DESCRIPTION

Table: Transactions

```
+---------------+---------+
| Column Name   | Type    |
+---------------+---------+
| id            | int     |
| country       | varchar |
| state         | enum    |
| amount        | int     |
| trans_date    | date    |
+---------------+---------+
```

id is the primary key of this table.

The table has information about incoming transactions.

The state column is an enum of type ["approved", "declined"].

Write an SQL query to find for each month and country, the number of transactions and their total amount, the number of approved transactions and their total amount.

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

The query result format is in the following example.

Example 1:

Input:

Transactions table:

```
+------+---------+----------+--------+------------+
| id   | country | state    | amount | trans_date |
+------+---------+----------+--------+------------+
| 121  | US      | approved | 1000   | 2018-12-18 |
| 122  | US      | declined | 2000   | 2018-12-19 |
| 123  | US      | approved | 2000   | 2019-01-01 |
| 124  | DE     | approved | 2000   | 2019-01-07 |
+------+---------+----------+--------+------------+
```

Output:

```
+----------+---------+-------------+----------------+--------------------+-----------------------+
| month    | country | trans_count | approved_count | trans_total_amount | approved_total_amount |
+----------+---------+-------------+----------------+--------------------+-----------------------+
| 2018-12  | US      | 2           | 1              | 3000               | 1000                  |
| 2019-01  | US      | 1           | 1              | 2000               | 2000                  |
| 2019-01  | DE      | 1           | 1              | 2000               | 2000                  |
+----------+---------+-------------+----------------+--------------------+-----------------------+
```

SOLUTION

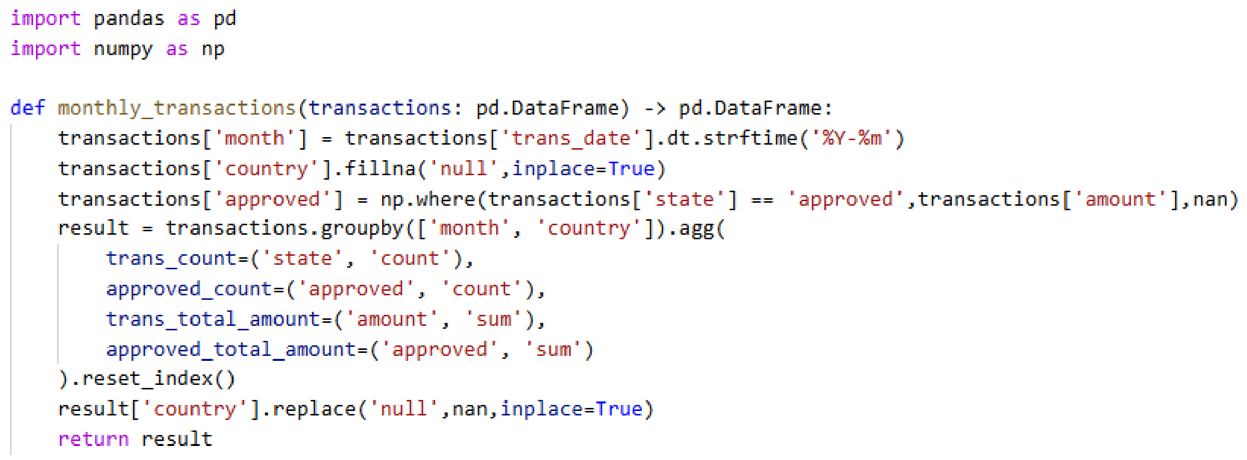
Option 1:

- Extract ‘year’ and ‘month from ‘trans_date’ using dt.strftime()
- Replace null values in ‘country’ with ‘null’ string using fillna()
- Add the ‘approved’ column using np.where and replace ‘unapproved’ transactions with np.nan
- Construct the result dataframe with the desired columns using groupby() and agg() in which ‘count’ and ‘sum’ functions are used to calculate transaction counts and total amounts
- Replace ‘null’ string literals with np.nan

```python
import pandas as pd
import numpy as np

def monthly_transactions(transactions: pd.DataFrame) -> pd.DataFrame:
    transactions['month'] = transactions['trans_date'].dt.strftime('%Y-%m')
    transactions['country'].fillna('null',inplace=True)
    transactions['approved'] = np.where(transactions['state'] == 'approved',transactions['amount'],nan)
    result = transactions.groupby(['month', 'country']).agg(
        trans_count=('state', 'count'),
        approved_count=('approved', 'count'),
        trans_total_amount=('amount', 'sum'),
        approved_total_amount=('approved', 'sum')
    ).reset_index()
    result['country'].replace('null',nan,inplace=True)
    return result
```

- Snapshot of the same code above for readability purposes



```python
import pandas as pd
import numpy as np

def monthly_transactions(transactions: pd.DataFrame) -> pd.DataFrame:
    transactions['month'] = transactions['trans_date'].dt.strftime('%Y-%m')
    transactions['country'].fillna('null',inplace=True)
    transactions['approved'] = np.where(transactions['state'] == 'approved',transactions['amount'],nan)
    result = transactions.groupby(['month', 'country']).agg(
        trans_count=('state', 'count'),
        approved_count=('approved', 'count'),
        trans_total_amount=('amount', 'sum'),
        approved_total_amount=('approved', 'sum')
    ).reset_index()
    result['country'].replace('null',nan,inplace=True)
    return result
```

Option 2:

- Similar approach as above using pd.merge instead of agg()

```python
import pandas as pd
import numpy as np

def monthly_transactions(transactions: pd.DataFrame) -> pd.DataFrame:
    transactions['month'] = transactions['trans_date'].dt.strftime('%Y-%m')
    transactions['approved'] = (transactions['state']=='approved')
    transactions['approved_amount'] = transactions['approved']*transactions['amount']
    trans_count = transactions.groupby(['month', 'country'],
dropna=False)['id'].count().reset_index(name='trans_count')
    approved_count = transactions.groupby(['month', 'country'],
dropna=False)['approved'].sum().reset_index(name='approved_count')
    trans_total_amount = transactions.groupby(['month', 'country'],
dropna=False)['amount'].sum().reset_index(name='trans_total_amount')
    approved_total_amount = transactions.groupby(['month', 'country'],
dropna=False)['approved_amount'].sum().reset_index(name='approved_total_amount')
    merged1 = pd.merge(trans_count, approved_count, how='inner', on=['month', 'country'])
    merged2 = pd.merge(trans_total_amount, approved_total_amount, how='inner', on=['month', 'country'])
    result = pd.merge(merged1, merged2, how='inner', on=['month', 'country'])
    return result
```

- Snapshot of the same code above for readability purposes

```
import pandas as pd
import numpy as np

def monthly_transactions(transactions: pd.DataFrame) -> pd.DataFrame:
    transactions['month'] = transactions['trans_date'].dt.strftime('%Y-%m')
    transactions['approved'] = (transactions['state']=='approved')
    transactions['approved_amount'] = transactions['approved']*transactions['amount']
    trans_count = transactions.groupby(['month', 'country'], dropna=False)['id'].count().reset_index
(name='trans_count')
    approved_count = transactions.groupby(['month', 'country'], dropna=False)['approved'].sum().
reset_index(name='approved_count')
    trans_total_amount = transactions.groupby(['month', 'country'], dropna=False)['amount'].sum().
reset_index(name='trans_total_amount')
    approved_total_amount = transactions.groupby(['month', 'country'], dropna=False)
['approved_amount'].sum().reset_index(name='approved_total_amount')
    merged1 = pd.merge(trans_count, approved_count, how='inner', on=['month', 'country'])
    merged2 = pd.merge(trans_total_amount, approved_total_amount, how='inner', on=['month',
'country'])
    result = pd.merge(merged1, merged2, how='inner', on=['month', 'country'])
    return result
```