⊙ Group by is the operation in pandas that creates groups of rows by a column or set of columns.

⊙ Group by creates sub dataframes, that means it's actually split the main dataframe into sub dataframes based on a single column value or combination of multiple column value.

⊙ groupby() is the method used to do this, we can pass a single column name or a list of column names as the argument of this method.

```
data = pd.read_csv('filename.csv')
grp_result = data.groupby('city')
```

Pictorial representation

```
import pd as pd
nf = pd.read_excel(r'path\foldername with extension')
```

call
nf
↳ excel file (output)

object
↓
```
grp_result = nf.groupby('city')
```

~~grp_result = g~~

```
grp_result.get_group('paris')
```
↳ only pair city data represent

```
for city, city_df in grp_result:
    print(city)
    print(city_df)
```
↳ output
all city data and name

Group Operations

- Aggregate functions on group objects

```
grp_result.max()
grp_result.min()
```

- Analytics on specified group columns

```
grp_result.get_group('mumbai').max()
```

Output

Group by Multiple columns

- We can pass a list of columns

```
grp = objectname(['city', 'event'])
```

- While fetching you should pass a tuple to get the group

```
grp.get_group(('new york', 'Sunny'))
```

- Fetch all using for loop

```
for city, city_df in grp:        # It represent dictionary
    print(city)
    print(city_df)
```

Output

Concat...

- Concat() function is used to concatenate two or more Dataframes.
- Concat can do both vertically (axis=0) as well as horizontal (axis=1)

```
tcs = {'id': [101, 102, 103, 104],
       'name': ['Mithun', 'Dipin', 'jose', 'Rahul'],
       'Age': [25, 26, 27, 29]}

wipro = {'id': [101, 102, 103, 104],
         'Name': ['dev', 'sreenath', 'sreeraj', 'shafeel'],
         'Age': [24, 24, 27, 26]}

tcs_emp = pd.DataFrame(tcs)
wipro_emp = pd.DataFrame(wipro)
pd.concat([tcs_emp, wipro_emp])
```

Output

	id	Name	Age	Name
0	101	Mithun	25	
1	102	Dipin	26	
2	103	jose	27	
3	104	Rahul	28	
0	101	dev	24	
1	102	sreenath sreeraj	24	
2	103	shafeel	27	
3	104	shafeel	26	

Igonre index → To get common index after concatenate

pd.concat([tcs_emp, wipro_emp], ignore_index=True)

Output

all data show

	id	Name	Age
0	101	Mithun	25
1	102	Dipin	26
2	103	jose	27
3	104	Rahul	28
4	101	dev	24
5	102	sreenath	24
6	103	sreeraj	27
7	104	shafeel	26

⊙ Keys – To provide separate keys for each concatenate data frames.

data = pd.concat([tcs_emp, wipro_emp], keys=['TCS', 'WIPRO'])

cal

data

Output

		id	Name	Age
TCS	0	101	Mithun	25
	1	102	Dipin	26
	2	103	jose	27
	3	104	Rahul	28
WIPRO	0	101	dev	24
	1	102	sreenath	24
	2	103	sreeraj	27
	3	104	shafeel	28

⊙ Concatenating vertically

```
data = pd.concat([tcs_emp, wipro_emp], axis=1)
```

Call
data

Output

	id	Name	Age	id	Name	Age
0	101	Mithun	25	101	dev	24
1	102	Dipin	26	102	Sreenath	24
2	103	jose	27	103	sreeraj	27
3	104	Rahul	28	104	shajeek	26

⊙ Concatenating a dataframe with a series

```
salary = pd.Series([65000, 75000, 15000, 25000],
                   name='salary')
pd.concat([tcs_emp, salary], axis=1)
```

Output

	id	Name	Age	salary
0	101	Mithun	25	65000
1	102	Dipin	26	75000
2	103	jose	27	15000
3	104	Rahul	28	25000