Link - https://drive.google.com/file/d/1Ab5EevDUlSzQ1Rl_agV8nCxx-wUnUyb8/view?usp=sharing

Case Study -1: Operation Analytics

Project Description - As shown in the table, it is about the actors auditioning by different organizations in November. The table provides information about the organization, the audition date, the unique audition id assigned to each actor, the language in which they will perform**,** organizations spend time reviewing their performance (in seconds) and based on this, they make a decision whether to skip (reject), make a decision (select) or transfer.
From the table, one can discover which language is popular, which organizations prefer which language, no of jobs are reviewed each day or week, which actors are skipped, transferred, or selected, and which actors gave multiple auditions in November.

Approach - To get a better understanding of the table, we must first understand different events and their sense. Next, we can ask questions which will drive us to make various decisions.

Tech-stack used – PostgreSQL 14 (in which mainly used pgAdmin version 6.4) to manage tables and to get results with the help of SQL query tools.

Insights – Through the Operation analytics case study, we learned basic SQL queries, aggregation functions, sorting, and date functions.
As we can see that Persian is the most preferred language and organization A has not selected any actor in the month of November.

Result – While working on case study I get to know how important it is to understand database schema or tables first, which will help us to answer asked questions easily.

QA: **Calculate the number of jobs reviewed per hour per day for November 2020?**
SELECT ds, COUNT(job_id)*3600/SUM(time_spent) AS no_jobs_per_hour_day FROM job_data

```
WHERE ds BETWEEN '2020-11-01' AND '2020-11-30'
GROUP BY ds;
```

QB: **Let's say the above metric is called throughput. Calculate 7 day rolling average of throughput? For throughput, do you prefer daily metric or 7-day rolling and why?**

```
SELECT ds, ROUND(1.0*SUM(COUNT(job_id)) OVER(ORDER BY ds ROWS BETWEEN 6 PRECEDING AND CURRENT ROW)/SUM(SUM(time_spent)) OVER(ORDER BY ds ROWS BETWEEN 6 PRECEDING AND CURRENT ROW),2) AS rolling_average FROM job_data
WHERE ds BETWEEN '2020-11-01' AND '2020-11-30'
GROUP BY ds;
```

I will prefer a daily metric because the stats for each day are changing with respect to different actors for the job.

QC: **Calculate the percentage share of each language in the last 30 days?**

```
SELECT language, ROUND(((COUNT(*)*100)/(SELECT COUNT(*)FROM job_data)),2) AS percentage_of_each_language
FROM job_data
WHERE ds BETWEEN '2020-11-01' AND '2020-11-30'
GROUP BY language
ORDER BY language;
```

QD: **Let's say you see some duplicate rows in the data. How will you display duplicates from the table?**

```
SELECT * FROM
(SELECT *, COUNT(*)
OVER
(PARTITION BY job_id) AS duplicate_count
FROM job_data) tableWithCount
WHERE tableWithCount.duplicate_count>1;
```

Case Study – 2: Investigating metric Spike

Project Description – From the given table we can get a sense that this data belongs to a company named YAMMER (mentioned in table 2 -events) which is a social networking platform for communicating within groups(like Facebook).
When we answer weekly user engagement questions, we find that from 28-July-2014 there is a sudden dip in user engagement.

Approach – Knowing there is a dip in user engagement, we must note all possible reasons for the dip before using our SQL knowledge to get insights, which will help us gain a better understanding of the problem.

Tech-stack used - PostgreSQL 14 (in which mainly used pgAdmin version 6.4) to manage tables and to get results with help of SQL query tool and Microsoft Excel for getting visual insights with help of pivot tables.

Insights – User's engagement is high during week days and low on weekends, we can see steep dip in engagement using phones from which we might say that there may be problem related to mobile application and we can also see that drop is more in Greece, Ireland, Pakistan, Chile…. whereas biggest engagement in United States, Japan, France, Germany…. .

Result - Through case study 2, we are not just thinking about how to solve queries using SQL, but also about the meaning behind what we are measuring. Due to which I get to retrieve more useful insights from data.

QA: Calculate the weekly user engagement?

```
SELECT DATE_TRUNC('week', e.occurred_At), COUNT(DISTINCT e.user_id) AS
weekly_users_active
FROM events e
WHERE e.event_type='engagement'
AND e.event_name='login'
```

```
GROUP BY DATE_TRUNC('week', occurred_at)
ORDER BY DATE_TRUNC('week', occurred_at);
```

QB: Calculate the user growth for the product?

```
SELECT DATE_TRUNC('week', u.created_at) AS day, COUNT(*) AS weekly_all_users,
COUNT(CASE WHEN activated_at IS NOT NULL THEN u.user_id ELSE NULL END) AS
activated_users
FROM users u
WHERE created_at >='2013-01-01'
AND created_at<='2014-08-31'
GROUP BY DATE_TRUNC('week', created_at)
ORDER BY DATE_TRUNC('week', created_at);
```

QD: Calculate the weekly engagement per device?

```
SELECT e.device,COUNT(DISTINCT e.user_id) AS weekly_active_users
FROM events e
WHERE e.event_type='engagement'
AND e.event_name='login'
GROUP BY e.device
ORDER BY weekly_active_users;
```

NOTE – To get a better understanding we may divide all devices into three types – phone, tablets and computers.

QE: Calculate the email engagement metrics?

```
SELECT DATE_TRUNC('week',occurred_at) AS week, COUNT(CASE WHEN
e.action='sent_weekly_digest' THEN e.user_id ELSE NULL END) AS weekly_emails,
COUNT(CASE WHEN e.action='sent_reengagement_email' THEN e.user_id ELSE NULL
END) AS reengagement_emails,
COUNT(CASE WHEN e.action='email_open' THEN e.user_id ELSE NULL END) AS
emails_open,
COUNT(CASE WHEN e.action='email_clickthrough' THEN e.user_id ELSE NULL END)
AS emails_clickthrough
```

```
FROM email_events e
GROUP BY DATE_TRUNC('week', occurred_at);
```