Throughput:

SELECT SUM(time\_spent) / TIMESTAMPDIFF(SECOND, MIN(ds), MAX(ds)) AS throughput

FROM jobs;

|  |
| --- |
| throughput |
| 0.0007 |

We cannot find 7 day moving average because the data provided contains only 5 days of data. C - Percentage share of each language:

select language, (count(\*)/(select count(\*) from jobs))\*100 as perc

from jobs

group by language;

|  |  |
| --- | --- |
| language | perc |
| English | 12.5 |
| Arabic | 12.5 |
| Persian | 37.5 |
| Hindi | 12.5 |
| French | 12.5 |
| Italian | 12.5 |

Persian is used most number of times as compared to other languages

D - Duplicate rows:

We can see that there are no duplicate rows present in the table although if we check for every column separately we can find that some values are repeating. To find those values we can use the following syntax : -

select ds

from jobs

GROUP BY ds

HAVING COUNT(\*) > 1;

Using above query we have extracted all the dates that have duplicate values.

|  |
| --- |
| ds |
| 30-11-2020 00:00 |
| 28-11-2020 00:00 |

**Case Study 2 (Investigating metric spike):**

A - User Engagement:

SELECT

(SELECT COUNT(DISTINCT(user\_id)) AS active\_users

FROM users

WHERE state = 'active') / (SELECT COUNT(DISTINCT(user\_id)) AS active\_users

FROM users) \* 100 as user\_engagement\_rate

FROM users

By running the above query we get to know that the user engagement rate is just at **49.2%.**

B - User Growth:

SELECT

new\_users.device,

avg(new\_users.new\_users / first\_month.total\_users\*100) AS growth\_rate

FROM (

SELECT

device,

month(occurred\_at) AS signup\_month,

COUNT(DISTINCT user\_id) AS new\_users

FROM events

GROUP BY device, signup\_month

) AS new\_users

JOIN (

SELECT

device,

COUNT(DISTINCT user\_id) AS total\_users

FROM events

GROUP BY device

) AS first\_month

ON new\_users.device = first\_month.device

group by new\_users.device;

|  |  |
| --- | --- |
| **device** | **growth\_rate** |
| acer aspire desktop | 36.49 |
| acer aspire notebook | 35.58 |
| amazon fire phone | 32.2 |
| asus chromebook | 34.89 |
| dell inspiron desktop | 35.56 |
| dell inspiron notebook | 36.38 |
| hp pavilion desktop | 35.3 |
| htc one | 33.37 |
| ipad air | 33.14 |
| ipad mini | 32.34 |
| iphone 4s | 34.7 |
| iphone 5 | 35.11 |
| iphone 5s | 33.99 |
| device | **growth\_rate** |
| kindle fire | 32.64 |
| lenovo thinkpad | 36.15 |
| mac mini | 39.41 |
| macbook air | 35.63 |
| macbook pro | 35.86 |
| nexus 10 | 33 |
| nexus 5 | 34.28 |
| nexus 7 | 32.81 |
| nokia lumia 635 | 36.25 |
| samsumg galaxy tablet | 31.85 |
| samsung galaxy note | 33.67 |
| samsung galaxy s4 | 33.64 |
| windows surface | 32.35 |

Here’s a list of all the devices and their growth rate with respect to the first month.

C - Weekly Retention:

SELECT

WEEK(occurred\_at) AS week,

COUNT(DISTINCT user\_id) AS signups,

COUNT(DISTINCT CASE WHEN event\_type = 'engagement' THEN user\_id END) AS engagements,

COUNT(DISTINCT CASE WHEN event\_type = 'engagement' THEN user\_id END) / COUNT(DISTINCT user\_id) \*100 AS retention\_rate

FROM events

GROUP BY WEEK(occurred\_at)

ORDER BY week;

|  |  |  |  |
| --- | --- | --- | --- |
| **week** | **signups** | **engagements** | **retention\_rate** |
| 17 | 740 | 663 | 89.5946 |
| 18 | 1260 | 1068 | 84.7619 |
| 19 | 1287 | 1113 | 86.4802 |
| 20 | 1351 | 1154 | 85.4182 |
| 21 | 1299 | 1121 | 86.2972 |
| 22 | 1381 | 1186 | 85.8798 |
| 23 | 1446 | 1232 | 85.2006 |
| 24 | 1471 | 1275 | 86.6757 |
| 25 | 1459 | 1264 | 86.6347 |
| 26 | 1509 | 1302 | 86.2823 |
| 27 | 1573 | 1372 | 87.2219 |
| 28 | 1577 | 1365 | 86.5568 |
| 29 | 1607 | 1376 | 85.6254 |
| 30 | 1706 | 1467 | 85.9906 |
| 31 | 1514 | 1299 | 85.7992 |
| 32 | 1454 | 1225 | 84.2503 |
| 33 | 1438 | 1225 | 85.1878 |
| 34 | 1443 | 1204 | 83.4373 |
| 35 | 118 | 104 | 88.1356 |

The above table shows the retention rate of users in every week.

D - Weekly Engagement:

SELECT

WEEK(occurred\_at) AS week,

COUNT(DISTINCT CASE WHEN event\_type = 'engagement' THEN user\_id END) / COUNT(DISTINCT user\_id)/(select count(distinct(device)) from events) \* 100 AS engagement\_rate\_per\_device

FROM events

GROUP BY WEEK(occurred\_at)

ORDER BY week;

|  |  |
| --- | --- |
| **week** | **engagement\_rate\_per\_device** |
| 17 | 3.44594594 |
| 18 | 3.26007326 |
| 19 | 3.32616102 |
| 20 | 3.28531572 |
| 21 | 3.31912122 |
| 22 | 3.30306912 |
| 23 | 3.27694435 |
| 24 | 3.33368195 |
| 25 | 3.33210312 |
| 26 | 3.31855023 |
| 27 | 3.35468727 |
| 28 | 3.3291059 |
| 29 | 3.29328419 |
| 30 | 3.30733159 |
| 31 | 3.29996951 |
| 32 | 3.24039784 |
| 33 | 3.27645233 |
| 34 | 3.20912628 |
| 35 | 3.38983051 |

E - Email Engagement:

SELECT

engagements,

total\_users,

engagements/total\_users\*100 AS engagement\_rate

from ( SELECT

count(distinct(user\_id)) AS total\_users,

COUNT(DISTINCT

CASE

WHEN action = 'email\_open' THEN user\_id

WHEN action = 'email\_clickthrough' THEN user\_id END) AS engagements

FROM email\_events) as counts

|  |  |  |
| --- | --- | --- |
| **engagements** | **total\_users** | **engagement\_rate** |
| 5927 | 6179 | 95.9217 |

**Result :**

Extracted many useful insights that could help in making data driven decisions for the business.

Used MySQL, had to study few advanced topics like Window functions,CTEs and datetime functions.