Case Study 1: Job Data Analysis

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

```
SELECT
   ds AS review_date,
   HOUR(time_spent) AS review_hour,
   COUNT(job_id) AS jobs_reviewed
FROM
   JobEvents
WHERE
   ds >= '2020-11-01'
   AND ds <= '2020-11-30'
GROUP BY
   review_date,
   review_hour
ORDER BY
   review_date,
   review_hour;
```

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?

```
WITH DailyThroughput AS (
   SELECT ds AS review_date, COUNT(job_id) AS daily_throughput
   FROM JobEvents
   WHERE
      ds >= '2020-11-01'
      AND ds <= '2020-11-30'
   GROUP BY ds
)
SELECT
   review_date,
   daily_throughput,
   AVG(daily_throughput) OVER ( ORDER BY review_date
      ROWS BETWEEN 6 PRECEDING AND CURRENT ROW ) AS rolling_avg_throughput
FROM
   DailyThroughput
ORDER BY
   review_date;
```

Preference: I prefer using the **7-day rolling average** for throughput because it provides a more stable and clear view of trends over time. While daily metrics are useful for immediate, short-term analysis, they can be noisy and less informative for understanding overall performance and trends. The rolling average mitigates daily volatility and offers a better sense of the underlying patterns in throughput, which is particularly valuable for strategic planning and long-term decision-making.

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

```
WITH Last30DaysData AS (
   SELECT
      language,
      COUNT(job_id) AS job_count
   FROM
      JobEvents
   WHERE
      ds >= DATE_SUB(CURDATE(), INTERVAL 30 DAY)
   GROUP BY
      language
),
TotalJobCount AS (
   SELECT
      SUM(job_count) AS total_jobs
   FROM
      Last30DaysData
)
SELECT
   l.language,
   l.job_count,
   ROUND((l.job_count / t.total_jobs) * 100, 2) AS percentage_share
FROM
   Last30DaysData l
   CROSS JOIN TotalJobCount t
ORDER BY
   percentage_share DESC;
```

Display duplicate rows from the job_data table

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

Case Study 2: Investigating Metric Spike

Weekly User Engagement: calculate the weekly user engagement

```
WITH UserEvents AS (
   SELECT
      user_id,
      DATE_TRUNC('week', occurred_at) AS week,
      COUNT(*) AS event_count
   FROM events
   GROUP BY
      user_id,
      week
),
EmailEvents AS (
   SELECT
      user_id,
      DATE_TRUNC('week', occurred_at) AS week,
      COUNT(*) AS email_event_count
   FROM email_events
   GROUP BY
      user_id,
      week
),
WeeklyEngagement AS (
   SELECT
      ue.user_id,
      ue.week,
      ue.event_count + COALESCE(ee.email_event_count, 0) AS total_events
   FROM
      UserEvents ue
   LEFT JOIN
      EmailEvents ee ON ue.user_id = ee.user_id AND ue.week = ee.week
)
SELECT
   we.user_id,
   we.week,
   we.total_events
FROM
   WeeklyEngagement we
ORDER BY
   we.week,
   we.user_id;
```

User Growth Analysis: calculate the user growth for the product

```
WITH MonthlyNewUsers AS (
   SELECT
      DATE_TRUNC('month', created_at) AS month,
      COUNT(*) AS new_users
   FROM
      users
   GROUP BY
      month
),
MonthlyGrowth AS (
   SELECT
      month,
      new_users,
      LAG(new_users) OVER (ORDER BY month) AS prev_month_users,
      CASE
         WHEN LAG(new_users) OVER (ORDER BY month) IS NULL THEN 0
         ELSE new_users - LAG(new_users) OVER (ORDER BY month)
      END AS user_growth,
      CASE
         WHEN LAG(new_users) OVER (ORDER BY month) IS NULL THEN 0
         ELSE (new_users - LAG(new_users) OVER (ORDER BY month)) / LAG(new_users)
OVER (ORDER BY month)::FLOAT * 100
      END AS growth_percentage
   FROM
      MonthlyNewUsers
)
SELECT
   month,
   new_users,
   prev_month_users,
   user_growth,
   growth_percentage
FROM
   MonthlyGrowth
ORDER BY
   month;
```

Weekly Retention Analysis: calculate the weekly retention of users based on their sign-up cohort.

```
WITH UserCohorts AS (
    SELECT
        user_id,
        DATE_TRUNC('week', created_at) AS sign_up_week
    FROM
        users
),
UserActivity AS (
    SELECT
        user_id,
        DATE_TRUNC('week', occurred_at) AS activity_week
    FROM
        events
),
CohortActivity AS (
    SELECT
        uc.user_id,
        uc.sign_up_week,
        ua.activity_week,
        EXTRACT(WEEK FROM ua.activity_week - uc.sign_up_week) AS week_number
    FROM
        UserCohorts uc
    JOIN
        UserActivity ua ON uc.user_id = ua.user_id
),
WeeklyRetention AS (
    SELECT
        sign_up_week,
        week_number,
        COUNT(DISTINCT user_id) AS retained_users
    FROM
        CohortActivity
    GROUP BY
        sign_up_week,
        week_number
)
SELECT
    sign_up_week,
    week_number,
    retained_users,
    LAG(retained_users) OVER (PARTITION BY sign_up_week ORDER BY week_number) AS
previous_week_retained,
    CASE
        WHEN LAG(retained_users) OVER (PARTITION BY sign_up_week ORDER BY
week_number) IS NULL THEN NULL
        ELSE (retained_users / LAG(retained_users) OVER (PARTITION BY sign_up_week
ORDER BY week_number)::FLOAT) * 100
    END AS retention_rate
FROM
    WeeklyRetention
ORDER BY
    sign_up_week,
    week_number;
```

Weekly Engagement Per Device: calculate the weekly engagement per device

```
WITH WeeklyDeviceEngagement AS (
    SELECT
        DATE_TRUNC('week', occurred_at) AS week,
        device,
        COUNT(*) AS event_count
    FROM
        events
    GROUP BY
        week,
        device
)
SELECT
    week,
    device,
    event_count
FROM
    WeeklyDeviceEngagement
ORDER BY
    week,
    device;
```

Email Engagement Analysis: calculate the email engagement metrics

```
WITH EmailEngagement AS (
    SELECT
        DATE_TRUNC('week', occurred_at) AS week,
        action,
        COUNT(*) AS event_count
    FROM email_events
    GROUP BY
        week,
        action
),
EmailMetrics AS (
    SELECT
        week,
        SUM(CASE WHEN action = 'sent_weekly_digest' THEN event_count ELSE 0 END) AS
emails_sent,
        SUM(CASE WHEN action = 'open_email' THEN event_count ELSE 0 END) AS
emails_opened,
        SUM(CASE WHEN action = 'click_link' THEN event_count ELSE 0 END) AS
emails_clicked
    FROM EmailEngagement
    GROUP BY
        week
)
SELECT
    week,
    emails_sent,
    emails_opened,
    emails_clicked,
    CASE
        WHEN emails_sent = 0 THEN 0
        ELSE (emails_opened::FLOAT / emails_sent) * 100
    END AS open_rate,
    CASE
        WHEN emails_sent = 0 THEN 0
        ELSE (emails_clicked::FLOAT / emails_sent) * 100
    END AS click_through_rate,
    CASE
        WHEN emails_opened = 0 THEN 0
        ELSE (emails_clicked::FLOAT / emails_opened) * 100
    END AS click_to_open_rate
FROM EmailMetrics
ORDER BY week;
```

Project Description : This project aims to analyze andl provide an understanding of user behavior, engagement patterns, and the effectiveness of email campaigns. This information is crucial for optimizing user retention strategies, enhancing user experience, and improving email marketing efforts.

Approach

1. **Data Preparation**:
    - Load the data from the CSV files (`users`, `events`, `email_events`) into a SQL database.
    - Ensure data consistency and handle missing or erroneous entries.

Tech-Stack Used

- **MySQL Workbench**: Employed for database management, query execution, and result visualization.

Insights

- **Weekly Engagement**: Identified patterns in user activity, showing peak engagement periods. This can guide marketing and user engagement strategies.
- **User Growth**: Monthly analysis revealed trends in user acquisition, highlighting periods of significant growth and potential factors contributing to these trends.
- **User Retention**: Cohort analysis uncovered retention rates, showing how user engagement evolves over time from their sign-up date.
- **Email Engagement**: Metrics such as open rates, click-through rates, and click-to-open rates provided insights into the effectiveness of email campaigns, indicating areas for improvement in email content and targeting.

Result

The project achieved a comprehensive analysis of user engagement, growth, retention, and email engagement metrics. Key achievements include: