Pandas – gym example – add and remove columns

Create a new column with the number of years at the gym and insert it at position 3

```
In [33]: df.insert(3, 'years', 2021 - df['date'].dt.year)
```

Create new columns at last position

```
In [34]: df['days'] = df['years'] * 365
```

```
In [35]: df['old_client'] = df['years'] > 7
```

```
In [36]: df.head(3)
```

Out[36]:

	name	id	date	years	height	weight	age	hours	status	children	sex	days	old_client
0	Marisa Martins	1373913	2013-02-05	8	155	48	45	3	married	2	F	2920	True
1	Rita Fonseca	1109818	2018-08-28	3	166	54	45	3	married	3	F	1095	False
2	Joana Freitas	1158813	2013-10-21	8	150	42	52	3	widow	1	F	2920	True

Remove the columns years, days and old_client

```
In [37]: df.drop('years', axis = 1, inplace = True)
```

```
In [38]: del df['days']
```

```
In [39]: old_client = df.pop('old_client')
```

```
In [40]: df.head(3)
```

Out[40]:

	name	id	date	height	weight	age	hours	status	children	sex
0	Marisa Martins	1373913	2013-02-05	155	48	45	3	married	2	F
1	Rita Fonseca	1109818	2018-08-28	166	54	45	3	married	3	F
2	Joana Freitas	1158813	2013-10-21	150	42	52	3	widow	1	F

Pandas – gym example – column names

Set the column names

```
In [41]: df.columns = ['Name', 'Id', 'Date', 'Height', 'Weight', 'Age', 'Hours', 'Status', 'Children', 'Sex']
```

```
In [42]: df.head(3)
```

Out[42]:

	Name	Id	Date	Height	Weight	Age	Hours	Status	Children	Sex
0	Marisa Martins	1373913	2013-02-05	155	48	45	3	married	2	F
1	Rita Fonseca	1109818	2018-08-28	166	54	45	3	married	3	F
2	Joana Freitas	1158813	2013-10-21	150	42	52	3	widow	1	F

Change the column name using the str replace method

```
In [43]: df.columns = df.columns.str.replace('Age', 'age')
```

```
In [44]: df.head(3)
```

Out[44]:

	Name	Id	Date	Height	Weight	age	Hours	Status	Children	Sex
0	Marisa Martins	1373913	2013-02-05	155	48	45	3	married	2	F
1	Rita Fonseca	1109818	2018-08-28	166	54	45	3	married	3	F
2	Joana Freitas	1158813	2013-10-21	150	42	52	3	widow	1	F

Change the column names using the rename method

```
In [45]: df.rename(columns={'Name':'name', 'Id': 'id'}, inplace=True)
```

```
In [46]: df.head(3)
```

Out[46]:

	name	id	Date	Height	Weight	age	Hours	Status	Children	Sex
0	Marisa Martins	1373913	2013-02-05	155	48	45	3	married	2	F
1	Rita Fonseca	1109818	2018-08-28	166	54	45	3	married	3	F
2	Joana Freitas	1158813	2013-10-21	150	42	52	3	widow	1	F

Change the column names using list comprehension

```
In [47]: df.columns = [x.lower() for x in df.columns]
```

```
In [48]: df.head(3)
```

Out[48]:

	name	id	date	height	weight	age	hours	status	children	sex
0	Marisa Martins	1373913	2013-02-05	155	48	45	3	married	2	F
1	Rita Fonseca	1109818	2018-08-28	166	54	45	3	married	3	F
2	Joana Freitas	1158813	2013-10-21	150	42	52	3	widow	1	F

Pandas – gym example – row values

Assign a new set of values to a row

```
In [49]: df.loc[0] = ['Antonio Gomes',1354891, '2018-05-20',100, 20, 30, 2, 'widow',1, 'M']
         df.head(2)
```

Out[49]:

	name	id	date	height	weight	age	hours	status	children	sex
0	Antonio Gomes	1354891	2018-05-20	100	20	30	2	widow	1	M
1	Rita Fonseca	1109818	2018-08-28	166	54	45	3	married	3	F

Change some of the values in a row

```
In [50]: df.loc[0,['id', 'status']] = [1234567, 'single']
         df.head(2)
```

Out[50]:

	name	id	date	height	weight	age	hours	status	children	sex
0	Antonio Gomes	1234567	2018-05-20	100	20	30	2	single	1	M
1	Rita Fonseca	1109818	2018-08-28	166	54	45	3	married	3	F

Append a row to a DataFrame

```
In [51]: df = df.append({'name':'Xavier Furtado','id':1111111,'height':150},ignore_index=True)
         df.tail(2)
```

Out[51]:

	name	id	date	height	weight	age	hours	status	children	sex
59	Joao Tavares	1769504	2004-06-08	177	85.0	32.0	3.0	married	2.0	M
60	Xavier Furtado	1111111	NaT	150	NaN	NaN	NaN	NaN	NaN	NaN

Delete a row

```
In [52]: df.drop(60, inplace=True)
         df.tail(2)
```

Out[52]:

	name	id	date	height	weight	age	hours	status	children	sex
58	Manuel Freitas	1658815	2015-11-06	170	51.0	57.0	1.0	widow	2.0	M
59	Joao Tavares	1769504	2004-06-08	177	85.0	32.0	3.0	married	2.0	M

Pandas – gym example – filters

Select the customers over 55 years old

```
In [53]: df.loc[df['age'] > 55]
```

Out[53]:

	name	id	date	height	weight	age	hours	status	children	sex
3	Joana Goncalves	1566515	2015-11-16	161	49.0	59.0	2.0	married	2.0	F
7	Rita Cruz	1930916	2016-10-11	168	52.0	56.0	3.0	married	2.0	F
15	Susana Madeira	1436908	2008-09-05	160	49.0	56.0	2.0	divorced	2.0	F
19	Catarina Goncalves	1055806	2006-09-21	168	53.0	59.0	2.0	widow	3.0	F
32	Joana Marinho	1466415	2015-11-26	164	51.0	57.0	1.0	married	3.0	F
43	Catarina Carvalho	1058304	2004-09-25	150	45.0	59.0	2.0	married	4.0	F
58	Manuel Freitas	1658815	2015-11-06	170	51.0	57.0	1.0	widow	2.0	M

Select the customers over 55 years old and weight over 50kg

```
In [54]: df.loc[(df['age'] > 55) & (df['weight'] > 50)]
```

Out[54]:

	name	id	date	height	weight	age	hours	status	children	sex
7	Rita Cruz	1930916	2016-10-11	168	52.0	56.0	3.0	married	2.0	F
19	Catarina Goncalves	1055806	2006-09-21	168	53.0	59.0	2.0	widow	3.0	F
32	Joana Marinho	1466415	2015-11-26	164	51.0	57.0	1.0	married	3.0	F
58	Manuel Freitas	1658815	2015-11-06	170	51.0	57.0	1.0	widow	2.0	M

Select the customers over 55 or under 24 years old. View name and age

```
In [55]: df.loc[(df['age'] > 57) | (df['age'] < 24), ['name', 'age']]
```

Out[55]:

	name	age
3	Joana Goncalves	59.0
16	Francisco Pinho	23.0
19	Catarina Goncalves	59.0
38	Francisco Madeira	23.0
43	Catarina Carvalho	59.0
44	Francisco Carvalho	23.0

Select the customers with status: divorced or widow

```
In [56]: df.loc[df['status'].isin(['divorced','widow'])]
```

Out[56]:

	name	id	date	height	weight	age	hours	status	children	sex
2	Joana Freitas	1158813	2013-10-21	150	42.0	52.0	3.0	widow	1.0	F
10	Manuel Marinho	1417018	2018-06-20	173	57.0	40.0	3.0	divorced	2.0	M
15	Susana Madeira	1436908	2008-09-05	160	49.0	56.0	2.0	divorced	2.0	F
19	Catarina Goncalves	1055806	2006-09-21	168	53.0	59.0	2.0	widow	3.0	F
58	Manuel Freitas	1658815	2015-11-06	170	51.0	57.0	1.0	widow	2.0	M

Select the customers with name Freitas

```
In [57]: df.loc[df['name'].str.contains('Freitas',na=False)]
```

Out[57]:

	name	id	date	height	weight	age	hours	status	children	sex
2	Joana Freitas	1158813	2013-10-21	150	42.0	52.0	3.0	widow	1.0	F
6	Florbela Freitas	1071208	2008-09-26	166	53.0	28.0	5.0	single	0.0	F
11	Joao Freitas	1642316	2016-10-23	165	50.0	54.0	3.0	married	2.0	M
13	Francisco Freitas	1560118	2018-07-15	152	53.0	41.0	3.0	married	2.0	M
24	Catarina Freitas	1105321	2021-10-07	166	70.0	40.0	4.0	single	0.0	F
58	Manuel Freitas	1658815	2015-11-06	170	51.0	57.0	1.0	widow	2.0	M

Pandas – gym example – apply, value_counts()

Apply a function to all the elements of a Series

```
In [58]: df['status'] = df['status'].apply(str.upper)
         df.head(2)
```

Out[58]:

	name	id	date	height	weight	age	hours	status	children	sex
0	Antonio Gomes	1234567	2018-05-20	100	20.0	30.0	2.0	SINGLE	1.0	M
1	Rita Fonseca	1109818	2018-08-28	166	54.0	45.0	3.0	MARRIED	3.0	F

Get the unique elements from a series

```
In [59]: df['status'].unique()
```

```
Out[59]: array(['SINGLE', 'MARRIED', 'WIDOW', 'DIVORCED'], dtype=object)
```

Return the number of counts of unique values

```
In [60]: df['status'].value_counts()
```

```
Out[60]: MARRIED     35
         SINGLE      20
         WIDOW        3
         DIVORCED     2
         Name: status, dtype: int64
```

```
In [61]: df['status'] = df['status'].apply(str.lower)
```

Groupby can be used to count values by group

```
In [62]: df.groupby('sex')['status'].value_counts()
```

```
Out[62]: sex  status
         F    married     19
              single       7
              widow        2
              divorced     1
         M    married     16
              single      13
              divorced     1
              widow        1
         Name: status, dtype: int64
```

Write a DataFrame to a .csv and Excel files

```
In [63]: df.to_csv('gym_new.csv', sep = ';', index=False)
```

```
In [64]: df.to_excel('gym_new.xlsx', index=False)
```