**DO NOT USE GPT Tools!!! PostgreSQL Preferred!**

**If using other database languages, please indicate.**

**1.** Assume you're given a table containing information about **DELL** user transactions for different products. Write a query to calculate the year-on-year growth rate for the total spend of each product, grouping the results by product ID.

The output should include the year in ascending order, product ID, current year's spend, previous year's spend and year-on-year growth percentage, rounded to 2 decimal places.

**user\_transactions Table:**

| **Column Name** | **Type** |
| --- | --- |
| transaction\_id | integer |
| product\_id | integer |
| spend | decimal |
| transaction\_date | datetime |

**user\_transactions Example Input:**

| **transaction\_id** | **product\_id** | **spend** | **transaction\_date** |
| --- | --- | --- | --- |
| E2341341 | 123424 | 1500.60 | 12/31/2020 11:56:00 |
| E2341423 | 123424 | 1000.20 | 12/31/2021 12:10:00 |
| E2341623 | 123424 | 1246.44 | 12/31/2022 09:30:00 |
| E2341322 | 123424 | 2145.32 | 12/31/2023 14:23:00 |

**Example Output:**

| **Year** | **product\_id** | **curr\_year\_spend** | **prev\_year\_spend** | **yoy\_rate** |
| --- | --- | --- | --- | --- |
| 2020 | 123424 | 1500.60 | NULL | NULL |
| 2021 | 123424 | 1000.20 | 1500.60 | -33.35 |
| 2022 | 123424 | 1246.44 | 1000.20 | 24.62 |
| 2023 | 123424 | 2145.32 | 1246.44 | 72.12 |

**2.** Sometimes, payment transactions are repeated by accident; it could be due to user error, API failure or a retry error that causes a credit card to be charged twice.

Using the transactions table, identify any payments made at the same merchant with the same credit card for the same amount within 10 minutes of each other. Count such repeated payments.

**Assumptions:**

* The first transaction of such payments should not be counted as a repeated payment. This means, if there are two transactions performed by a merchant with the same credit card and for the same amount within 10 minutes, there will only be 1 repeated payment.

**transactions Table:**

| **Column Name** | **Type** |
| --- | --- |
| transaction\_id | integer |
| merchant\_id | integer |
| credit\_card\_id | integer |
| amount | integer |
| transaction\_timestamp | datetime |

**transactions Example Input:**

| **transaction\_id** | **merchant\_id** | **credit\_card\_id** | **amount** | **transaction\_timestamp** |
| --- | --- | --- | --- | --- |
| 1 | 101 | 1 | 100 | 09/25/2022 12:00:00 |
| 2 | 101 | 1 | 100 | 09/25/2022 12:08:00 |
| 3 | 101 | 1 | 100 | 09/25/2022 12:28:00 |
| 4 | 102 | 2 | 300 | 09/25/2022 12:00:00 |
| 6 | 102 | 2 | 400 | 09/25/2022 14:00:00 |
| 7 | 103 | 3 | 560 | 09/25/2022 15:05:00 |
| 8 | 104 | 4 | 120 | 09/25/2022 15:20:00 |
| 9 | 104 | 4 | 130 | 09/25/2022 15:26:00 |
| 10 | 104 | 4 | 130 | 09/25/2022 15:31:00 |

**Example Output:**

| **repeated\_payment\_count** |
| --- |
| 1 |

**3.** Given a table of tweet data over a specified time period, calculate the 3-day rolling average of tweets for each user. Output the user ID, tweet date, and rolling averages rounded to 2 decimal places.

**Notes:**

* A rolling average, also known as a moving average or running mean is a time-series technique that examines trends in data over a specified period of time.
* In this case, we want to determine how the tweet count for each user changes over a 3-day period.

**tweets Table:**

| **Column Name** | **Type** |
| --- | --- |
| user\_id | integer |
| tweet\_date | timestamp |
| tweet\_count | integer |

**tweets Example Input:**

| **user\_id** | **tweet\_date** | **tweet\_count** |
| --- | --- | --- |
| 111 | 06/01/2022 00:00:00 | 2 |
| 111 | 06/02/2022 00:00:00 | 1 |
| 111 | 06/03/2022 00:00:00 | 3 |
| 111 | 06/04/2022 00:00:00 | 4 |
| 111 | 06/05/2022 00:00:00 | 5 |

**Example Output:**

| **user\_id** | **tweet\_date** | **rolling\_avg\_3d** |
| --- | --- | --- |
| 111 | 06/01/2022 00:00:00 | 2.00 |
| 111 | 06/02/2022 00:00:00 | 1.50 |
| 111 | 06/03/2022 00:00:00 | 2.00 |
| 111 | 06/04/2022 00:00:00 | 2.67 |
| 111 | 06/05/2022 00:00:00 | 4.00 |