SQL QUERIES FOR THE ANALYSIS

-- CASE STUDY 1: A) Job Data Analysis

-- 1. Jobs Reviewed Over Time:

```
SELECT COUNT(DISTINCT job_id)/(30*24) AS per_day_jobs
FROM job_data;
```

-- 2. Throughput Analysis:

```
SELECT
   ds,
   tot_events,
   AVG(tot_events) OVER (ORDER BY ds ROWS BETWEEN 6
PRECEDING AND CURRENT ROW) AS 7_day_rolling_average
FROM (
   SELECT
      ds,
      COUNT(DISTINCT event) AS tot_events
   FROM job_data
   GROUP BY ds
) sub
ORDER BY ds;
```

-- 3. Language Share Analysis:

```
WITH LanguageCounts AS (
   SELECT
      language,
      COUNT(language) AS total_language,
      SUM(COUNT(*)) OVER () AS total_count
   FROM job_data
   GROUP BY language
)
SELECT
   language,
   total_language,
   (total_language * 100.0) / total_count AS percentage
FROM LanguageCounts
ORDER BY language;
```

-- 4. Duplicate Rows Detection:

```
WITH cte AS (
   SELECT *,
         ROW_NUMBER() OVER (PARTITION BY job_id ORDER BY
(SELECT 0)) AS row_numb
   FROM job_data
)
SELECT *
FROM cte
WHERE row_numb > 1;
```

-- CASE STUDY 2: B) Investigating Metric Spike

-- 1. Weekly User Engagement :

```
SELECT
   WEEK(STR_TO_DATE(occurred_at, '%d-%m-%Y %H:%i')) AS
week_number,
   COUNT(DISTINCT user_id) AS active_users
FROM
   events
GROUP BY
   week_number
ORDER BY
   week_number;
```

-- 2. User Growth Analysis:

```
SELECT
   YEAR(STR_TO_DATE(created_at, '%d-%m-%Y')) AS year,
   MONTH(STR_TO_DATE(created_at, '%d-%m-%Y')) AS month_number,
   COUNT(DISTINCT user_id) AS new_users
FROM
   users
GROUP BY
   year, month_number
ORDER BY
   year, month_number;
```

-- 3. Weekly Retention Analysis:

```
WITH ctel AS (
   SELECT DISTINCT
      user_id,
      EXTRACT(WEEK FROM occurred_at) AS signup_week
```

```
  FROM
    events
  WHERE
    event_type = 'signup_flow'
    AND event_name = 'complete_signup'
    AND EXTRACT(WEEK FROM occurred_at) = 18
),
cte2 AS (
  SELECT DISTINCT
    user_id,
    EXTRACT(WEEK FROM occurred_at) AS engagement_week
  FROM
    events
  WHERE
    event_type = 'engagement'
)
SELECT
  COUNT(user_id) AS total_engaged_users,
  SUM(CASE WHEN retention_week > 0 THEN 1 ELSE 0 END) AS
retained_users
FROM (
  SELECT
    a.user_id,
    a.signup_week,
    b.engagement_week,
    b.engagement_week - a.signup_week AS retention_week
  FROM
    ctel a
  LEFT JOIN
    cte2 b ON a.user_id = b.user_id
  ORDER BY
    a.user_id
) sub;
```

-- 4. Weekly Engagement Per Device:

```
SELECT
  device_name,
  AVG(num_users_using_device) AS avg_weekly_users,
  AVG(times_device_use_current_week) AS avg_times_used_weekly
FROM (
  SELECT
    WEEK(occurred_at) AS week,
    device AS device_name,
    COUNT(DISTINCT user_id) AS num_users_using_device,
        COUNT(device) AS times_device_use_current_week
    FROM events
    WHERE event_name = 'login'
    GROUP BY 1, 2
) a
GROUP BY 1;
```

-- 5. Email Engagement Analysis:

```
SELECT
    100 * SUM(CASE WHEN email_cat = 'email_open' THEN 1 ELSE 0
END) /
        SUM(CASE WHEN email_cat = 'email_sent' THEN 1 ELSE 0 END) AS
email_open_rate,
    100 * SUM(CASE WHEN email_cat = 'email_clicked' THEN 1 ELSE 0
END) /
        SUM(CASE WHEN email_cat = 'email_sent' THEN 1 ELSE 0 END) AS
email_click_rate
FROM (
    SELECT
        *,
        CASE
            WHEN action IN ('sent_weekly_digest',
'sent_reengagement_email') THEN 'email_sent'
            WHEN action IN ('email_open') THEN 'email_open'
            WHEN action IN ('email_clickthrough') THEN 'email_clicked'
        END AS email_cat
    FROM email_events
) sub;
```