これでクレンジングと整形まで終わったデータが無事ダンプされました。

元のデータに変更があった場合は、再度データ加工処理を行い直す必要がありますが、元のデータが変わらない限りは、このファイルを読み込んで分析する事ができます。

さっそくダンプファイルを読み込んで、集計を行ってみましょう。

ノック20：データを集計しよう

まずは、ダンプファイルを読み込みます。

```
import_data = pd.read_csv("dump_data.csv")
import_data
```

図2-26：インポート結果

```
In [27]: import_data = pd.read_csv("dump_data.csv")
         import_data
```

Out[27]:

	purchase_date	purchase_month	item_name	item_price	顧客名	かな	地域	メールアドレス	登録日
0	2019-06-13 10:02:34	201906	商品A	100.0	深井菜々美	ふかいななみ	C市	fukai_nanami@example.com	2017-01-26 00:00:00
1	2019-07-13 13:05:29	201907	商品S	1900.0	浅田賢二	あさだ けんじ	C市	asada_kenji@example.com	2018-04-07 00:00:00
2	2019-05-11 19:42:07	201905	商品A	100.0	南部慶二	なんぶ けいじ	A市	nannbu_keiji@example.com	2018-06-19 00:00:00
3	2019-02-12 23:40:45	201902	商品Z	2600.0	原生昭雄	あそう りお	D市	asou_rio@example.com	2018-07-22 00:00:00
4	2019-04-22 03:09:35	201904	商品A	100.0	平田鉄二	ひらた てつじ	D市	hirata_tetsuji@example.com	2017-06-07 00:00:00
5	2019-03-20 19:13:01	201903	商品S	1900.0	堀江治	ほりえ たすく	H市	horie_tasuku@example.com	2018-05-14 00:00:00
6	2019-05-18 19:18:53	201905	商品A	100.0	深井照生	ふかいてるお	A市	fukai_teruo@example.com	2018-02-21 00:00:00

もはや説明するまでもありませんが、read_csvにてダンプデータを読み込んでいます。

続いて、purchase_monthを縦軸に、商品毎の集計を行いましょう。

```
byItem = import_data.pivot_table(index="purchase_month", columns="item_name", aggfunc="size", fill_value=0)
byItem
```


図2-27：購入年月、商品の集計結果

```
In [28]: byItem = import_data.pivot_table(index="purchase_month", columns="item_name", aggfunc="size", fill_value=0)
         byItem
```

Out[28]:

item_name	商品A	商品B	商品C	商品D	商品E	商品F	商品G	商品H	商品I	商品J	...	商品Q	商品R	商品S	商品T	商品U	商品V	商品W	商品X	商品Y	商品Z
purchase_month																					
201901	18	13	19	17	18	15	11	16	18	17	...	17	21	20	17	7	22	13	14	10	0
201902	19	14	26	21	16	14	14	17	12	14	...	22	22	22	23	19	22	24	16	11	1
201903	17	21	20	17	9	27	14	18	12	16	...	23	16	20	12	23	18	16	21	16	0
201904	17	19	24	20	18	17	14	11	18	13	...	20	20	16	16	11	15	14	16	20	0
201905	24	14	16	14	19	18	23	15	16	11	...	13	22	15	16	16	9	21	16	20	0
201906	24	12	11	19	13	15	15	13	19	22	...	15	16	21	12	18	20	17	15	13	0
201907	20	20	17	17	12	17	19	19	19	23	...	15	19	23	21	13	28	16	18	12	0

7 rows × 26 columns

下部に「26 columns」とあるように、正しく商品A～Zの26商品毎の購入年月の集計が行えました。

続いて、purchase_monthを縦軸に、売上金額、顧客、地域の集計を行いましょう。

```
byPrice = import_data.pivot_table(index="purchase_month", columns="item_name", values="item_price", aggfunc="sum", fill_value=0)
byPrice
```

図2-28：購入年月、売上金額の集計結果

```
In [29]: byPrice = import_data.pivot_table(index="purchase_month", columns="item_name", values="item_price", aggfunc="sum", fill_value=0)
         byPrice
```

Out[29]:

item_name	商品A	商品B	商品C	商品D	商品E	商品F	商品G	商品H	商品I	商品J	...	商品Q	商品R	商品S	商品T	商品U	商品V	商品W	商品X	商品Y	商
purchase_month																					
201901	1800	2600	5700	6800	9000	9000	7700	12800	18200	17000	...	28900	37800	38000	34000	14700	48400	29900	33600	25000	
201902	1900	2800	7800	8400	8000	8400	9800	13600	10500	14000	...	37400	39800	41800	46000	39900	48400	55200	38400	27500	26
201903	1700	4200	6000	6800	4500	16200	9800	14400	10800	16000	...	39100	28800	38000	24000	48300	39600	36800	50400	40000	
201904	1700	3800	7200	8000	9000	10200	9800	8800	16200	13000	...	34000	36000	30400	32000	23100	33000	32200	38400	50000	
201905	2400	2800	4800	5600	9500	10800	16100	12000	14400	11000	...	22100	39600	34200	32000	33600	19800	48300	38400	50000	
201906	2400	2400	3300	7600	6500	10800	10500	10400	17100	22000	...	25500	28800	39900	24000	37800	44000	39100	36000	32500	
201907	2000	4000	5100	6800	6000	10200	13300	15200	17100	23000	...	25500	34200	43700	42000	27300	61600	36800	43200	30000	

7 rows × 26 columns

```
byCustomer = import_data.pivot_table(index="purchase_month", columns="顧客名", aggfunc="size", fill_value=0)
byCustomer
```