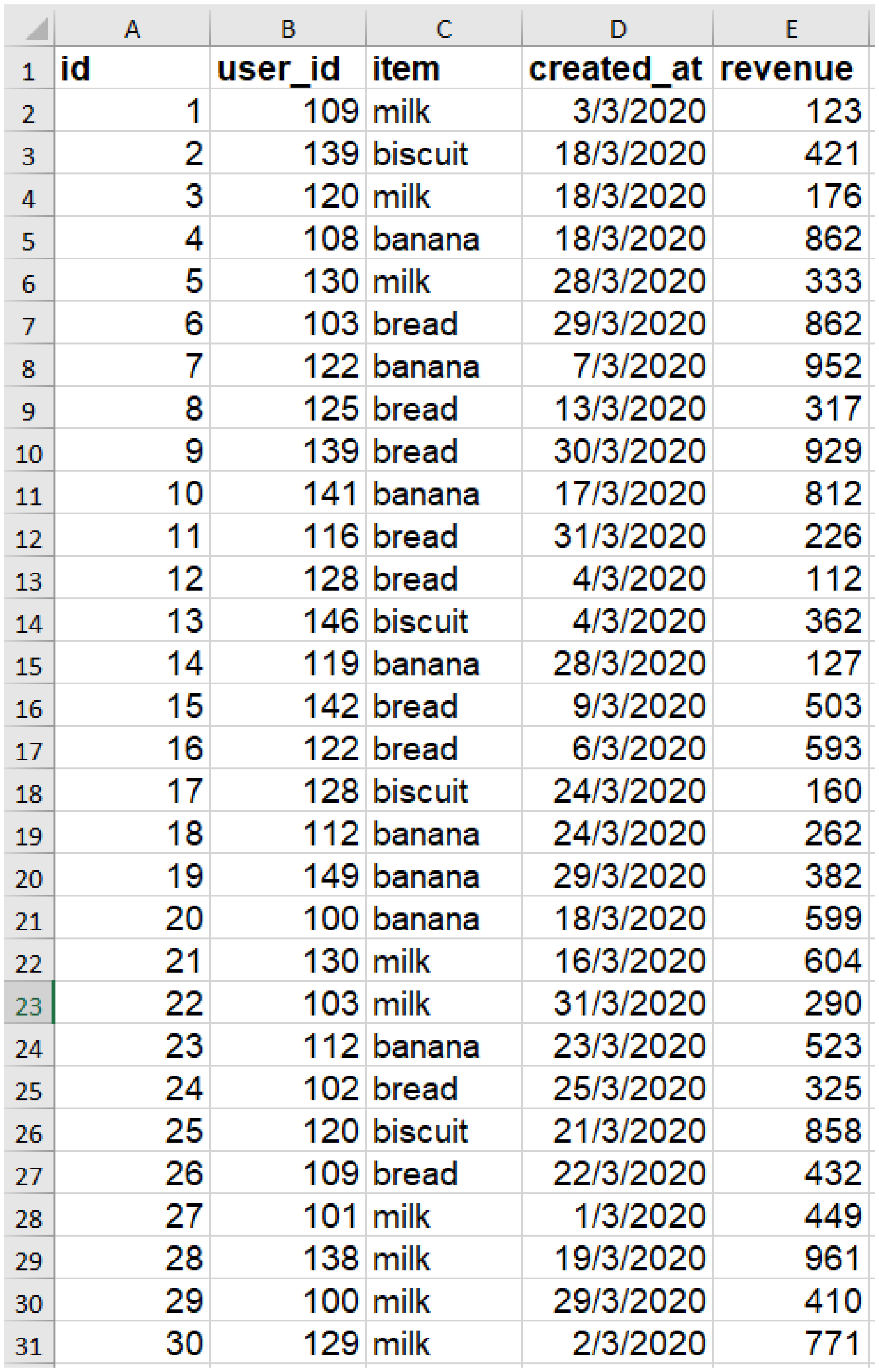
Visualisation of Daily Revenue and Purchases

Given the following data, visualise:

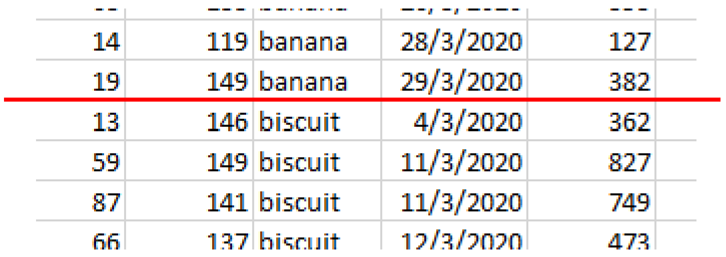
a) The daily trend of revenue by item purchased
b) The number of purchases by item purchased and day of week



	A	B	C	D	E
1	id	user_id	item	created_at	revenue
2	1	109	milk	3/3/2020	123
3	2	139	biscuit	18/3/2020	421
4	3	120	milk	18/3/2020	176
5	4	108	banana	18/3/2020	862
6	5	130	milk	28/3/2020	333
7	6	103	bread	29/3/2020	862
8	7	122	banana	7/3/2020	952
9	8	125	bread	13/3/2020	317
10	9	139	bread	30/3/2020	929
11	10	141	banana	17/3/2020	812
12	11	116	bread	31/3/2020	226
13	12	128	bread	4/3/2020	112
14	13	146	biscuit	4/3/2020	362
15	14	119	banana	28/3/2020	127
16	15	142	bread	9/3/2020	503
17	16	122	bread	6/3/2020	593
18	17	128	biscuit	24/3/2020	160
19	18	112	banana	24/3/2020	262
20	19	149	banana	29/3/2020	382
21	20	100	banana	18/3/2020	599
22	21	130	milk	16/3/2020	604
23	22	103	milk	31/3/2020	290
24	23	112	banana	23/3/2020	523
25	24	102	bread	25/3/2020	325
26	25	120	biscuit	21/3/2020	858
27	26	109	bread	22/3/2020	432
28	27	101	milk	1/3/2020	449
29	28	138	milk	19/3/2020	961
30	29	100	milk	29/3/2020	410
31	30	129	milk	2/3/2020	771

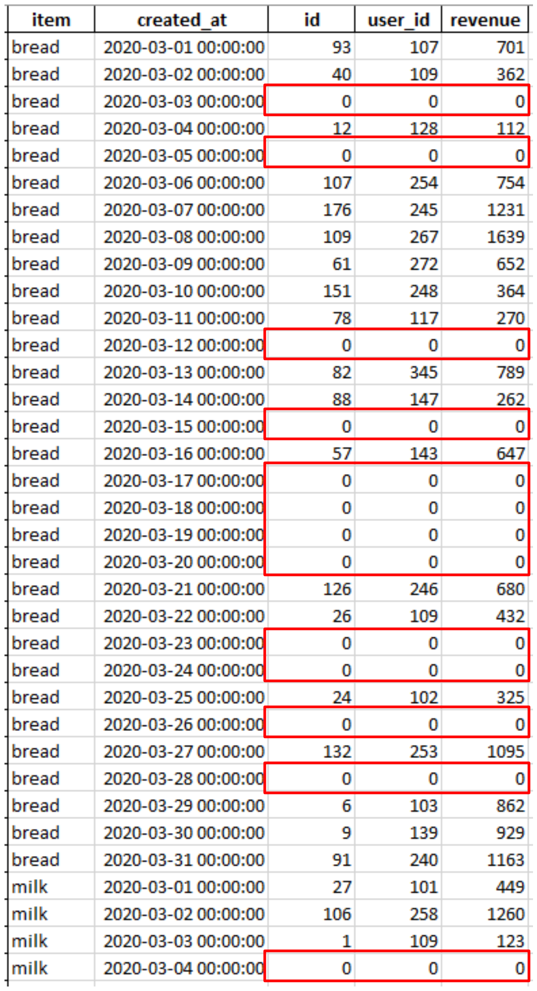
Daily trend of revenue by item purchased

The data provided only shows records where a purchase was made. For example, it does not contain any records of biscuit purchase on 1 March 2020.



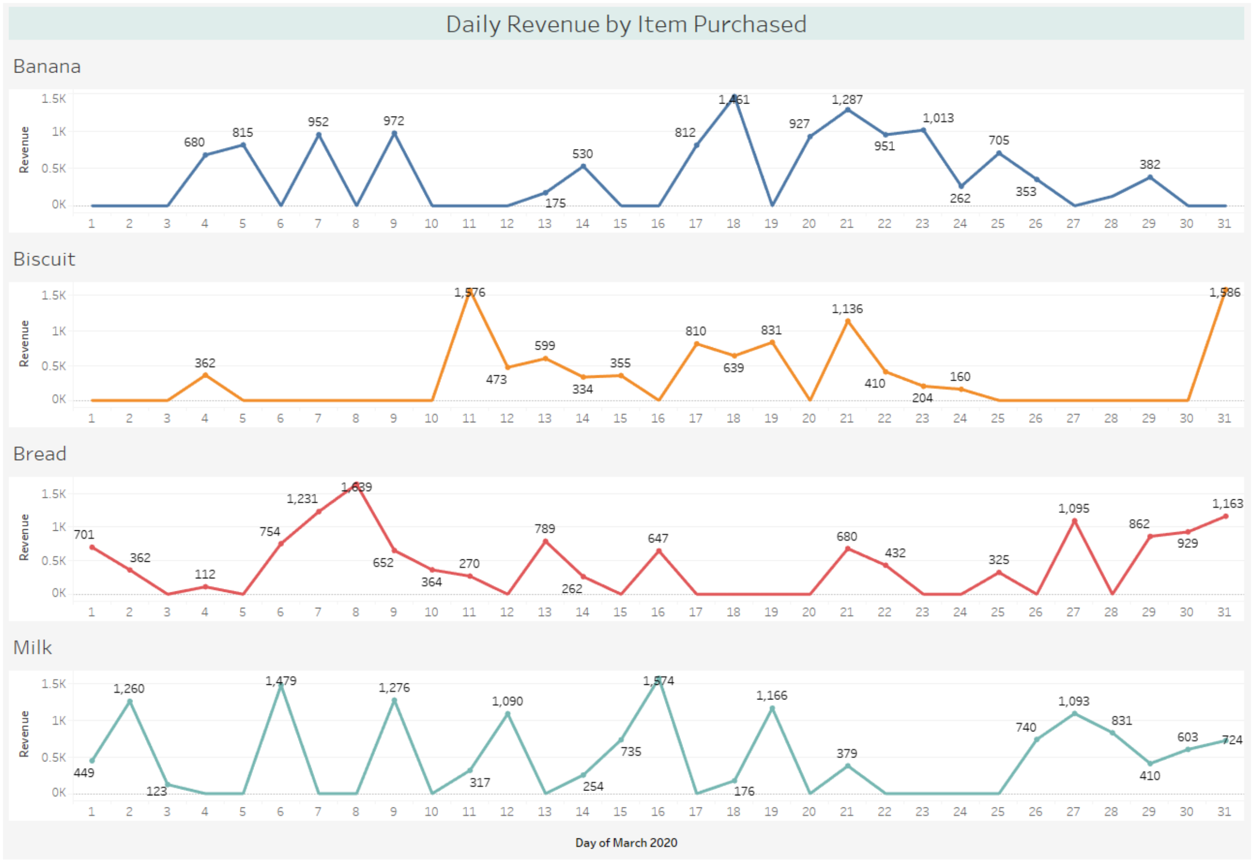
14	119	banana	28/3/2020	127
19	149	banana	29/3/2020	382
13	146	biscuit	4/3/2020	362
59	149	biscuit	11/3/2020	827
87	141	biscuit	11/3/2020	749
66	137	biscuit	12/3/2020	473

Using Python Pandas library, I have created new rows even when a purchase was not made and filled up the null values with 0 to represent 0 revenue (see screenshot below). This is so that we can plot a realistic and continuous graph on Tableau, showing 0 revenue when an item purchase was not made on specific days. Please refer to **Annex A** below for the Python code.



item	created_at	id	user_id	revenue
bread	2020-03-01 00:00:00	93	107	701
bread	2020-03-02 00:00:00	40	109	362
bread	2020-03-03 00:00:00	0	0	0
bread	2020-03-04 00:00:00	12	128	112
bread	2020-03-05 00:00:00	0	0	0
bread	2020-03-06 00:00:00	107	254	754
bread	2020-03-07 00:00:00	176	245	1231
bread	2020-03-08 00:00:00	109	267	1639
bread	2020-03-09 00:00:00	61	272	652
bread	2020-03-10 00:00:00	151	248	364
bread	2020-03-11 00:00:00	78	117	270
bread	2020-03-12 00:00:00	0	0	0
bread	2020-03-13 00:00:00	82	345	789
bread	2020-03-14 00:00:00	88	147	262
bread	2020-03-15 00:00:00	0	0	0
bread	2020-03-16 00:00:00	57	143	647
bread	2020-03-17 00:00:00	0	0	0
bread	2020-03-18 00:00:00	0	0	0
bread	2020-03-19 00:00:00	0	0	0
bread	2020-03-20 00:00:00	0	0	0
bread	2020-03-21 00:00:00	126	246	680
bread	2020-03-22 00:00:00	26	109	432
bread	2020-03-23 00:00:00	0	0	0
bread	2020-03-24 00:00:00	0	0	0
bread	2020-03-25 00:00:00	24	102	325
bread	2020-03-26 00:00:00	0	0	0
bread	2020-03-27 00:00:00	132	253	1095
bread	2020-03-28 00:00:00	0	0	0
bread	2020-03-29 00:00:00	6	103	862
bread	2020-03-30 00:00:00	9	139	929
bread	2020-03-31 00:00:00	91	240	1163
milk	2020-03-01 00:00:00	27	101	449
milk	2020-03-02 00:00:00	106	258	1260
milk	2020-03-03 00:00:00	1	109	123
milk	2020-03-04 00:00:00	0	0	0

Next, I plotted the daily revenue for each item on tableau and combined them into a one chart.



From the above chart, we can note that the revenue brought by the purchase of each item was not constant throughout the month. It follows cycles of high-to-low and low-to-high revenue.

Banana
Banana brought about **higher revenues at the start and end of the month (from 4 Mar 2020 to 9 Mar 2020 and from 17 Mar 2020 to 25 Mar 2020)**. Revenue was observed to be relatively lower in other periods of the month.

We can also observe that almost every other day, the **revenue would pick up for 1-2 days before dropping for 1-2 days. Revenue peaked every 2-3 days**.

Warehouse team may wish to stock up on bananas every 2 to 3 days, and avoid stocking up too many bananas on periods with lower revenue.

Biscuit
Biscuit brought about **higher revenues at the middle of March 2020** (from 11 Mar 2020 to 22 Mar 2020), whereas revenue was observed to be low at the start and end of the month.

In fact, it can be observed that there was **0 demand for biscuit from 5 Mar 2020 to 10 Mar 2020 and from 25 Mar 2020 to 30 Mar 2020**.

Warehouse team may wish to stock up on biscuits in the middle of the month, and avoid stocking up too many biscuits during other periods. Sales team can explore promotions/discounts to increase sales of biscuits at the start and end of the month. They may also wish to dig deeper into why sales of biscuits were low during these periods.

Bread
Revenue peaked every 4-5 days.

Warehouse team may wish to stock up on breads every 4-5 days.

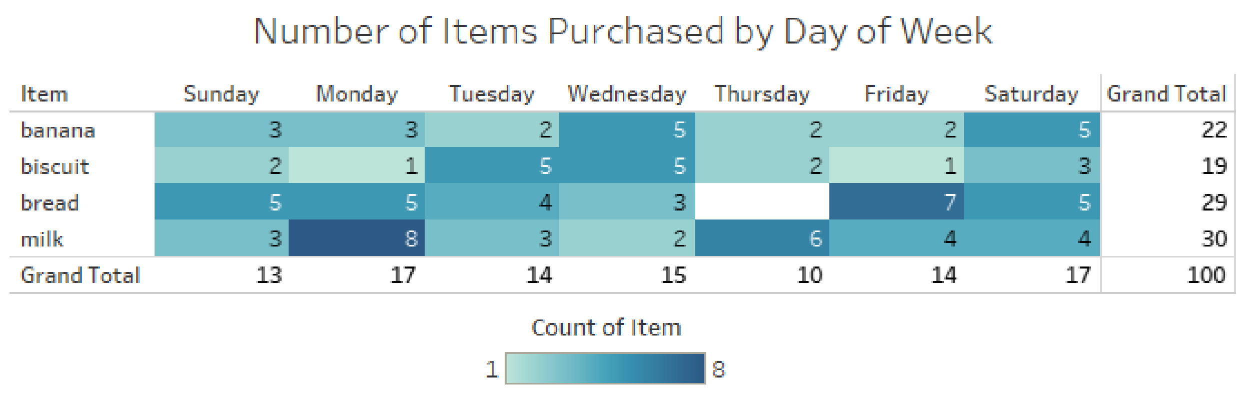
Milk
There appears to be a relatively regular cyclical pattern of high and low revenues for milk, **where revenue peaks every 3-4 days**. Revenue appears to be low from 20 Mar 2020 to 25 Mar 2020.

Warehouse team may wish to stock up on milk every 3-4 days. Sales team may wish to look into the reason for low milk sales from 20 Mar 2020 to 25 Mar 2020.

<u>Number of purchases by item purchased and day of week</u>
Using a highlight table, we can visualise the sales of items by the day of week.

It can be observed that the **sale of biscuit is weak on Monday and Friday** and the **sale of bread is weak on Thursday**. The sales team may wish to explore promotions or discounts to boost sales on these days.

On the other hand, **demand for bread and milk is high on Friday and Monday respectively**. The warehouse team may want to ensure sufficient stock the day before.



Number of Items Purchased by Day of Week

Item	Sunday	Monday	Tuesday	Wednesday	Thursday	Friday	Saturday	Grand Total
banana	3	3	2	5	2	2	5	22
biscuit	2	1	5	5	2	1	3	19
bread	5	5	4	3		7	5	29
milk	3	8	3	2	6	4	4	30
Grand Total	13	17	14	15	10	14	17	100

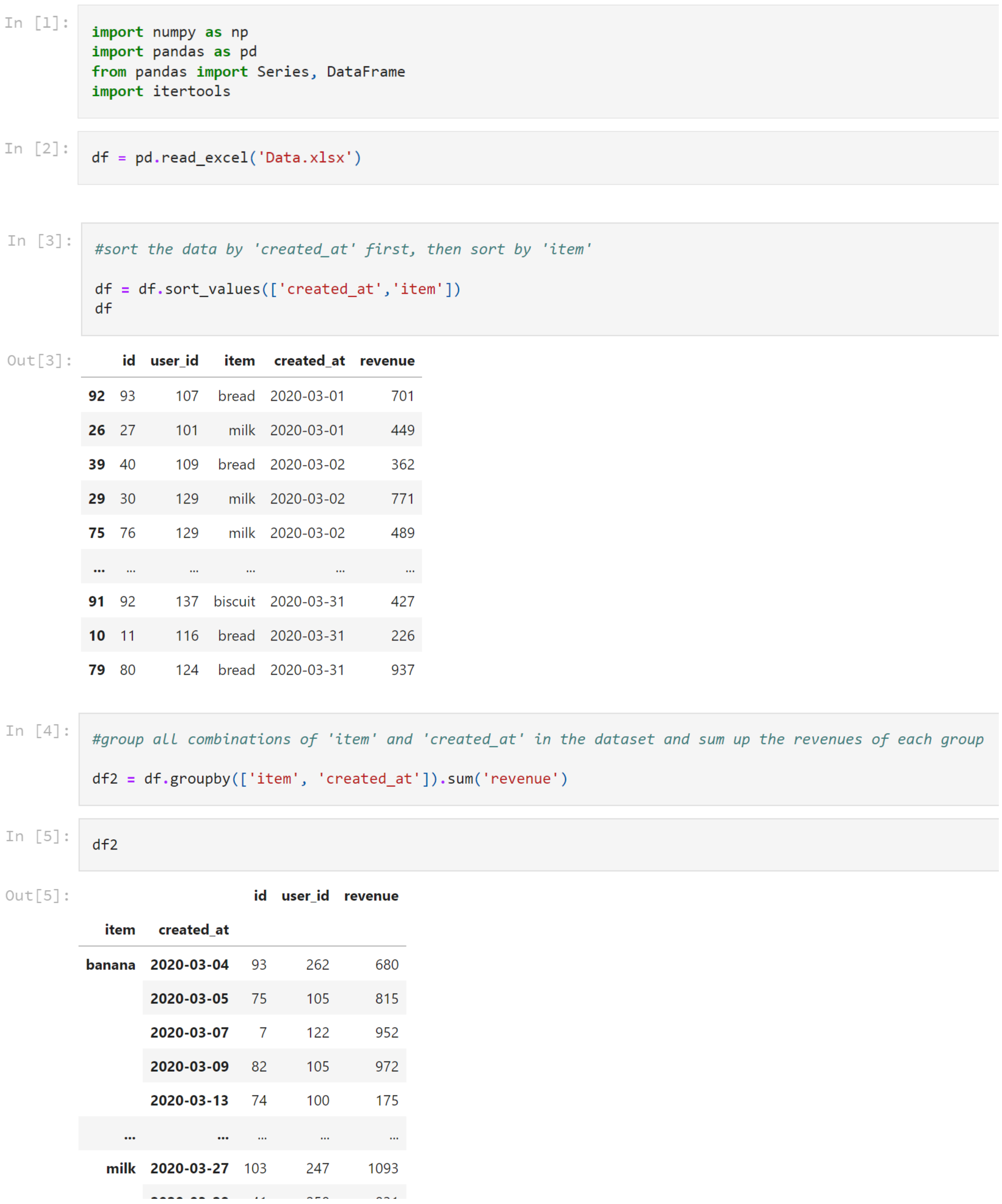
Count of Item
1
8

For both tasks a) and b), **it is important that we also analyse past months' data to ascertain if the observed trends persist**.

For task b), it is important to note that while **there are 5 Sundays, Mondays and Tuesdays in March 2020**, **there are only 4 Wednesdays, Thursdays, Fridays and Saturdays in March 2020**. We should **normalise** the data for a better understanding of how the day of week affects the number of items purchased.

Annex A

Python Code in Jupyter Notebook



In [1]:
```
import numpy as np
import pandas as pd
from pandas import Series, DataFrame
import itertools
```

In [2]:
```
df = pd.read_excel('Data.xlsx')
```

In [3]:
```
#sort the data by 'created_at' first, then sort by 'item'

df = df.sort_values(['created_at','item'])
df
```

Out[3]:

	id	user_id	item	created_at	revenue
92	93	107	bread	2020-03-01	701
26	27	101	milk	2020-03-01	449
39	40	109	bread	2020-03-02	362
29	30	129	milk	2020-03-02	771
75	76	129	milk	2020-03-02	489
...	...	...	...	...	...
91	92	137	biscuit	2020-03-31	427
10	11	116	bread	2020-03-31	226
79	80	124	bread	2020-03-31	937

In [4]:
```
#group all combinations of 'item' and 'created_at' in the dataset and sum up the revenues of each group

df2 = df.groupby(['item', 'created_at']).sum('revenue')
```

In [5]:
```
df2
```

Out[5]:

		id	user_id	revenue
item	**created_at**			
banana	**2020-03-04**	93	262	680
	2020-03-05	75	105	815
	2020-03-07	7	122	952
	2020-03-09	82	105	972
	2020-03-13	74	100	175
...	**...**	...	...	...
milk	**2020-03-27**	103	247	1093

In [6]:

```
#call itertools.product() to find all possible combinations of 'item' for all 31 days of March 2020
#reindex each combination as a single row
#for rows without values, fill it up with the value, 0

df3 = df2.reindex(itertools.product(df['item'].unique(),
                                    df['created_at'].unique(),)
                 ).fillna(0).reset_index()
```

In [7]:

```
df3
```

Out[7]:

	item	created_at	id	user_id	revenue
0	bread	2020-03-01	93.0	107.0	701.0
1	bread	2020-03-02	40.0	109.0	362.0
2	bread	2020-03-03	0.0	0.0	0.0
3	bread	2020-03-04	12.0	128.0	112.0
4	bread	2020-03-05	0.0	0.0	0.0
...	...	...	...	...	...
119	biscuit	2020-03-27	0.0	0.0	0.0

In [9]:

```
#output the file to excel format

df3.to_excel("output.xlsx")
```