DESCRIPTION

Table: Activity

+----------------+---------+

| Column Name | Type |

+----------------+---------+

| machine_id | int |

| process_id | int |

| activity_type | enum |

| timestamp | float |

+----------------+---------+

The table shows the user activities for a factory website.

(machine_id, process_id, activity_type) is the primary key (combination of columns with unique values) of this table.

machine_id is the ID of a machine.

process_id is the ID of a process running on the machine with ID machine_id.

activity_type is an ENUM (category) of type ('start', 'end').

timestamp is a float representing the current time in seconds.

'start' means the machine starts the process at the given timestamp and 'end' means the machine ends the process at the given timestamp.

The 'start' timestamp will always be before the 'end' timestamp for every (machine_id, process_id) pair.

It is guaranteed that each (machine_id, process_id) pair has a 'start' and 'end' timestamp.

There is a factory website that has several machines each running the **same number of processes**. Write a solution to find the **average time** each machine takes to complete a process.

The time to complete a process is the 'end' timestamp minus the 'start' timestamp. The average time is calculated by the total time to complete every process on the machine divided by the number of processes that were run.

The resulting table should have the machine_id along with the **average time** as processing_time, which should be **rounded to 3 decimal places**.

Return the result table in **any order**.

The result format is in the following example.

Example 1:

Input:

Activity table:

+------------+------------+---------------+-----------+

| machine_id | process_id | activity_type | timestamp |

+------------+------------+---------------+-----------+

| 0 | 0 | start | 0.712 |

| 0 | 0 | end | 1.520 |

| 0 | 1 | start | 3.140 |

| 0 | 1 | end | 4.120 |

| 1 | 0 | start | 0.550 |

| 1 | 0 | end | 1.550 |

| 1 | 1 | start | 0.430 |

| 1 | 1 | end | 1.420 |

| 2 | 0 | start | 4.100 |

| 2 | 0 | end | 4.512 |

| 2 | 1 | start | 2.500 |

| 2 | 1 | end | 5.000 |

+------------+------------+---------------+-----------+

Output:

+------------+-----------------+

| machine_id | processing_time |

+------------+-----------------+

| 0 | 0.894 |

| 1 | 0.995 |

| 2 | 1.456 |

+------------+-----------------+

Explanation:

There are 3 machines running 2 processes each.

Machine 0's average time is ((1.520 - 0.712) + (4.120 - 3.140)) / 2 = 0.894

Machine 1's average time is ((1.550 - 0.550) + (1.420 - 0.430)) / 2 = 0.995

Machine 2's average time is ((4.512 - 4.100) + (5.000 - 2.500)) / 2 = 1.456

SOLUTION in Pandas

Option 1:

- Sort 'activity' using **DataFrame.sort_values**
- Create a groupby object using **DataFrame.groupby**, find time differences in 'timestamp' column using **DataFrame.diff** and add a column, 'time_difference', for the calculated time differences
- Create a new dataframe with unique 'machine_id' using **DataFrame.unique and** average 'processing_time' for each machine id using **DataFrame.mean** (grouped by 'machine_id') and **DataFrame.round** for 3 decimal places

```
import pandas as pd

def get_average_time(activity: pd.DataFrame) -> pd.DataFrame:
    activity = activity.sort_values(by=['machine_id', 'process_id', 'timestamp'])
    activity['time_difference'] = activity.groupby(['machine_id', 'process_id'])['timestamp'].diff()
    d = {'machine_id' : activity['machine_id'].unique(), 'processing_time' : activity.groupby
('machine_id')['time_difference'].mean().round(3)}
    df = pd.DataFrame(data = d)
    return df
```