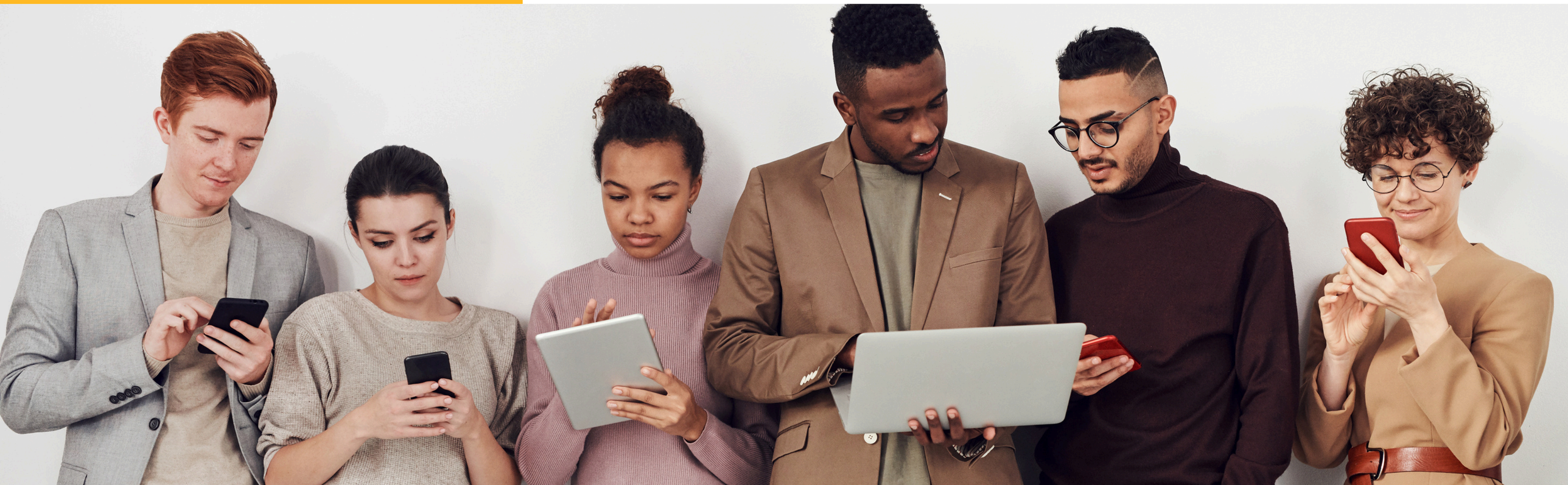
SQL ANALYSIS

USER LOGIN ANALYSIS

Presented by AJIT TIWARI

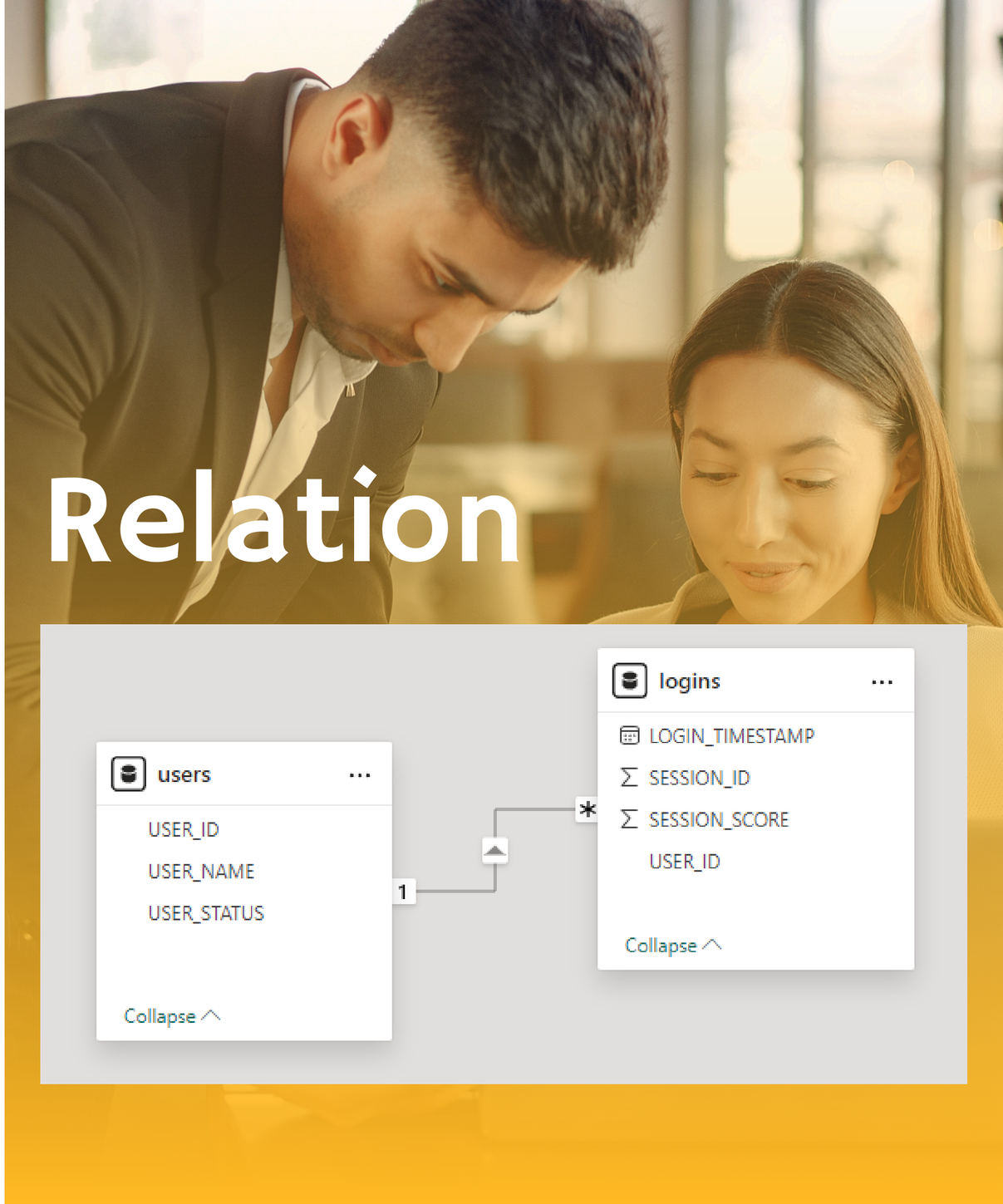


Agenda

- Introduction
- Objective
- Query
- Analysis
- Recommendations

Introduction of User Logins

This analysis involves a detailed examination of user login behavior and activity patterns within our system. The insights derived from this analysis will aid in understanding user engagement, identifying trends, and making data-driven decisions to enhance user retention and activity.



Objectives

1. Percentage Change in session from the last quarter.
   Return first date of quarter, total users, total sessions, percentage change

2. To identify our best users- Return the users that had a session on every single day since their first login.

3. On what dates there were no logins at all.

4. Display the user which has the highest session score for each day.

5. Display Users that logged-in in January 2024 and did not logged-in in November 2023.

6. For the business unit's quaterly analysis. Calculate how many users and how many sessions were at each quarter. Return first day of the quarter, user counts, session counts.

7. Management wants to see all users that did not login in the last 5 Months.

Query

Percentage Change in session from the last quarter. Return first date of quarter, total users, total sessions, percentage change

```
WITH sessions AS(
                SELECT DATE_FORMAT(MIN(login_timestamp), '%Y-%m-%01') AS first_qtr_date,
                        COUNT(DISTINCT user_id) AS total_users,
                            COUNT(session_id) AS total_sessions
                    FROM logins
                GROUP BY QUARTER(login_timestamp)
                )
SELECT *,
 (total_sessions - LAG(total_sessionS) OVER (ORDER BY first_qtr_date) ) *100
    / LAG(total_sessionS) OVER (ORDER BY first_qtr_date)  AS session_pct_change
FROM sessions;
```

Output

first_qtr_date	total_users	total_sessions	session_pct_change
2023-07-01	6	9	NULL
2023-10-01	6	7	-22.2222
2024-01-01	5	8	14.2857
2024-04-01	5	8	0.0000

Query

To identify our best users- Return the users that had a session on every single day since their first login.

```
WITH cte AS(
            SELECT user_id
                FROM (
                    SELECT user_id,
                           DATEDIFF(CURDATE(), MIN(DATE(login_timestamp))) +1 AS required_log_ins,
                            COUNT(DATE(login_timestamp)) AS no_days_logged_in
                       FROM logins
                    GROUP BY user_id
                    )Z
            WHERE required_log_ins = no_days_logged_in
            )
SELECT u.user_id, u.user_name
    FROM users u
INNER JOIN cte c
ON u.user_id = c.user_id;
```

Output

user_id	user_name
11	Jake

Query

On what dates there were no logins at all.

```sql
WITH RECURSIVE cte AS (
        SELECT
            MIN(DATE(login_timestamp)) AS start_date,
                CURDATE() AS end_date
        FROM logins
        UNION ALL
        SELECT ADDDATE(start_date , INTERVAL 1 DAY), end_date
            FROM cte
        WHERE start_date < end_date
        )
SELECT start_date AS no_login_date
FROM cte
WHERE
    start_date NOT IN (
    SELECT DISTINCT DATE(login_timestamp) FROM logins);
```

no_login_date
2023-07-16
2023-07-17
2023-07-18
2023-07-19
2023-07-20
2023-07-21
2023-07-23
2023-07-24
2023-07-25
2023-07-26
2023-07-27
2023-07-28
2023-07-29
2023-07-30
2023-07-31
2023-08-01
2023-08-02
2023-08-03
2023-08-04
2023-08-05

Output

Query

Display the user which has the highest session score for each day.

```sql
WITH cte AS (
            SELECT user_id, DATE(login_timestamp) AS login_dates, SUM(session_score) AS total_session_score
            FROM logins
            GROUP BY user_id, DATE(login_timestamp)
            )
SELECT user_id, login_dates, total_session_score
FROM (
        SELECT *,
                RANK() OVER(PARTITION BY login_dates ORDER BY total_session_score DESC) AS rn
        FROM cte
        )Z
WHERE rn=1;
```

user_id	login_dates	total_session_score
1	2023-07-15	85
2	2023-07-22	90
3	2023-08-10	75
4	2023-08-20	88
5	2023-09-05	82
6	2023-10-12	77
2	2023-11-10	82
6	2023-11-15	80
7	2023-11-18	81
4	2023-11-25	84
8	2023-12-01	84
9	2023-12-15	79
1	2024-01-10	172
5	2024-01-15	78
2	2024-01-25	89
3	2024-01-25	89
3	2024-02-05	78
4	2024-03-01	91
5	2024-03-15	83
6	2024-04-12	80

Output

Query

Display Users that logged-in in January 2024 and did not logged-in in November 2023.

```sql
WITH cte AS (
            SELECT DISTINCT user_id
                FROM logins
            WHERE login_timestamp BETWEEN '2024-01-01' AND '2024-01-31'
                AND user_id NOT IN
                        ( SELECT user_id
                            FROM logins
                            WHERE login_timestamp BETWEEN '2023-11-01' AND '2023-11-30'
                        )
                )
SELECT u.user_id, u.user_name
    FROM cte c
INNER JOIN users u
ON c.user_id= u.user_id;
```

Output

user_id	user_name
1	Alice
3	Charlie
5	Eve

Query

For the business unit's quaterly analysis. Calculate how many users and how many sessions were at each quarter. Return first day of the quarter, user counts, session counts.

```sql
SELECT DATE_FORMAT(MIN(login_timestamp), '%Y-%m-%01') AS first_qtr_date,
        COUNT(DISTINCT user_id) AS total_users,
            COUNT(session_id) AS total_sessions
    FROM logins
GROUP BY QUARTER(login_timestamp);
```

Output

first_qtr_date	total_users	total_sessions
2024-01-01	5	8
2024-04-01	5	8
2023-07-01	6	9
2023-10-01	6	7

Query

Management wants to see all users that did not login in the last 5 Months.

```
WITH no_login AS
                (
                SELECT user_id, MAX(login_timestamp) AS last_login
                    FROM logins
                 GROUP BY user_id
                HAVING
                    MAX(login_timestamp) < SUBDATE(CURDATE(), INTERVAL 5 MONTH)
                )
SELECT u.user_id, u.user_name, n.last_login
    FROM no_login n
INNER JOIN users u
ON n.user_id = u.user_id;
```

Output

user_id	user_name	last_login
1	Alice	2024-01-10 07:45:00
2	Bob	2024-01-25 09:30:00
3	Charlie	2024-02-05 11:00:00
4	David	2024-03-01 14:30:00

Analysis & Recommendations

- The percentage change in session count from one quarter to the next is calculated. This metric is crucial for understanding growth or decline in user activity over time.
- The query identifies users who have logged in every single day since their first login. These users are considered the most consistent and engaged, valuable for loyalty programs and rewards.
- This analysis identifies dates with no login activity at all. Understanding these patterns can help in recognizing periods of low engagement and addressing potential underlying issues.
- This analysis determines the user with the highest session score for each day. This can help in identifying highly engaged users on a daily basis.
- The query identifies users who were active in January 2024 but inactive in November 2023. This helps in understanding seasonal or month-to-month variations in user activity.
- This analysis calculates the total number of distinct users and sessions for each quarter. It returns the first day of each quarter along with these counts, providing insights into user engagement trends over time.
- The query identifies users who have not logged into the system in the past 5 months. This helps in targeting re-engagement campaigns to bring these users back to the platform.

Thank you!