Projecting and Filtering

Let us understand how to project as well as filter data in Data Frames.


```
%run 06_csv_to_pandas_data_frame.ipynb
```

```
orders
```

	order_id	order_date	order_customer_id	order_status
0	1	2013-07-25 00:00:00.0	11599	CLOSED
1	2	2013-07-25 00:00:00.0	256	PENDING_PAYMENT
2	3	2013-07-25 00:00:00.0	12111	COMPLETE
3	4	2013-07-25 00:00:00.0	8827	CLOSED
4	5	2013-07-25 00:00:00.0	11318	COMPLETE
...	...	...	...	...
68878	68879	2014-07-09 00:00:00.0	778	COMPLETE
68879	68880	2014-07-13 00:00:00.0	1117	COMPLETE
68880	68881	2014-07-19 00:00:00.0	2518	PENDING_PAYMENT
68881	68882	2014-07-22 00:00:00.0	10000	ON_HOLD
68882	68883	2014-07-23 00:00:00.0	5533	COMPLETE

68883 rows × 4 columns

```
order_items
```

	order_item_id	order_item_order_id	order_item_product_id	order_item_quantity	order
0	1	1	957	1	
1	2	2	1073	1	
2	3	2	502	5	
3	4	2	403	1	
4	5	4	897	2	
...	...	...	...	...	
172193	172194	68881	403	1	
172194	172195	68882	365	1	
172195	172196	68882	502	1	
172196	172197	68883	208	1	
172197	172198	68883	502	3	

172198 rows × 6 columns

- Projecting data

```
orders.order_date
```

```
0        2013-07-25 00:00:00.0
1        2013-07-25 00:00:00.0
2        2013-07-25 00:00:00.0
3        2013-07-25 00:00:00.0
4        2013-07-25 00:00:00.0
                ...
68878    2014-07-09 00:00:00.0
68879    2014-07-13 00:00:00.0
68880    2014-07-19 00:00:00.0
68881    2014-07-22 00:00:00.0
68882    2014-07-23 00:00:00.0
Name: order_date, Length: 68883, dtype: object
```

```
orders['order_date']
```

```
0        2013-07-25 00:00:00.0
1        2013-07-25 00:00:00.0
2        2013-07-25 00:00:00.0
3        2013-07-25 00:00:00.0
4        2013-07-25 00:00:00.0
                 ...
68878    2014-07-09 00:00:00.0
68879    2014-07-13 00:00:00.0
68880    2014-07-19 00:00:00.0
68881    2014-07-22 00:00:00.0
68882    2014-07-23 00:00:00.0
Name: order_date, Length: 68883, dtype: object
```

```
# Project order_item_order_id and order_item_subtotal
order_items[['order_item_order_id', 'order_item_subtotal']]
```

	order_item_order_id	order_item_subtotal
0	1	299.98
1	2	199.99
2	2	250.00
3	2	129.99
4	4	49.98
...	...	...
172193	68881	129.99
172194	68882	59.99
172195	68882	50.00
172196	68883	1999.99
172197	68883	150.00

172198 rows × 2 columns

- Filter for order_item_order_id 2

```
order_items.columns
```

```
Index(['order_item_id', 'order_item_order_id', 'order_item_product_id',
       'order_item_quantity', 'order_item_subtotal',
       'order_item_product_price'],
      dtype='object')
```

```
order_items.order_item_order_id == 2
```

```
0         False
1          True
2          True
3          True
4         False
          ...
172193    False
172194    False
172195    False
172196    False
172197    False
Name: order_item_order_id, Length: 172198, dtype: bool
```

```
order_items[order_items.order_item_order_id == 2]
```

	order_item_id	order_item_order_id	order_item_product_id	order_item_quantity	order_item_
1	2	2	1073	1	
2	3	2	502	5	
3	4	2	403	1	

```
order_items['order_item_order_id'] == 2
```

```
order_items[order_items['order_item_order_id'] == 2]
```

```
order_items.query('order_item_order_id == 2')
```

```
order_items[
    (order_items.order_item_order_id == 2) &
    ((order_items.order_item_subtotal >= 150) &
     (order_items.order_item_subtotal <= 250)
    )]
```

Note

String passed to `query` API is broken into multiple lines for readability purposes.

```
order_items.query('order_item_order_id == 2 and ' +
                  'order_item_subtotal >= 150 and ' +
                  'order_item_subtotal <= 250')
```

```
orders[orders.order_date == '2013-08-01 00:00:00.0']
```

```
orders.query('order_date == "2013-08-01 00:00:00.0"')
```

Note

We can use the functions available as part of `str` usng `python` as engine.

```
order_date = '2013-08-01 00:00:00.0'
```

```
order_date.startswith?
```

```
order_date.startswith('2013-08')
```

```
orders.query('order_date.str.startswith("2013-08")', engine='python')
```

Task 1

Get all the orders placed by customer_id

```
orders[:10]
```

```
orders.query('order_customer_id == 12431')
```

```
orders[orders.order_customer_id == 12431]
```

```
orders[orders['order_customer_id'] == 12431]
```

Task 2

Get all the orders placed by customer_id for a given month. Month is passed as **yyyy-MM** format.

```
orders.query('order_customer_id == 12431 and order_date.str.startswith("2014-01")',
engine='python')
```

```
orders[(orders.order_customer_id == 12431) & (orders.order_date.str.startswith('2014-01'))]
```

Task 3

Get all the orders which are placed by customer with id 12431 in January 2014 and status is in PENDING_PAYMENT or PROCESSING

```
orders.query('order_customer_id == 12431 and ' +
             'order_date.str.startswith("2014-01") and ' +
             'order_status in ("PROCESSING", "PENDING_PAYMENT")',
             engine='python'
            )
```

```
orders[(orders.order_customer_id == 12431) &
       (orders.order_date.str.startswith('2014-01')) &
       (orders.order_status.isin(['PROCESSING', 'PENDING_PAYMENT']))
      ]
```

By Durga Gadiraju