**DESCRIPTION**

Table: Prices

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

| Column Name | Type |

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

| product\_id | int |

| start\_date | date |

| end\_date | date |

| price | int |

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

(product\_id, start\_date, end\_date) is the primary key (combination of columns with unique values) for this table.

Each row of this table indicates the price of the product\_id in the period from start\_date to end\_date.

For each product\_id there will be no two overlapping periods. That means there will be no two intersecting periods for the same product\_id.

Table: UnitsSold

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

| Column Name | Type |

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

| product\_id | int |

| purchase\_date | date |

| units | int |

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

This table may contain duplicate rows.

Each row of this table indicates the date, units, and product\_id of each product sold.

Write a solution to find the average selling price for each product. average\_price should be **rounded to 2 decimal places**. If a product does not have any sold units, its average selling price is assumed to be 0.

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

The result format is in the following example.

**Example 1:**

**Input:**

Prices table:

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

| product\_id | start\_date | end\_date | price |

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

| 1 | 2019-02-17 | 2019-02-28 | 5 |

| 1 | 2019-03-01 | 2019-03-22 | 20 |

| 2 | 2019-02-01 | 2019-02-20 | 15 |

| 2 | 2019-02-21 | 2019-03-31 | 30 |

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

UnitsSold table:

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

| product\_id | purchase\_date | units |

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

| 1 | 2019-02-25 | 100 |

| 1 | 2019-03-01 | 15 |

| 2 | 2019-02-10 | 200 |

| 2 | 2019-03-22 | 30 |

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

**Output:**

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

| product\_id | average\_price |

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

| 1 | 6.96 |

| 2 | 16.96 |

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

**Explanation:**

Average selling price = Total Price of Product / Number of products sold.

Average selling price for product 1 = ((100 \* 5) + (15 \* 20)) / 115 = 6.96

Average selling price for product 2 = ((200 \* 15) + (30 \* 30)) / 230 = 16.96

**SOLUTION**

Option 1:

* Filter the dataframe for odd ids and not boring using loc
* Sort in descending order using sort\_values()

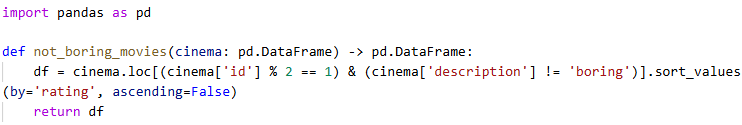
import pandas as pd

def not\_boring\_movies(cinema: pd.DataFrame) -> pd.DataFrame:

    df = cinema.loc[(cinema['id'] % 2 == 1) & (cinema['description'] != 'boring')].sort\_values(by='rating', ascending=False)

    return df

* Snapshot of the same code above for readability purposes



Option 2:

* Filter the dataframe for odd ids and not boring using loc
* Sort in descending order using sort\_values()

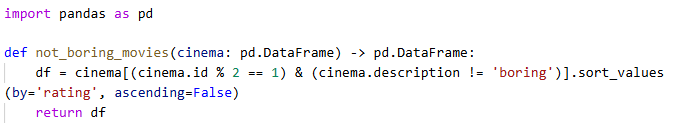
import pandas as pd

def not\_boring\_movies(cinema: pd.DataFrame) -> pd.DataFrame:

    df = cinema[(cinema.id % 2 == 1) & (cinema.description != 'boring')].sort\_values(by='rating', ascending=False)

    return df

* Snapshot of the same code above for readability purposes



import pandas as pd

def average\_selling\_price(prices: pd.DataFrame, units\_sold: pd.DataFrame) -> pd.DataFrame:

    if units\_sold.empty:

        return units\_sold

    else:

        units\_sold['price'] = np.where((units\_sold['purchase\_date']>= prices['start\_date']) & (units\_sold['purchase\_date'] <= prices['end\_date']), prices.price, 0)

        units\_sold['total\_price'] = units\_sold['units'] \* units\_sold['price']

        print(units\_sold)

        units\_sold['total\_units'] =  units\_sold.groupby('product\_id')['units'].sum()

        #units\_sold['total\_units'] =  units\_sold['units'].sum()

        dfp = units\_sold[['product\_id','total\_price']]

        dfu = units\_sold.groupby('product\_id')['units'].sum().reset\_index()

        print(units\_sold)

        print(dfp)

        print(dfu)

        df = dfp[['product\_id', 'total\_price']].groupby('product\_id')['total\_price'].sum().reset\_index()

        df1 = dfu[['product\_id', 'units']].groupby('product\_id')['units'].sum().reset\_index()

        #df = units\_sold[['product\_id', 'total\_price']].groupby('product\_id')['total\_price'].sum().reset\_index()

        #df1 = units\_sold[['product\_id', 'total\_units']].groupby('product\_id')['total\_units'].sum().reset\_index()

        df3 = df.merge(df1, how = 'inner', on = 'product\_id')

        df3['average\_price'] = (df3['total\_price']/df3['units']).round(2)

        print(df3)

        return df3[['product\_id', 'average\_price']]