■図2-24：結合結果

```
In [24]: join_data = pd.merge(uriage_data, kokyaku_data, left_on="customer_name", right_on="顧客名", how="left")
         join_data = join_data.drop("customer_name", axis=1)
         join_data
```

Out[24]:

	purchase_date	item_name	item_price	purchase_month	顧客名	かな	地域	メールアドレス	登録日	登録年月
0	2019-06-13 18:02:34	商品A	100.0	201906	深井菜々美	ふかい ななみ	C市	fukai_nanami@example.com	2017-01-26	201701
1	2019-07-13 13:05:29	商品S	1900.0	201907	浅田賢二	あさだ けんじ	C市	asada_kenji@example.com	2018-04-07	201804
2	2019-05-11 19:42:07	商品A	100.0	201905	南部慶二	なんぶ けいじ	A市	nannbu_keiji@example.com	2018-06-19	201806
3	2019-02-12 23:40:45	商品Z	2600.0	201902	麻生莉緒	あそう りお	D市	asou_rio@example.com	2018-07-22	201807
4	2019-04-22 03:09:35	商品A	100.0	201904	平田鉄二	ひらた てつじ	D市	hirata_tetsuji@example.com	2017-06-07	201706
5	2019-03-20 19:16:01	商品S	1900.0	201903	堀江佑	ほりえ たすく	H市	horie_tasuku@example.com	2018-05-14	201805
6	2019-05-18 19:16:53	商品A	100.0	201905	深井照生	ふかい てるお	A市	fukai_teruo@example.com	2018-02-21	201802

left_on、right_onで、結合するキーとなるデータを指定します。

left_onには引数最初に指定したuriage_dataに含まれるキー候補を記載します。

right_onには次に指定したkokyaku_dataに含まれるキー候補を記載します。

howは結合方法で「left」を指定しました。これはuriage_dataを主として、kokyaku_dataを副として結合するという意味になります。

データの加工により、分析に適したデータの形になってきたと思います。

このようなデータ加工を「**クレンジング**」と呼ぶ事もあります。

ノック19：クレンジングしたデータをダンプしよう

データの補正処理を実施し、綺麗に整えたデータができましたが、ここでプログラムを中断してしまうと、またすべての処理をやり直す必要があります。

目的は「データ加工・クレンジング」ではなく「データ分析」です。

綺麗になったデータをファイルに出力(ダンプ)して、分析をする際は出力ファイルから読み込み分析を行う事で、クレンジングのやり直しを省略する事ができます。

さて、さっそくCSVにダンプしてみましょう。

……と言いたいところですが、ダンプする前に最後の調整を行いましょう。

クレンジングされたデータの列の並び順がpurchase_date、item_name、item_price、purchase_month、顧客名、(略)となっています。

purchase_dateとpurchase_monthは近くにあった方がデータとして分かりやすくなります。

そのため、列の配置を調整してから、ファイルに吐き出した方が後々データを見たときに直観的に分かりやすくなります。

細かい事ではありますが、実際の現場でもっと膨大なデータを扱う時に苦労しないために、普段から整形癖を付けておく事をお勧めします。

```
dump_data = join_data[["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"]]
dump_data
```

■図2-25：整形結果

```
In [25]: dump_data = join_data[["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"]]
         dump_data
```

Out[25]:

	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 18:02:34	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
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
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
3	2019-02-12 23:40:45	201902	商品Z	2600.0	麻生莉緒	あそう りお	D市	asou_rio@example.com	2018-07-22
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
5	2019-03-20 19:16:01	201903	商品S	1900.0	堀江佑	ほりえ たすく	H市	horie_tasuku@example.com	2018-05-14
6	2019-05-18 19:16:53	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
7	2019-04-18 00:14:21	201904	商品V	2200.0	牧田玲那	まきた れな	A市	maikita_rena@example.com	2017-05-13
8	2019-01-10 15:51:01	201901	商品O	1500.0	堀北雅彦	ほりきた まさひこ	H市	horikita_masahiko@example.com	2017-05-05
9	2019-01-28 10:47:03	201901	商品A	100.0	大島礼子	おおしま れいこ	F市	oojhi_reiko@example.com	2017-05-09

join_dataから必要な列を任意の順番に並び替えています。カラム名を指定するだけでできますので、とっても簡単に整形ができました。

これをdump_dataという変数に格納します。

```
dump_data.to_csv("dump_data.csv", index=False)
```

整形したdump_dataをto_csv()でファイル出力を行います。

この行を実行すると、同じ階層にdump_data.csvが出力されている事が確認できるかと思います。