"Exploring Database Management: A Case Study"

A step-by-step guide to creating, managing, and analyzing data in SQL



Managing User Logins and Sessions for Analytical Insights



e de la companyation de la compa

QUERYING RECENT INACTIVE USERS

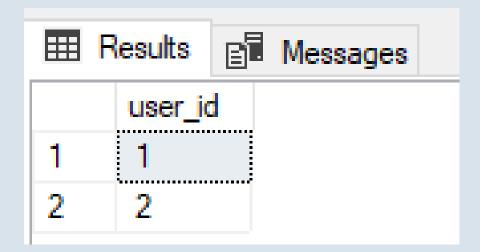


Queries to identify users who haven't logged in within the past five months.

SOLUTION:

select distinct user_id from logins where user_id not in (
select user_id from logins where LOGIN_TIMESTAMP>dateadd(month,-5,'2 024-06-28'));

OUTPUT:



Insight:

"Great for engagement analysis or churn prediction"

200

Quarterly Insights on User Activity



Uses CTE to count sessions and users by quarter.

SOLUTION:

select datepart(quarter,login_timestamp)
 as quarterr,count(*) as
 session_cnt,count(distinct user_id) as
 user_cnt,
 min(login_timestamp) as first_date,
 datetrunc(quarter
 ,min(login_timestamp))
 from logins
 group by
 datepart(quarter,login_timestamp);

OUTPUT:

	quarterr	session_cnt	user_cnt	first_date	(No column name)
	1	8	5	2024-01-10 07:45:00.000	2024-01-01 00:00:00.000
2	2	8	5	2024-04-12 08:00:00.000	2024-04-01 00:00:00.000
3	3	5	5	2023-07-15 09:30:00.000	2023-07-01 00:00:00.000
4	4	7	6	2023-10-12 08:30:00.000	2023-10-01 00:00:00.000

Insight:

Quarterly analysis shows stable engagement with 8 sessions in Q1 and Q2, a dip to 5 in Q3, and a rebound to 7 in Q4

and the second second

Identifying User Activity

To find users who logged in during January 2024 but did not log in at all in November 2024.

SOLUTION:

SELECT * 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');

OUTPUT:

	Results		Messages		
	USER	_ID	LOGIN_TIMESTAMP	SESSION_ID	SESSION_SCORE
1	1		2024-01-10 07:45:00.000	1011	86
2	1		2024-01-10 07:45:00.000	1101	86
3	3		2024-01-25 09:30:00.000	1102	89
4	5		2024-01-15 11:00:00.000	1103	78

Insight:

User Engagement Tracking: This query helps identify users with inconsistent login behavior.

Quarterly Session Analysis with Percentage Change

To analyze session counts by quarter and calculate the percentage change in sessions compared to the previous quarter.

SOLUTION:

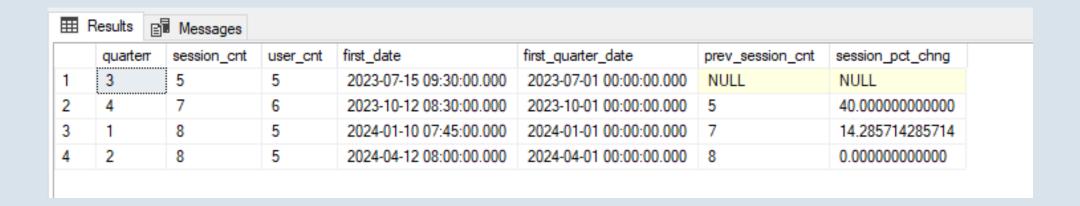
WITH cte AS (SELECT

SELECT*,

LAG(session_cnt, 1) OVER (ORDER BY first_quarter_date) AS prev_session_cnt,

(session_cnt - LAG(session_cnt, 1) OVER (ORDER BY first_quarter_date)) * 100.0 / LAG(session_cnt, 1) OVER (ORDER BY first_quarter_date) AS session_pct_chng FROM cte;

OUTPUT:



INSIGHTS:

In Q4 2023, sessions increased by 40% from Q3, while Q1 2024 saw a 14.3% increase from Q4, maintaining steady user engagement; however, Q2 2024 plateaued with no session growth compared to Q1.

2000

e de la companya della companya della companya de la companya della companya dell

Daily Top Users by Session Score

To identify and display the user with the highest session score for each day.

SOLUTION:

```
WITH cte AS (
SELECT
user_id,
CAST(login_timestamp AS DATE) AS
login_date,
SUM(session_score) AS score
FROM logins
GROUP BY user_id, CAST(login_timestamp AS DATE)

DATE)
)
```

```
SELECT * FROM (
SELECT *,
ROW_NUMBER() OVER (PARTITION BY login_date ORDER BY score DESC) AS rn
FROM cte
) a
WHERE rn = 1;
```

200



OUTPUT:

Results							
	user_id	login_date	score	m			
1	1	2023-07-15	85	1			
2	2	2023-07-22	90	1			
3	3	2023-08-10	75	1			
4	4	2023-08-20	88	1			
5	5	2023-09-05	82	1			
6	6	2023-10-12	77	1			
7	2	2023-11-10	82	1			
8	6	2023-11-15	80	1			
9	7	2023-11-18	81	1			
10	4	2023-11-25	84	1			
11	8	2023-12-01	84	1			
12	9	2023-12-15	79	1			
13	1	2024-01-10	172	1			
14	5	2024-01-15	78	1			
15	2	2024-01-25	89	1			
16	3	2024-02-05	78	1			
17	4	2024-03-01	91	1			
18	5	2024-03-15	83	1			

INSIGHTS:

User 1 peaked at 172 on January 10, 2024, while User 10 consistently scored above 90 in late June, reflecting strong engagement and performance variability across users.

e de la company de la company

Identifying Best Users with Continuous Engagement

Find Users with Daily Sessions Since First Login

SOLUTION:

SELECT user_id,

MIN(cast(login_timestamp AS date)) AS first_login, DATEDIFF(day, MIN(cast(login_timestamp AS date)), '2024-06-28') + 1 AS no_of_days,

COUNT(DISTINCT cast(login_timestamp AS date))

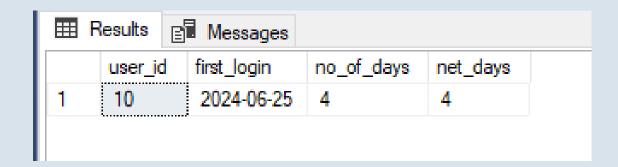
AS net_days

FROM logins

GROUP BY user_id

HAVING DATEDIFF(day, MIN(cast(login_timestamp AS date)), '2024-06-28') + 1 = COUNT(DISTINCT cast(login_timestamp AS date));

OUTPUT:



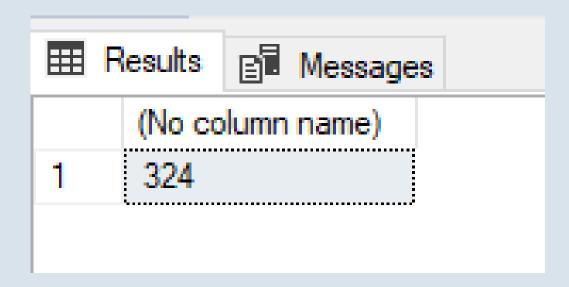


Days with No Logins

SOLUTION:

select count(*) from calender_dim c inner join (select cast(min(login_timestamp) as date) as first_date, cast('2024-06-28' as date) as last_date from logins)a on c.cal_date between first_date and last_date where cal_date not in (select distinct cast(login_timestamp as date) from logins)

OUTPUT:



INSIGHTS:

Many days of inactivity suggest engagement issues and opportunities for targeted re-engagement efforts.