Problem 3673: Find Zombie Sessions

Problem Information

Difficulty: Hard
Acceptance Rate: 0.00%
Paid Only: No

Problem Description

Table:

app_events

+------------------+----------+ | Column Name | Type | +------------------+----------+ | event_id | int | | user_id | int | | event_timestamp | datetime | | event_type | varchar | | session_id | varchar | | event_value | int | +------------------+----------+ event_id is the unique identifier for this table. event_type can be app_open, click, scroll, purchase, or app_close. session_id groups events within the same user session. event_value represents: for purchase - amount in dollars, for scroll - pixels scrolled, for others - NULL.

Write a solution to identify

zombie sessions,

sessions where users appear active but show abnormal behavior patterns. A session is considered a

zombie session

if it meets ALL the following criteria:

The session duration is

more than

30

minutes.

Has

at least

5

scroll events.

The

click-to-scroll ratio

is less than

0.20

.

No purchases

were made during the session.

Return

the result table ordered by

scroll_count

in

descending

order, then by

session_id

in

ascending

order

.

The result format is in the following example.

Example:

Input:

app_events table:

event_id	user_id	event_timestamp	event_type	session_id	event_value
1	201	2024-03-01 10:00:00	app_open	S001	NULL
2	201	2024-03-01 10:05:00	scroll	S001	500
3	201	2024-03-01 10:10:00	scroll	S001	750
4	201	2024-03-01 10:15:00	scroll	S001	600
5	201	2024-03-01 10:20:00	scroll	S001	800
6	201	2024-03-01 10:25:00	scroll	S001	550
7	201	2024-03-01 10:30:00	scroll	S001	900
8	201	2024-03-01 10:35:00	app_close	S001	NULL
9	202	2024-03-01 11:00:00	app_open	S002	NULL
10	202	2024-03-01 11:02:00	click	S002	NULL
11	202	2024-03-01 11:05:00	scroll	S002	400
12	202	2024-03-01 11:08:00	click	S002	NULL
13	202	2024-03-01 11:10:00	scroll	S002	350
14	202	2024-03-01 11:15:00	purchase	S002	50
15	202	2024-03-01 11:20:00	app_close	S002	NULL
16	203	2024-03-01 12:00:00	app_open	S003	NULL
17	203	2024-03-01 12:10:00	scroll	S003	1000
18	203	2024-03-01 12:20:00	scroll	S003	1200
19	203	2024-03-01 12:25:00	click	S003	NULL
20	203	2024-03-01 12:30:00	scroll	S003	800
21	203	2024-03-01 12:40:00	scroll	S003	900
22	203	2024-03-01 12:50:00	scroll	S003	1100
23	203	2024-03-01 13:00:00	app_close	S003	NULL
24	204	2024-03-01 14:00:00	app_open	S004	NULL
25	204	2024-03-01 14:05:00	scroll	S004	600
26	204	2024-03-01 14:08:00	scroll	S004	700
27	204	2024-03-01 14:10:00	click	S004	NULL
28	204	2024-03-01 14:12:00	app_close	S004	NULL

Output:

```
+------------+---------+--------------------------+--------------+ | session_id | user_id |
session_duration_minutes | scroll_count |
+------------+---------+--------------------------+--------------+ | S001 | 201 | 35 | 6 |
+------------+---------+--------------------------+--------------+
```

Explanation:

Session S001 (User 201)

:

Duration: 10:00:00 to 10:35:00 = 35 minutes (more than 30)

Scroll events: 6 (at least 5)

Click events: 0

Click-to-scroll ratio: 0/6 = 0.00 (less than 0.20)

Purchases: 0 (no purchases)

S001 is a zombie session (meets all criteria)

Session S002 (User 202)

:

Duration: 11:00:00 to 11:20:00 = 20 minutes (less than 30)

Has a purchase event

S002 is not a zombie session

Session S003 (User 203)

:

Duration: 12:00:00 to 13:00:00 = 60 minutes (more than 30)

Scroll events: 5 (at least 5)

Click events: 1

Click-to-scroll ratio: 1/5 = 0.20 (not less than 0.20)

Purchases: 0 (no purchases)

S003 is not a zombie session (click-to-scroll ratio equals 0.20, needs to be less)

Session S004 (User 204)

:

Duration: 14:00:00 to 14:12:00 = 12 minutes (less than 30)

Scroll events: 2 (less than 5)

S004 is not a zombie session

The result table is ordered by scroll_count in descending order, then by session_id in ascending order.

Code Snippets

MySQL:

```
# Write your MySQL query statement below
```

MS SQL Server:

```
/* Write your T-SQL query statement below */
```

PostgreSQL:

```
-- Write your PostgreSQL query statement below
```

Oracle:

```
/* Write your PL/SQL query statement below */
```

Pandas:

```
import pandas as pd


def find_zombie_sessions(app_events: pd.DataFrame) -> pd.DataFrame:
```

Solutions

MySQL Solution:

```
# Write your MySQL query statement below
```

MS SQL Server Solution:

```
/* Write your T-SQL query statement below */
```

PostgreSQL Solution:

```
-- Write your PostgreSQL query statement below
```

Oracle Solution:

```
/* Write your PL/SQL query statement below */
```

Pandas Solution:

```
import pandas as pd


def find_zombie_sessions(app_events: pd.DataFrame) -> pd.DataFrame:
```