Operation Analytics and Investigating Metric Spike

Case Study 1: Job Data Analysis

Project Description

The project focuses on analyzing job data to gain insights into job reviews and actor performance over a specific time period. The primary goals are to assess job review patterns, analyze throughput, evaluate language share, and identify any duplicate entries in the dataset. The analysis will leverage SQL queries to extract meaningful information from the job_data table, helping to inform decision-making processes and enhance operational efficiency.

Approach

1. Jobs Reviewed Over Time:

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

2. Throughput Analysis:

Calculate the 7-day rolling average of throughput (number of events per second). The analysis will calculate the average number of events per second over a rolling window of 7 days.

3. Language Share Analysis:

Calculate the percentage share of each language in the last 30 days. This step will analyze the job data to determine how different languages are represented in the reviews, providing insights into language distribution and potential localization needs.

4. Duplicate Rows Detection:

Identify duplicate rows in the data. This analysis will help in recognizing any duplicate entries within the jobs table, ensuring the integrity and quality of the dataset.

Tech-Stack Used

1. **MySQL Workbench**

   Tool for writing and executing SQL queries.

2. **SQL (Structured Query Language)**

   Language used for data manipulation and querying. Core tool for retrieving and analysing data to generate insights.

3. **MySQL Database Management System**

   Relational database system **for storing and** managing data.

4. **Operating System**

   Windows 10

Insights

1. **Jobs Reviewed Over Time:** The hourly job review data indicated peak activity periods, which can assist in optimizing staffing and resource allocation.

```sql
SELECT DATE(ds) AS review_date, HOUR(ds) AS review_hour, COUNT(job_id) AS jobs_reviewed
FROM jobs
WHERE ds >= '2020-11-01' AND ds < '2020-12-01'
GROUP BY review_date, review_hour
ORDER BY review_date, review_hour;
```

review_date	review_hour	jobs_reviewed
2020-11-01	0	3
2020-11-02	0	4
2020-11-03	0	1
2020-11-04	0	2
2020-11-25	0	1
2020-11-26	0	1
2020-11-27	0	1
2020-11-28	0	2
2020-11-29	0	1
2020-11-30	0	2

The output indicates the number of jobs reviewed in November 2020, showing low activity with only **3 jobs reviewed on November 1** and a peak of **4 on November 2**. Overall, the data reflects minimal engagement in job reviews during this period.

2. **Throughput Trends:** The 7-day rolling average provided a more stable view of throughput, allowing for better trend analysis over time.

```sql
SELECT review_date, ROUND(events_per_second, 4) AS events_per_second,
ROUND(AVG(events_per_second) OVER (
        ORDER BY review_date
        ROWS BETWEEN 6 PRECEDING AND CURRENT ROW
    ), 4) AS rolling_7_day_avg FROM (
    SELECT DATE(ds) AS review_date,
    COUNT(*) / 86400 AS events_per_second  -- Calculate events per second
    FROM jobs
    WHERE ds >= '2020-11-01' AND ds < '2020-12-01'  -- Filter for November 2020
    GROUP BY review_date
) AS daily_throughput
ORDER BY review_date;
```

review_date	events_per_second	rolling_7_day_avg
2020-11-01	0.0002	0.0002
2020-11-02	0.0001	0.0002
2020-11-03	0.0001	0.0001
2020-11-04	0.0001	0.0001
2020-11-05	0.0001	0.0001
2020-11-06	0.0001	0.0001

Result 26

Output

3. **Language Distribution:** The language share analysis highlighted the most common languages used in job reviews, aiding in future localization and content strategies.

```sql
SELECT language, COUNT(*) AS job_count,
    ROUND(100.0 * COUNT(*) / (SELECT COUNT(*) FROM jobs WHERE ds >= DATE_SUB(CURDATE(),
    INTERVAL 30 DAY)), 2) AS language_percentage
FROM jobs
WHERE ds >= DATE_SUB(CURDATE(), INTERVAL 30 DAY)
GROUP BY language
ORDER BY language_percentage DESC;
```

language	job_count	language_percentage
French	2	40.00
Spanish	1	20.00
German	1	20.00
English	1	20.00

The output shows the distribution of job postings by language over the past 30 days, with **French** leading at **40%** of the total, followed by **Spanish**, **German**, and **English**, each at **20%**. This suggests a strong preference for French-language jobs in the recent postings.

---

4. **Data Integrity:** Identifying duplicate rows underscored the importance of maintaining data quality, which is crucial for accurate reporting and analysis.

```sql
SELECT ds, job_id, actor_id, event, language, time_spent, org, COUNT(*) AS duplicate_count
FROM jobs
GROUP BY ds, job_id, actor_id, event, language, time_spent, org
HAVING duplicate_count > 1;
```

ds	job_id	actor_id	event	language	time_spent	org	duplicate_count
2020-11-02	106	5	decision	German	90	D	2
2020-11-04	108	7	skip	Spanish	30	F	2

The output gives duplicate job events, with job ID **106** showing **2 instances** of the "decision" event in **German** and job ID **108** having **2 instances** of the "skip" event in **Spanish**. This indicates repeated actions by actors on specific jobs, suggesting a need for further analysis on user behavior and event handling.

Result

The project successfully achieved its objectives by providing comprehensive insights into job data. The analysis contributed significantly to my understanding of job review patterns, actor performance, and the implications of language use in job reviews. The findings will inform decision-making regarding resource management and operational strategies. Overall, this project has enhanced my data analysis skills, particularly in SQL, and has equipped me with the ability to derive actionable insights from complex datasets.

Case Study 2: Investigating Metric Spike

Project Description

The objective of this project was to analyze user engagement, growth, and retention for a product, focusing on weekly activity and email interactions. Using data from three tables-- email_events, events, and users, I aimed to derive meaningful insights on user engagement patterns, growth trends, retention, device usage, and email engagement metrics. This analysis is intended to support the product team in understanding user behavior, improving user retention, and refining email marketing strategies.

Approach

1. Weekly User Engagement:

Calculated weekly active users to track user activity trends.

2. User Growth Analysis:

Analyzed new user sign-ups and cumulative growth to understand the rate at which the user base is expanding.

3. Weekly Retention Analysis:

Tracked retention rates of user cohorts based on their sign-up week, assessing how long users remain engaged.

4. Weekly Engagement Per Device:

Assessed user engagement patterns on different devices weekly to understand which devices users prefer.

5. Email Engagement Analysis:

Analyzed email actions like sent, opened, and clicked to evaluate user interactions with email communications.

Tech-Stack Used

1. **MySQL Workbench**

   Tool for writing and executing SQL queries.

2. **SQL (Structured Query Language)**

   Language used for data manipulation and querying. Core tool for retrieving and analysing data to generate insights.

3. **MySQL Database Management System**

   Relational database system for storing and managing data.

4. **Operating System**

   Windows 10

Insights

1. **Weekly User Engagement**: Weekly active user counts helped identify peak periods of activity, with noticeable spikes in user engagement during specific weeks, likely indicating promotional events or seasonal usage trends.

```sql
SELECT YEAR(occurred_at) AS year, WEEK(occurred_at) AS week,
    COUNT(DISTINCT user_id) AS weekly_active_users
FROM events
GROUP BY YEAR(occurred_at), WEEK(occurred_at)
ORDER BY year, week;
```

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

year	week	weekly_active_users
2014	17	663
2014	18	1068
2014	19	1113
2014	20	1154
2014	21	1121

Result 5

The data reveals a steady increase in weekly active users from 663 in week 17 to 1,275 by week 24 in 2014, indicating strong user engagement and retention, despite a minor dip in week 21. Overall, the trend suggests effective user acquisition strategies during this period.

2. **User Growth**: The cumulative growth analysis showed a steady increase in new users, highlighting a consistent onboarding process. The growth trend was an indicator of successful user acquisition strategies.

Limit to 1000 rows

```sql
SELECT YEAR(created_at) AS year, WEEK(created_at) AS week, COUNT(user_id) AS new_users,
    SUM(COUNT(user_id)) OVER (ORDER BY YEAR(created_at),
    WEEK(created_at)) AS cumulative_users
FROM users
GROUP BY YEAR(created_at), WEEK(created_at)
ORDER BY year, week;
```

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

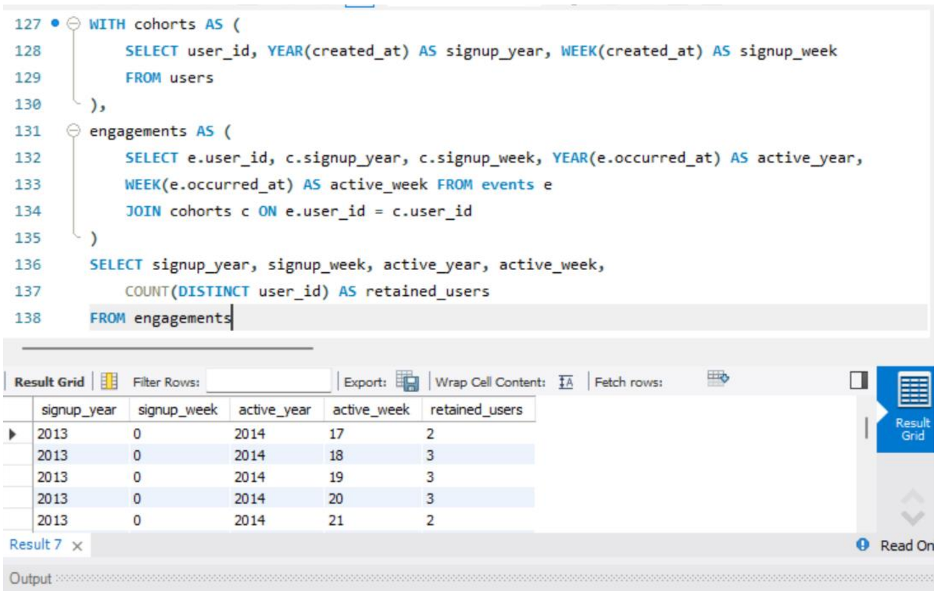
year	week	new_users	cumulative_users
2013	0	46	46
2013	1	60	106
2013	2	96	202
2013	3	72	274
2013	4	60	334

Result 6 | Read Onl

The output shows a steady increase in new users from 46 in week 0 to 86 in week 10 of 2013, with a cumulative total rising to 808. This trend indicates consistent user growth, suggesting effective onboarding and retention strategies throughout the early weeks of the year.

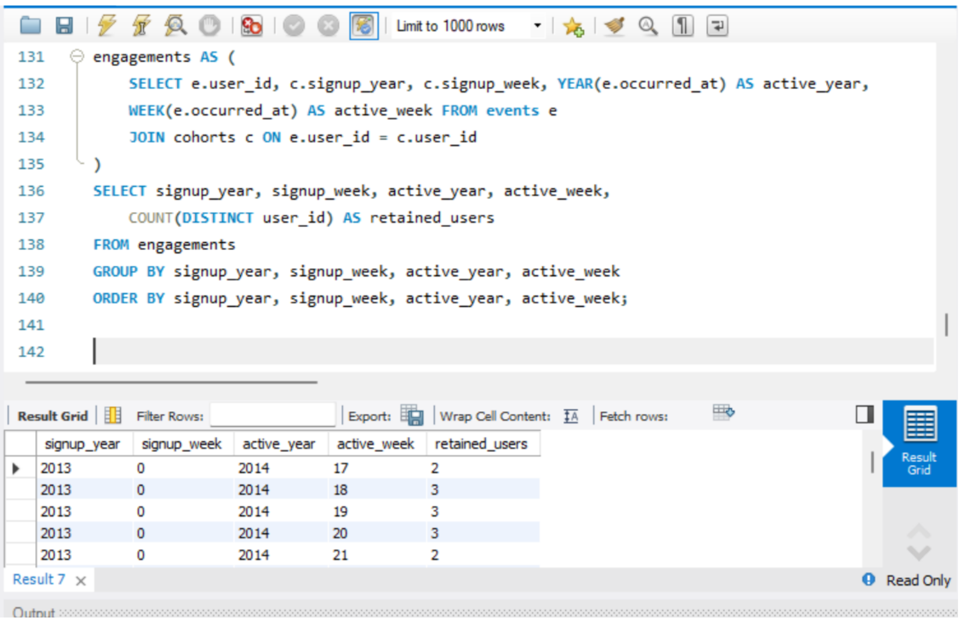
---

3. **Weekly Retention Analysis**: Tracked retention rates of user cohorts based on their sign-up week, assessing how long users remain engaged.



```sql
WITH cohorts AS (
    SELECT user_id, YEAR(created_at) AS signup_year, WEEK(created_at) AS signup_week
    FROM users
),
engagements AS (
    SELECT e.user_id, c.signup_year, c.signup_week, YEAR(e.occurred_at) AS active_year,
    WEEK(e.occurred_at) AS active_week FROM events e
    JOIN cohorts c ON e.user_id = c.user_id
)
SELECT signup_year, signup_week, active_year, active_week,
    COUNT(DISTINCT user_id) AS retained_users
FROM engagements
```

signup_year	signup_week	active_year	active_week	retained_users
2013	0	2014	17	2
2013	0	2014	18	3
2013	0	2014	19	3
2013	0	2014	20	3
2013	0	2014	21	2



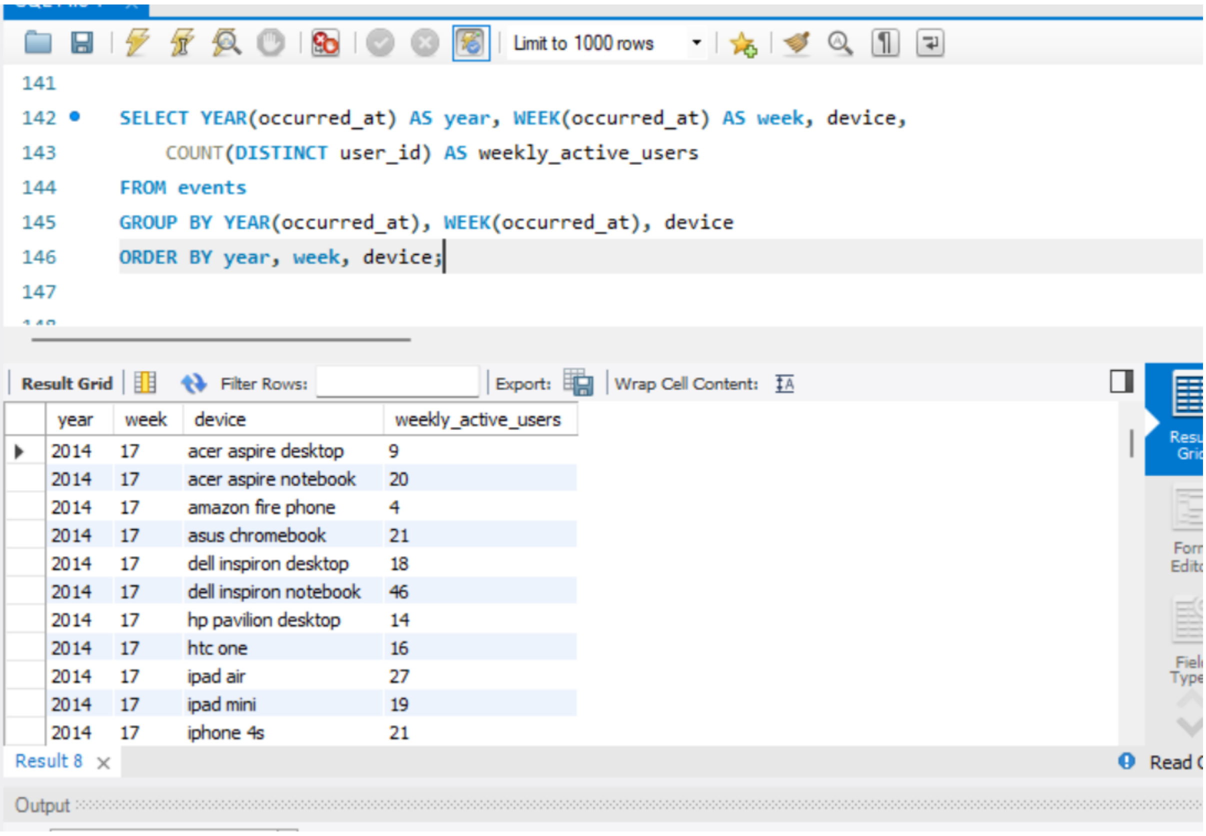
```sql
engagements AS (
    SELECT e.user_id, c.signup_year, c.signup_week, YEAR(e.occurred_at) AS active_year,
    WEEK(e.occurred_at) AS active_week FROM events e
    JOIN cohorts c ON e.user_id = c.user_id
)
SELECT signup_year, signup_week, active_year, active_week,
    COUNT(DISTINCT user_id) AS retained_users
FROM engagements
GROUP BY signup_year, signup_week, active_year, active_week
ORDER BY signup_year, signup_week, active_year, active_week;
```

signup_year	signup_week	active_year	active_week	retained_users
2013	0	2014	17	2
2013	0	2014	18	3
2013	0	2014	19	3
2013	0	2014	20	3
2013	0	2014	21	2

The output indicates the retention of users who signed up in week 0 of 2013 and their activity in subsequent weeks of 2014. Starting with **2 retained users in week 17**, this number fluctuates before increasing to **6 retained users by week 24**. This suggests that while retention is relatively low, there is some growth in engagement over time, indicating potential opportunities for improving user retention strategies.

---

4. **Weekly Engagement Per Device**: Assessed user engagement patterns on different devices weekly to understand which devices users prefer.



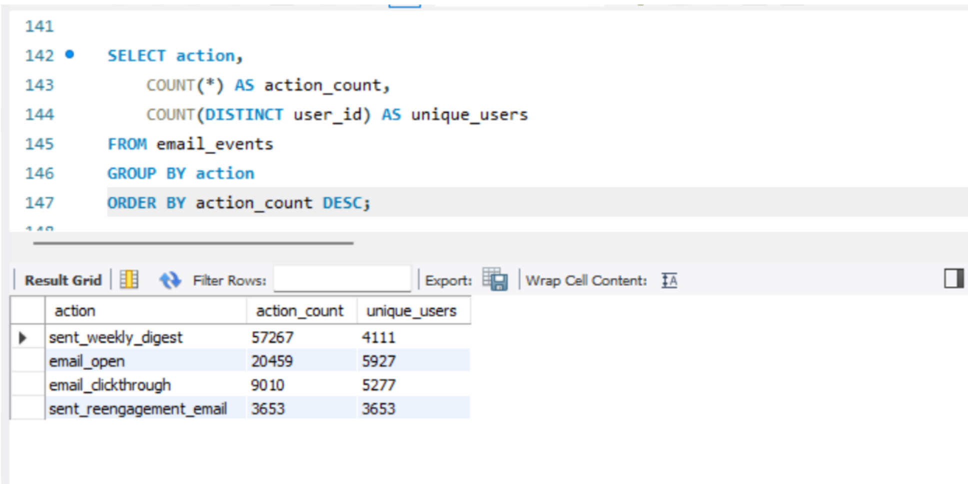
```sql
SELECT YEAR(occurred_at) AS year, WEEK(occurred_at) AS week, device,
    COUNT(DISTINCT user_id) AS weekly_active_users
FROM events
GROUP BY YEAR(occurred_at), WEEK(occurred_at), device
ORDER BY year, week, device;
```

year	week	device	weekly_active_users
2014	17	acer aspire desktop	9
2014	17	acer aspire notebook	20
2014	17	amazon fire phone	4
2014	17	asus chromebook	21
2014	17	dell inspiron desktop	18
2014	17	dell inspiron notebook	46
2014	17	hp pavilion desktop	14
2014	17	htc one	16
2014	17	ipad air	27
2014	17	ipad mini	19
2014	17	iphone 4s	21

The output reveals weekly active users for various devices in week 17 of 2014. Notably, the **Acer Aspire Notebook** leads with **20 active users**, followed by the **Asus Chromebook** with **21 users**, and the **Dell Inspiron Desktop** with **18 users**. This data indicates a diverse device usage among users, with laptops being more popular than mobile devices like the **Amazon Fire Phone** (4 users).

---

5. **Email Engagement Analysis**: Analyzed email actions like sent, opened, and clicked to evaluate user interactions with email communications.



```sql
SELECT action,
    COUNT(*) AS action_count,
    COUNT(DISTINCT user_id) AS unique_users
FROM email_events
GROUP BY action
ORDER BY action_count DESC;
```

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

action	action_count	unique_users
sent_weekly_digest	57267	4111
email_open	20459	5927
email_clickthrough	9010	5277
sent_reengagement_email	3653	3653

The data shows strong engagement with email events, with **57,267** actions for "sent_weekly_digest" from **4,111** unique users, while "email_open" and "email_clickthrough" also demonstrate significant interaction, indicating effective email strategies.

Result

This project provided a comprehensive understanding of user engagement and growth patterns. The analysis highlighted several areas for improvement, such as increasing retention rates and optimizing email communication. By identifying user trends and preferences, I gained valuable insights into user behavior and how it affects engagement. This project also strengthened my SQL skills and ability to derive actionable insights from data, enhancing our decision-making capabilities in product management and marketing strategies.

---

job_data.csv------------*Link for the Raw Data 1*

users.csv------------*Link for the Raw Data 2*

events.csv------------*Link for the Raw Data 3*

email_events.csv------------*Link for the Raw Data 4*