DESCRIPTION

Table: Users

```
+-------------+---------+
| Column Name | Type    |
+-------------+---------+
| user_id     | int     |
| user_name   | varchar |
+-------------+---------+
```

user_id is the primary key (column with unique values) for this table.

Each row of this table contains the name and the id of a user.

Table: Register

```
+-------------+---------+
| Column Name | Type    |
+-------------+---------+
| contest_id  | int     |
| user_id     | int     |
+-------------+---------+
```

(contest_id, user_id) is the primary key (combination of columns with unique values) for this table.

Each row of this table contains the id of a user and the contest they registered into.

Write a solution to find the percentage of the users registered in each contest rounded to **two decimals**.

Return the result table ordered by percentage in **descending order**. In case of a tie, order it by contest_id in **ascending order**.

The result format is in the following example.

Example 1:

Input:

Users table:

```
+---------+-----------+
| user_id | user_name |
+---------+-----------+
| 6       | Alice     |
| 2       | Bob       |
| 7       | Alex      |
+---------+-----------+
```

Register table:

```
+------------+---------+
| contest_id | user_id |
+------------+---------+
| 215        | 6       |
| 209        | 2       |
| 208        | 2       |
| 210        | 6       |
| 208        | 6       |
| 209        | 7       |
| 209        | 6       |
| 215        | 7       |
| 208        | 7       |
| 210        | 2       |
| 207        | 2       |
| 210        | 7       |
+------------+---------+
```

Output:

```
+------------+------------+
| contest_id | percentage |
+------------+------------+
| 208        | 100.0      |
| 209        | 100.0      |
| 210        | 100.0      |
| 215        | 66.67      |
| 207        | 33.33      |
+------------+------------+
```

Explanation:

All the users registered in contests 208, 209, and 210. The percentage is 100% and we sort them in the answer table by contest_id in ascending order.

Alice and Alex registered in contest 215 and the percentage is ((2/3) * 100) = 66.67%

Bob registered in contest 207 and the percentage is ((1/3) * 100) = 33.33%

SOLUTION

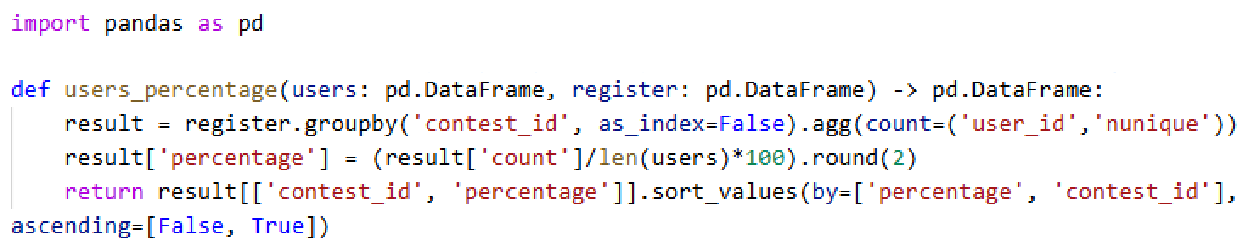
Option 1:

- Groupby ‘register’ table with ‘contest_id’ column
- Compute ‘count’ column using agg() on ‘user_id’ column with ‘unique’
- For ‘percentage’ column, divide ‘count’ by len(users) and multiply by 100, and round to 2 digits using round()
- Return the dataframe ordered by ‘percentage’ and ‘contest_id’ using sorted_values()

```
import pandas as pd

def users_percentage(users: pd.DataFrame, register: pd.DataFrame) -> pd.DataFrame:
    result = register.groupby('contest_id', as_index=False).agg(count=('user_id','nunique'))
    result['percentage'] = (result['count']/len(users)*100).round(2)
    return result[['contest_id', 'percentage']].sort_values(by=['percentage', 'contest_id'],
ascending=[False, True])
```

- Snapshot of the same code above for readability purposes



```
import pandas as pd

def users_percentage(users: pd.DataFrame, register: pd.DataFrame) -> pd.DataFrame:
    result = register.groupby('contest_id', as_index=False).agg(count=('user_id','nunique'))
    result['percentage'] = (result['count']/len(users)*100).round(2)
    return result[['contest_id', 'percentage']].sort_values(by=['percentage', 'contest_id'],
ascending=[False, True])
```

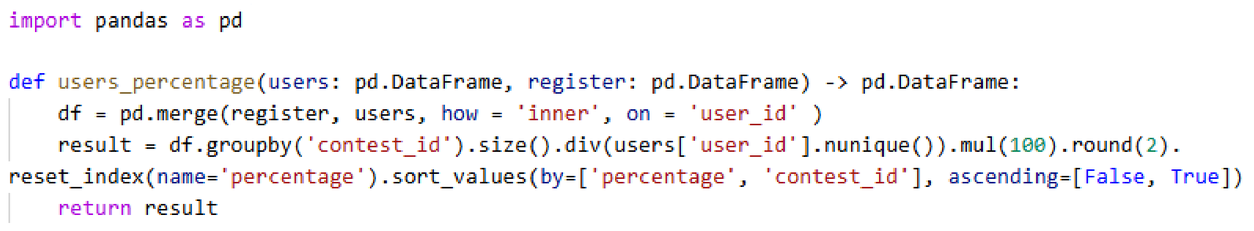
Option 2:

- Join tables using pd.merge()
- Groupby the joined table with ‘contest_id’
- For ‘percentage’ column, divide size of the joined table by unique number of ‘users’ using size() and nunique()
- Multiply by 100, and round to 2 digits using round()
- Return the table ordered by ‘percentage’ (descending) and ‘contest_id’ (ascending) using sorted_values()

```
import pandas as pd

def users_percentage(users: pd.DataFrame, register: pd.DataFrame) -> pd.DataFrame:
    df = pd.merge(register, users, how = 'inner', on = 'user_id' )
    result =
df.groupby('contest_id').size().div(users['user_id'].nunique()).mul(100).round(2).reset_index(name='percen
tage').sort_values(by=['percentage', 'contest_id'], ascending=[False, True])
    return result
```

- Snapshot of the same code above for readability purposes



```
import pandas as pd

def users_percentage(users: pd.DataFrame, register: pd.DataFrame) -> pd.DataFrame:
    df = pd.merge(register, users, how = 'inner', on = 'user_id' )
    result = df.groupby('contest_id').size().div(users['user_id'].nunique()).mul(100).round(2).
reset_index(name='percentage').sort_values(by=['percentage', 'contest_id'], ascending=[False, True])
    return result
```