Creating, Reading and Writing

1. Creating a DataFrame

```
import pandas as pd
pd.DataFrame({'col_A':[1,2],
              'col_B':[3,4]})
```

We can precise indexes this way:

```
import pandas as pd
pd.DataFrame({'col_A':[1,2],
              'col_B':[3,4]},
              index = ['idx_1','idx_2'])
```

2. Series

It's basically a single column DataFrame

```
pd.Series([30, 35, 40],
           index=['2015 Sales', '2016 Sales', '2017 Sales'],
           name='Product A')
```

3. Reading data files

```
data = pd.read_csv("data.csv", index_col=0)
```

We can easily get the size of a DataFrame in the format (row, columns) (it's an attribute)

```
data.shape
```

Indexing, Selecting & Assigning

1. Native accessors

We can access a column easily:

```
data.col_name
data["col_name"]
# we can also get a precise datum:
data["col_name"][idx]
```

2. Indexing in pandas

Whe can access to the data with iloc:

```
data.iloc[[rows_idx], [columns_idx]]
```

Whe can also access to the data with loc:

```
data.iloc[[rows_idx], [columns_names]]
```

iloc: 0:10 will select entries 0,...,9 loc: 0:10 will select entries 0,...,10

3. Manipulating the index

```
data.set_index("column_name")
```

4. Conditional selection

We have to use loc:

```
data.loc[bool_to_select_rows]
```

Summary Functions and Maps

1. Summary functions

```
data.col_name.describe() # Overall description
data.col_name.mean() # Get the mean
data.col_name.unique() # So that to get the unique values
```

2. Maps

An example of use to remean a column

```
# using map()
data_col_name_mean = data.col_name.mean()
data.col_name.map(lambda p: p - data_col_name_mean)

# using apply()
def remean_col_name(row):
    row.col_name = row.col_name - data_col_name_mean
    return row

data.apply(remean_col_name, axis='columns')
```

Note that map() and apply() return new, transformed Series and DataFrames, respectively. They don't modify the original data they're called on.

Grouping and Sorting

1. Groupwise analysis

```
data.groupby('col_name1').col_name2.count()
data.groupby(['col_name1']).col_name2.agg([len, min, max])
```

Renaming and combining

1. Renaming

```
# rename a columns:
data.rename(columns={'old_name': 'new_name'})
# rename an index:
data.rename(index={0: 'firstEntry', 1: 'secondEntry'})
```

Renaming and combining

1. Combining

```
# Concatenation:
data_1 = pd.read_csv("link_1")
data_2 = pd.read_csv("link_2")
data_concat = pd.concat([data_1, data_2])

# Joining
left = pd.read_csv("link_left")
right = pd.read_csv("link_right")

data_join = left.join(right)
```