Process for Creating Transaction Table

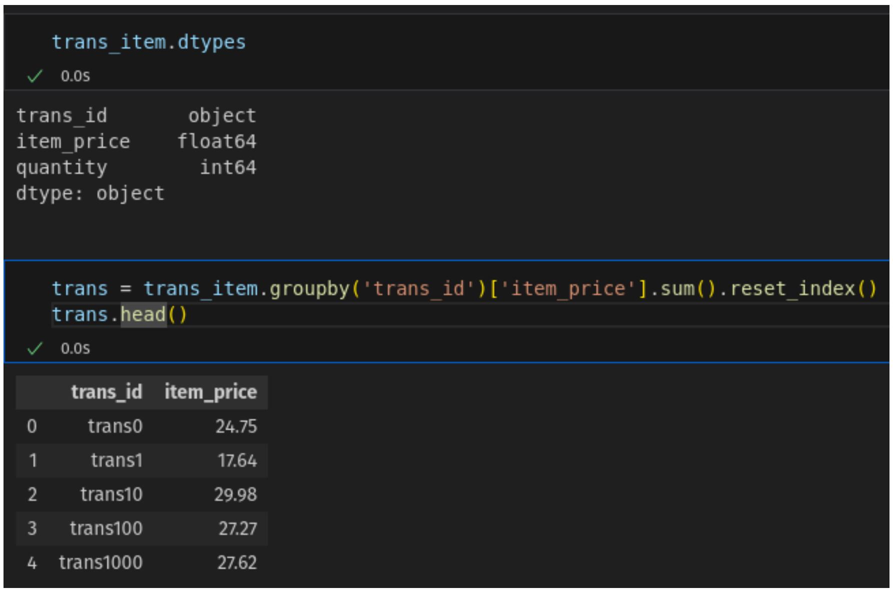
Preliminary Plan:

- Read in csv. Drop all columns except for transaction_id, item_price, and transaction_total.
- Calculate Transaction_total by first multiplying item price by 2 if the quantity is 2.
- Then I can group by transaction_id and calculate the sum of item_price. This groupby object will become the new table.
- The next step is to read in the customer csv and randomly assign customer IDs to each of the transactions.
- The final step will be to assign a random time of day between 9am and 8pm. This process is one that I don't have much initial experience with. I speculate that I will need to

What I had trouble with:

<u>Groupby trans_id:</u>

The problem:



```
trans_item.dtypes
✓ 0.0s

trans_id       object
item_price    float64
quantity        int64
dtype: object
```

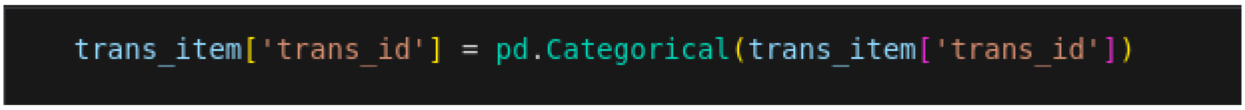
```
trans = trans_item.groupby('trans_id')['item_price'].sum().reset_index()
trans.head()
✓ 0.0s
```

	trans_id	item_price
0	trans0	24.75
1	trans1	17.64
2	trans10	29.98
3	trans100	27.27
4	trans1000	27.62

- Python wasn't grouping trans_id by its unique values. trans_id was being grouped as follows: trans0, trans1, trans10, trans100, trans1000.

- To troubleshoot the problem I looked for null values, extra white space, and examined the datatype. I found no Null values or whitespace, and an analysis of the original csv containing the data revealed that there were no issues. The datatype for trans_id was *object.*

- I tried using the pd.Categorial function to transform the data type to Categorical as online research suggested, using the following code, but the groupby method continued to incorrectly aggregate my trans_ids.



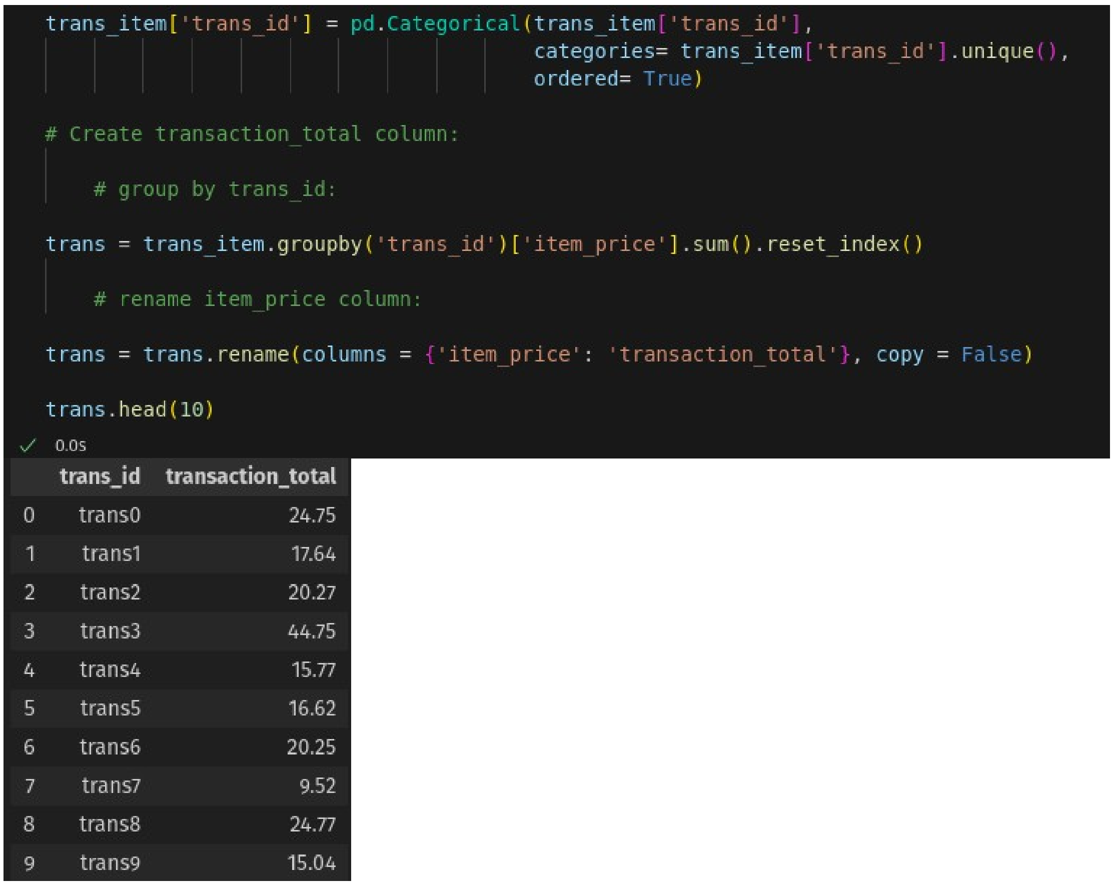
```
trans_item['trans_id'] = pd.Categorical(trans_item['trans_id'])
```

- Using the .dtypes() and .unique() method after attempting to change the dtype to categorical revealed that the data type had been indeed been changed to *Categorical*, however, the categories were trans0, trans1, trans10, trans100, trans1000.

- I returned to the pd.Categorical documentation and read that the user can specify the unique categories in the data. Even though it specified that if no categories were specified, the unique values of the data fed to the pd.Categorical function would be used as the categories, I decided to go ahead and specify that the categories should be the unique trans_ids since the .unique() method revealed that the categories assigned were incorrect anyway.

- After checking the datatype for trans_id had been changed to categorical using the .dtypes DataFrame method, I reran the groupby code I was using to aggregate the unique values of trans_id and they were grouped correctly. I verified the output was correct by using the calculator on my phone to find the sum of a few transactions.

- Here is the code and output:



```
trans_item['trans_id'] = pd.Categorical(trans_item['trans_id'],
                                        categories= trans_item['trans_id'].unique(),
                                        ordered= True)

# Create transaction_total column:

    # group by trans_id:

trans = trans_item.groupby('trans_id')['item_price'].sum().reset_index()

    # rename item_price column:

trans = trans.rename(columns = {'item_price': 'transaction_total'}, copy = False)

trans.head(10)
```

✓ 0.0s

	trans_id	transaction_total
0	trans0	24.75
1	trans1	17.64
2	trans2	20.27
3	trans3	44.75
4	trans4	15.77
5	trans5	16.62
6	trans6	20.25
7	trans7	9.52
8	trans8	24.77
9	trans9	15.04

<u>Generating random times</u>