จัดการข้อมูลด้วย Python & Pandas

ติดตามผู้เขียน
ผ่านช่องทางยูทูป

https://www.youtube.com/channel/UCQ1r_4x-P-fETLIU4pqf98w

https://www.facebook.com/KongRuksiamTutorial/

หรือสแกน QRCODE

เรียนเนื้อหา Pandas ได้ที่

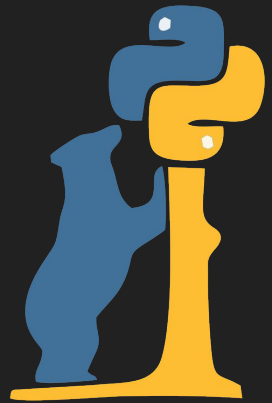
https://bit.ly/3sm4G5P

Pandas คืออะไร

เป็นไลบรารี่ในภาษา Python สำหรับจัดการและวิเคราะห์ข้อมูลที่เป็นแบบโครงสร้างทั้งรูปแบบมิติเดียวและหลายมิติ

การติดตั้ง

```
pip install pandas
```



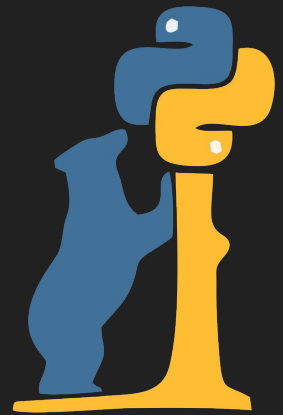
โครงสร้างข้อมูลของ Pandas

- Series
- DataFrame
- Panel

โครงสร้างข้อมูลของ Pandas | Series

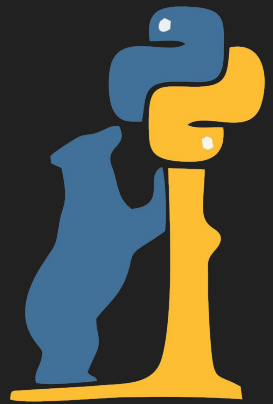
Series คือ เป็นโครงสร้างข้อมูลแบบ Array 1 มิติ (คอลัมน์เดียว) ที่เก็บข้อมูลต่างชนิดกันได้ มีส่วนประกอบ 2 ส่วน คือ

- Index ใช้อ้างอิงตำแหน่งของข้อมูล
- Element Value คือ ข้อมูลในแต่ละ Index



โครงสร้างข้อมูลของ Pandas | DataFrame

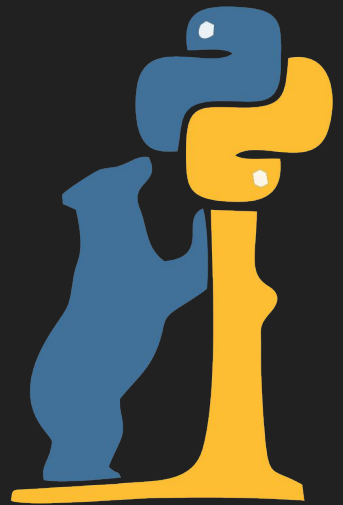
DataFrame เป็นโครงสร้างข้อมูลแบบ Array 2 มิติ (ลักษณะเป็นตารางประกอบด้วย 2 คอลัมน์ขึ้นไป) หรือ การนำ Series หลายๆอันมาเรียงต่อกัน เช่น ข้อมูลคะแนนนักเรียน ข้อมูลสินค้า ข้อมูลประชากร เป็นต้น



โครงสร้างข้อมูลของ Pandas | Panel

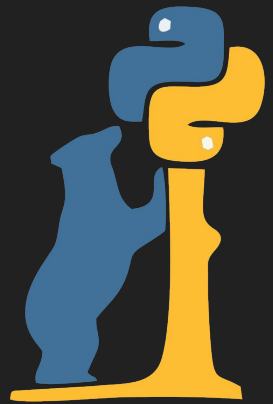
Panel เป็นโครงสร้างข้อมูลแบบ 3 มิติ เป็นการนำ DataFrame หลายๆอันมาซ้อนกันจนเกิดเป็นชั้นหลายๆชั้น ประกอบไปด้วย 3 แกน

- แกนที่ 0 - DataFrame
- แกนที่ 1 - แถวหรือ Index ของแต่ละ DataFrame
- แกนที่ 2 - คอลัมน์ของแต่ละ DataFrame



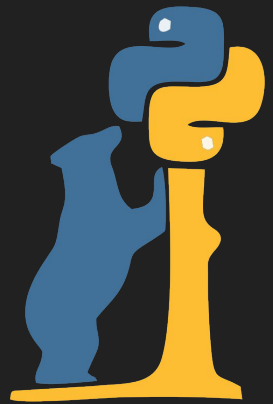
การสร้าง Series จาก List และ Tuple

```
import pandas as pd

data_ls = [10,20,'kong',15.05,'มะละกอ'] // list

data_tp = (10,20,'kong',15.05,'มะละกอ') // tuple

ps=pd.Series(data_ls)

ps=pd.Series(data_tp)

ps
```



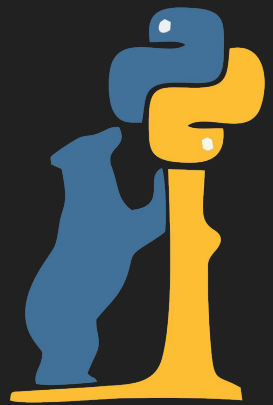
การสร้าง Series จาก Numpy

```
import pandas as pd

import numpy as np

data_ls = [10,20,'kong',15.05,'มะละกอ'] // list

ndata=np.array(data_ls)

ps=pd.Series(ndata)

ps
```



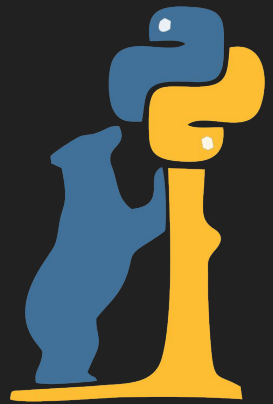
การสร้าง Series แบบกำหนดหมายเลข Index

```
import pandas as pd

items= [‘องุ่น’,’กล้วย’,’มะละกอ’] // list

idx=[50,20,30]

ps=pd.Series(items,index=idx)

ps
```



การสร้าง Series จาก Dictionary

```
import pandas as pd
data= {‘องุ่น’:50,’กล้วย’:20,’มะละกอ’:30} //dictionary
ps=pd.Series(data)
ps
```

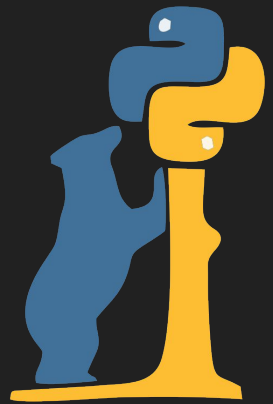


การเข้าถึงข้อมูลใน Series (อ้างอิง Index)

```
import pandas as pd

data= {‘องุ่น’:50,’กล้วย’:20,’มะละกอ’:30} //dictionary

ps[‘องุ่น’]

ps[‘กล้วย’]
```

หรือใช้เลข index

```
ps[0] , ps[1]
```



การเข้าถึงข้อมูลใน Series (แบบช่วงข้อมูล Slice)

โครงสร้างการเข้าถึง

ps[start:stop-1]

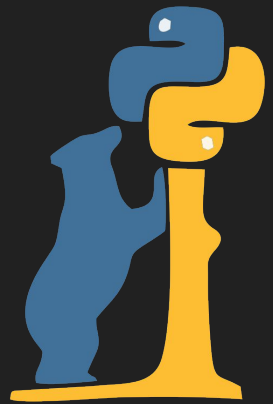
สัญลักษณ์

[:] - ทุกอัน

[2:] - Index ที่ 2 เป็นต้นไป

[:2] - เริ่มต้นไปจนก่อนถึง Index 2

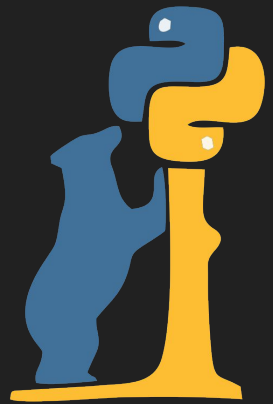
[1:3] - เริ่มต้นที่ Index 1 ไปจนก่อนถึง Index 3



การสร้าง DataFrame

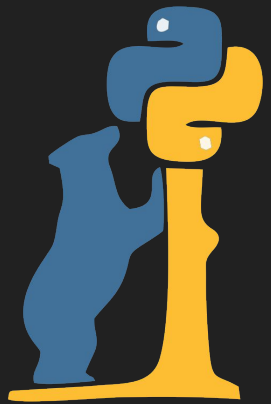
สามารถสร้างได้หลายวิธี

- Tuple และ List
- Series
- Numpy
- CSV



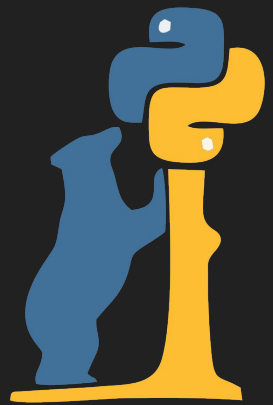
การสร้าง DataFrame จาก List และ Tuple

```
import pandas as pd

data_ls = [10,20,'kong',15.05,'มะละกอ'] // list

data_tp = (10,20,'kong',15.05,'มะละกอ') // tuple

df=pd.DataFrame(data_ls)

df=pd.DataFrame(data_tp)

df
```



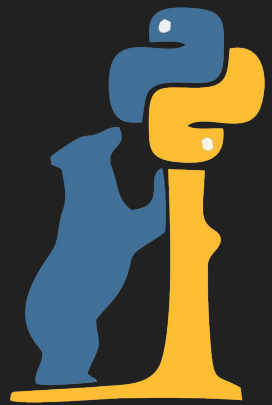
การสร้าง DataFrame แบบกำหนดคอลัมน์

```
import pandas as pd

data = [15,20,23,30]

cols=[‘Age’]

df=DataFrame(data,columns=cols)

df
```



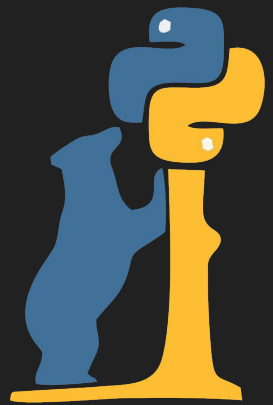
การสร้าง DataFrame จาก List แบบหลายมิติ

```
import pandas as pd

data = [

        [“คีย์บอร์ด”,”อุปกรณ์คอม”,1200],

        [“ตุ๊กตา”,”ของเล่น”,900]

        [“Iphone 12 “,”มือถือ”,30000]

]

cols=[‘Name’,’Category’,’Price’]

df=DataFrame(data,columns=cols)
```



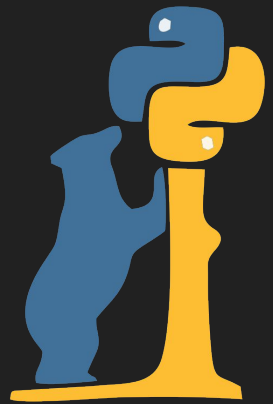
การสร้าง DataFrame จาก List หลายตัวด้วย zip

```
import pandas as pd
item1=['คีย์บอร์ด','อุปกรณ์คอม',1200]
item2 = ['ตุ๊กตา','ของเล่น',900]
item3=['Iphone 12 ','มือถือ',30000]

data=list(zip(item1,item2,item3))
cols=['Name','Category','Price']
df=DataFrame(data,columns=cols)
```



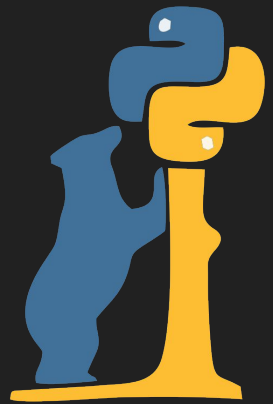
การสร้าง DataFrame จาก Dictionary

```
import pandas as pd
data = [

        {Name:‘คีย์บอร์ด’,Category:’อุปกรณ์คอม’,Price:1200},
        {Name:‘ตุ๊กตา’,Category:’ของเล่น’,Price:900},
        {Name:‘Iphone 12 ’,Category:’มือถือ’,Price:30000}

]
df=pd.DataFrame(data)
df
```



การสร้าง DataFrame แบบกำหนดชื่อคอลัมน์

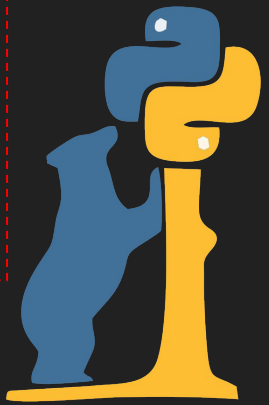
```
import pandas as pd
data = [

        ['คีย์บอร์ด','อุปกรณ์คอม',1200],
        ['ตุ๊กตา','ของเล่น',900],
         ['Iphone 12 ','มือถือ',30000]

]

cols=['Name','Category','Price']

df=DataFrame(data,columns=cols)

df.set_index(['ชื่อ'],inplace=True)
```



การสร้าง DataFrame จาก Series

```
import pandas as pd
name=['คีย์บอร์ด','ตุ๊กตา','Iphone12']
category=['อุปกรณ์คอม','ของเล่น','มือถือ']
price=[1200,900,30000]

names=pd.Series(name)
category=pd.Series(category)
price=pd.Series(price)
```

```
frame={
'Name':names,
'Category':category,
'Price':price
}

df=pd.DataFrame(frame)
```

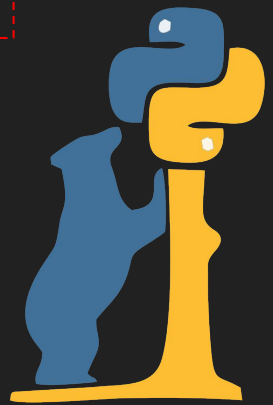


Dataframe ส่งออกเอกสาร CSV

DataFrame.to_csv(ตำแหน่งไฟล์ , attribute)

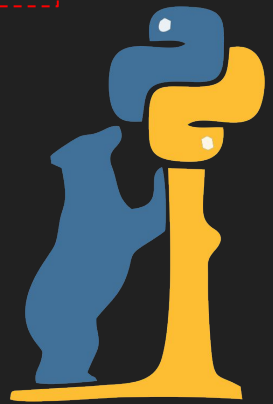
Attribute

- header ต้องการบันทึกส่วนหัวคอลัมน์หรือไม่ (True/False) (Default:True)
- index ต้องการบันทึกเลข Index หรือไม่ (True/False) (Default:True)



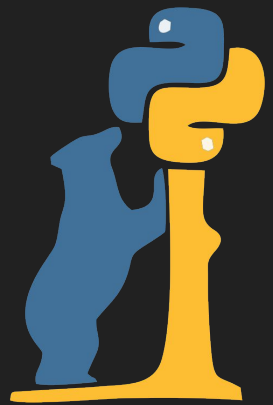
Dataframe อ่านเอกสาร CSV

```
import pandas as pd

df=pd.read_csv('ตำแหน่งไฟล์',encoding='utf-8')

df.head()
```



Dataframe อ่านเอกสาร CSV (เฉพาะบางคอลัมน์)

```
import pandas as pd

df=pd.read(‘ตำแหน่งไฟล์’)

cols=[‘Name’,’Price’]

df=pd.read_csv(‘ตำแหน่งไฟล์’,encoding=’utf-8’,usecols=cols)

df.head()
```



Dataframe อ่านเอกสาร CSV (การกำหนด Header)

```
import pandas as pd

df=pd.read('ตำแหน่งไฟล์')

cols=['Name','Price','Category']

ใช้ Name เป็น Index

df=pd.read_csv('ตำแหน่งไฟล์',encoding='utf-8',names=cols,index_col='Name')

df.head()
```

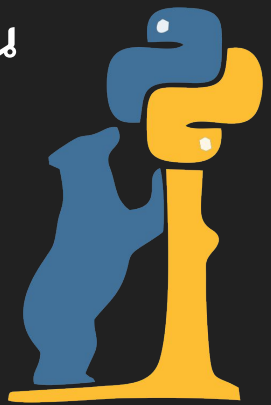
Dataframe อ่านเอกสาร Excel

```
pip install xlrd
```

```
import pandas as pd

df=pd.read_excel('ตำแหน่งไฟล์','ชื่อ sheet',encoding='utf-8')

df.head()
```

การตรวจสอบข้อมูลใน DataFrame

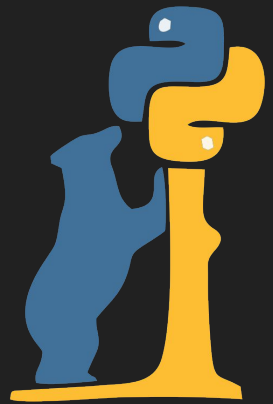
- head() – อ่านข้อมูล 5 แถวแรก
- head(n) – อ่านข้อมูล n แถวแรก
- tail() – อ่านข้อมูล 5 แถวสุดท้าย
- tail(n) – อ่านข้อมูล n แถวสุดท้าย
- sample(n) – สุ่มจำนวนข้อมูล n แถว
- describe() – ดูค่าทางสถิติ เช่น ค่าต่ำสุด-สูงสุด ค่าเฉลี่ยข้อมูล เป็นต้น
- shape – สำหรับนับจำนวนข้อมูล เช่น (20,4) คือ 20 แถว 4 คอลัมน์
- columns – ตรวจสอบว่ามีคอลัมน์อะไรบ้าง
- values – อ่านข้อมูลแบบ Array



ชนิดข้อมูล

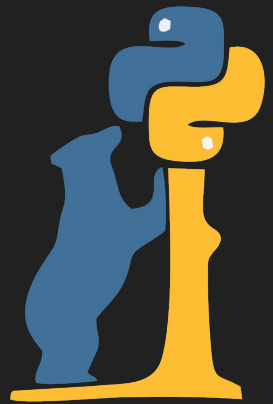
ชนิดข้อมูล	คำอธิบาย
Object	ชุดข้อความ (String), unicode
int64	จำนวนเต็ม
float64	ทศนิยม
Bool	ค่าทางตรรกศาสตร์ (True/False)
datetime64	วันเวลา
timedelta [ns]	ค่าแตกต่างระหว่าง datetime
category	List ของข้อความ

dtypes แสดงชนิดข้อมูลในแต่ละคอลัมน์



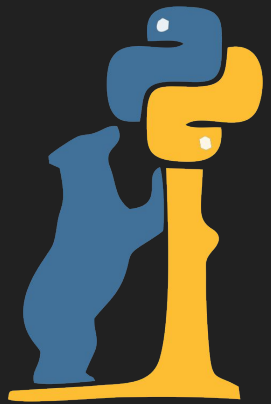
ดูค่าทางสถิติเบื้องต้น

- mean() – ค่าเฉลี่ย
- max() – ค่าสูงสุด
- min() – ค่าต่ำสุด
- count() – นับจำนวนข้อมูลทั้งหมด
- std() – ส่วนเบี่ยงเบนมาตรฐาน
- sum() – ผลรวมข้อมูล



การแปลงชนิดข้อมูลในคอลัมน์แบบ Category

```
df.Grade=df.Grade.astype('category');
```



การเลือกคอลัมน์และเลือกข้อมูลแบบช่วง

DataFrame.Columns

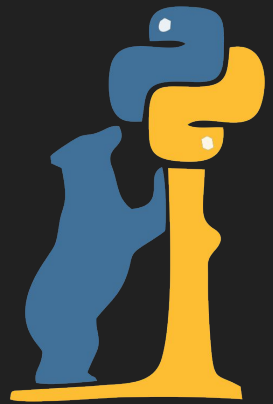
df.temperature

df.event

แบบกำหนดช่วง

df.event[1:3] คอลัมน์ event เลือกแถวที่ 1-2

df[['day','event']][1:3] คอลัมน์ day,event เลือกแถวที่ 1-2



การเลือกข้อมูลแถวที่ต้องการ

#กรณีใช้เลขเป็น Index

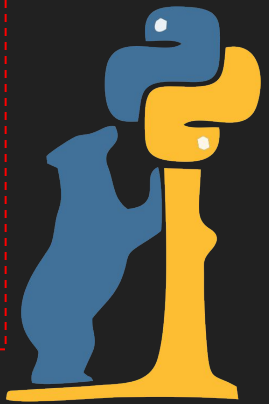
df = pd.read_csv(‘ตำแหน่งไฟล์’)

df[:] # ข้อมูลทั้งหมด

df[1:4] # แถวที่ 1-3

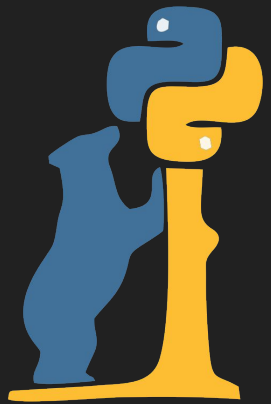
df[1:4].Temperature #เอาเฉพาะคอลัมน์ Name แถวที่ 1-3

df.loc[1:4,’Temperature’,’Event’] เอาแถว 1-3
ของคอลัมน์ ’Temperature’ และ ’Event’



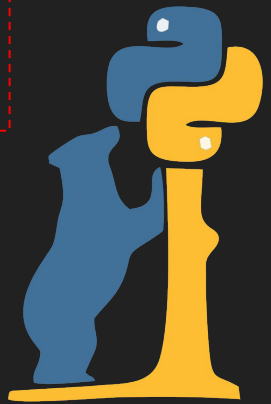
การค้นหาข้อมูล

- **match(string)** ค้นหาคำโดยเริ่มจากอักขระตัวแรก
- **contains(string)** ค้นหาคำไม่จำเป็นต้องเริ่มจากอักขระตัวแรก



การเลือกแถวแบบวนรอบ

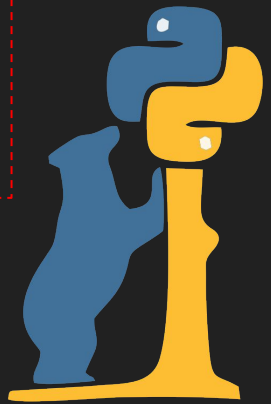
```
df

for idx , row in df.iterrows():

    print(idx,row.Name,row.Price)
```



การเลือกแถวแบบมีเงื่อนไข

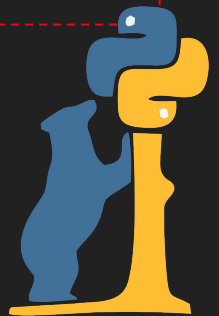
ต้องการทราบว่า

- วันใดบ้างที่มีอุณหภูมิ <=20
- วันใดบ้างที่มีแดดร้อน
- วันใดบ้างที่ฝนตก และ มีอุณหภูมิ >=35
- วันใดบ้างที่มีเมฆมาก หรือ อุณหภูมิ <=18



เลือกช่วงข้อมูลด้วย isin(List) ใช้แทน OR

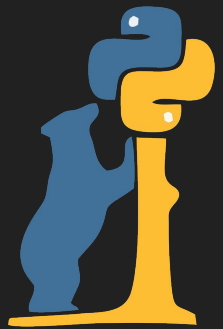
- เลือกวันที่มีอุณหภูมิ 35 องศาเท่านั้น
- เลือกวันที่มีอุณหภูมิ 20 องศาหรือ 35 องศา **เท่านั้น**
- เลือกวันที่มีอุณหภูมิ 30 องศาหรือ 35 องศา **เท่านั้นและเป็นวันที่แดดร้อน**



การจัดเรียงข้อมูลตาม Index

คำสั่งที่ใช้

```
sort_index() # เรียงตาม Index


df=pd.read_csv('ชื่อไฟล์');

result = df.sort_index() #จัดเรียงแล้วเก็บใน result

df.sort_index(inplace=True) # จัดเรียงแล้วเก็บใน df
```



การจัดเรียงข้อมูลตาม Value

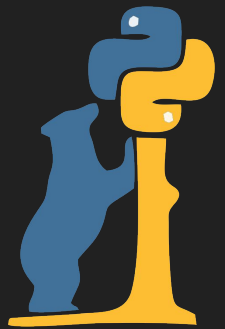
คำสั่งที่ใช้

sort_values() # เรียงตาม value

```
df=pd.read_csv('ชื่อไฟล์');

result = df.sort_value('Price') #จัดเรียงจากน้อยไปมากแล้วเก็บใน result

df.sort_value(inplace=True) # จัดเรียงจากน้อยไปมากแล้วเก็บใน df

df.sort_value(inplace=True,ascending=False) # จัดเรียงจากมากไปน้อยแล้วเก็บใน df
```

การเพิ่มคอลัมน์

- df[‘delivery’] = 100;
  // เพิ่มคอลัมน์ delivery โดยนำราคาไปรวมกับค่าขนส่ง



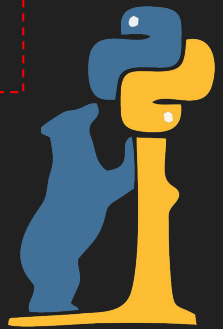
การเปลี่ยนชื่อคอลัมน์

cols={‘Price’:’Amount’} เปลี่ยนจาก Price เป็น Amount

df.rename(columns=cols , inplace=True) เปลี่ยนแล้วแทนที่

cols={‘Price’:’Amount’,’Name’:”ProductName”}

df.rename(columns=cols , inplace=True) เปลี่ยนแล้วแทนที่แบบหลายคอลัมน์



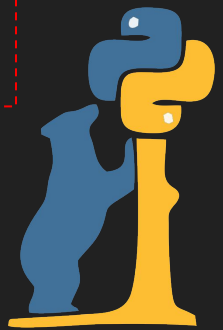
การลบคอลัมน์

```
# ลบคอลัมน์เดียวไม่มีผลกับ df
```

- df.drop('delivery',axis=1)

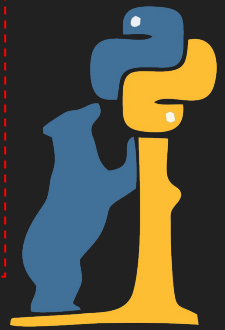
```
#ลบคอลัมน์เดียวแล้วบันทึกทับลงใน df
```

- df.drop('deliery',axis=1,inplace=True)



การเพิ่มข้อมูล /การเพิ่มแถว

```
products = [['หูฟัง',1500,'อุปกรณ์คอม'],['สายชาร์จ',500,'อุปกรณ์คอม']]

cols=['Name','Price','Category']

newdf=pd.DataFrame(data=products,columns=cols)

newdf.set_index('Name',inplace=True)

newdf

df=df.append(newdf)

df
```

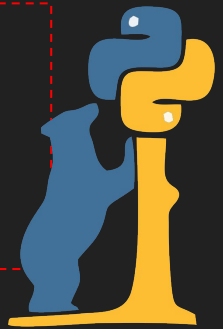


การลบแถวกรณี Index เป็นตัวเลข

```
rows = 2

df.drop(rows , axis = 0, inplace=True) เลข Index ที่จะลบ
```

การลบแถวหลายแถว

```
rows = [1,2,3]

df.drop(rows , axis = 0, inplace=True)
```



การลบแถวกรณี Index เป็นข้อความ

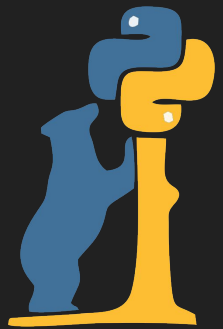
rows = ‘มะม่วง’

df.drop(rows , axis = 0, inplace=True) ชื่อ Index ที่จะลบ

การลบแถวหลายแถว

rows = [‘มะม่วง’,’มะยม’]

df.drop(rows , axis = 0, inplace=True)



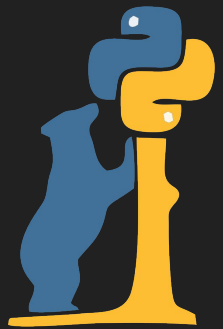
หาผลรวมข้อมูลคอลัมน์และแถว

axis = 0 คอลัมน์ (บนลงล่าง)

axis = 1 แถว (ซ้ายไปขวา)

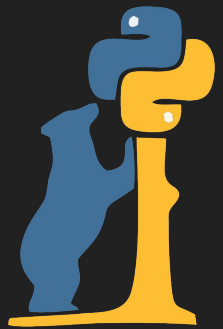
df.sum() ค่าเริ่มต้นคือ axis=0

df.sum(axis=1)



จัดการข้อมูลสูญหาย

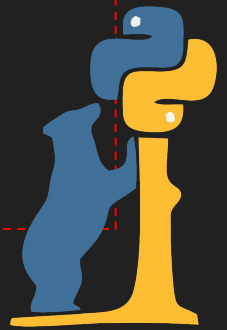
การรวบรวมข้อมูลมาวิเคราะห์นั้น บางครั้งอาจจะมีข้อมูลที่ได้มาไม่ครบบ้าง ตกหล่นหรือขาดหายไปบ้างเรียกส่วนนี้ว่า Missing Data หรือ Missing Value ในหัวข้อนี้จะมาตรวจสอบข้อมูลและจัดการข้อมูลสูญหาย (Clean Data)



การตรวจสอบข้อมูลสูญหายด้วย isnull()

- df.isnull()
  - ค่า True แสดงว่ามีข้อมูลไม่ครบหรือสูญหาย
  - ค่า False แสดงว่าข้อมูลครบ

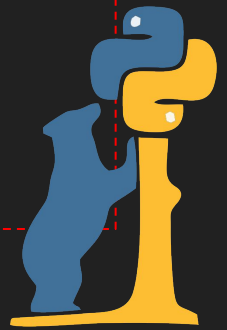
- df.isnull().any() - หาว่าคอลัมน์ใดที่มีค่าว่าง
- df.isnull().sum()- มีค่าว่างกี่แถว



การตรวจสอบข้อมูลครบถ้วนด้วย notnull()

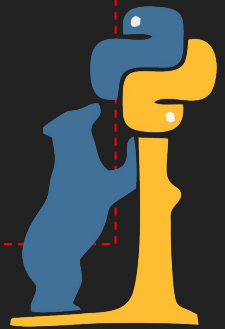
- df.notnull()
  - ค่า True แสดงว่ามีข้อมูลครบ
  - ค่า False แสดงว่าข้อมูลไม่ครบหรือสูญหาย

- df.notnull().any() - หาว่าคอลัมน์ใดที่ข้อมูลครบ
- df.notnull().sum()- ข้อมูลมีครบถ้วนกี่แถว



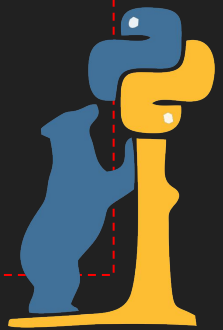
วิธีจัดการข้อมูลสูญหาย

- แทนที่ด้วยค่าเฉลี่ยข้อมูลทั้งหมด
- แทนที่ด้วยค่าตรงๆที่กำหนดขึ้นมา
- แทนที่ด้วยค่าก่อนหน้า
- แทนที่ด้วยค่าถัดไป
- ลบข้อมูล



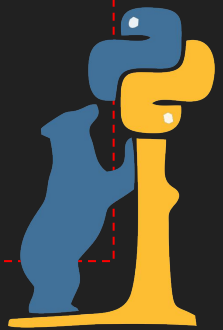
แทนค่าข้อมูลด้วยค่าเฉลี่ย

```
df.describe() // เช็คค่าเฉลี่ย

select = ‘Salary’//เลือกคอลัมน์

df[select]=df[select].fillna(df[select].mean())

df
```



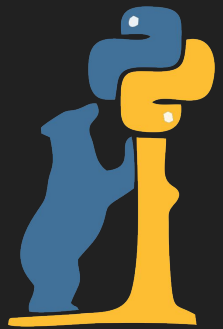
แทนค่าข้อมูลด้วยค่าที่กำหนด

```
select = ‘Salary’//เลือกคอลัมน์

df[select]=df[select].fillna(18000)

//ใส่ 18000 เข้าไป

df
```



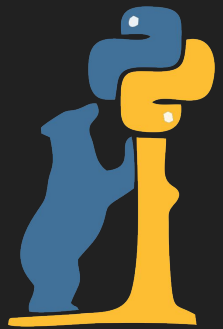
แทนค่าข้อมูลด้วยค่าก่อนหน้า

```
df

df.fillna(method='pad',inplace=True)
```



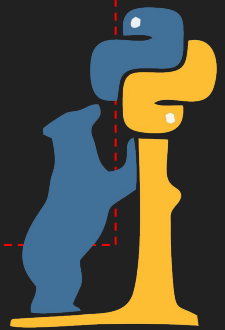
แทนค่าข้อมูลด้วยค่าถัดไป

```
df

df.fillna(method='bfill',inplace=True)
```



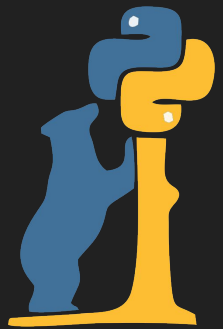
วิธีจัดการข้อมูลสูญหายด้วยการลบ

- ลบทิ้งทั้งหมด
- ลบแถวบางส่วน
- ลบคอลัมน์บางส่วน
- ลบทั้งแถว
- ลบทั้งคอลัมน์



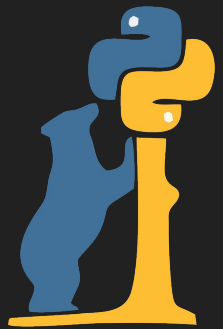
ลบแถวทั้งหมดที่มีค่าว่าง

```
df

df.dropna(inplace=True)
```



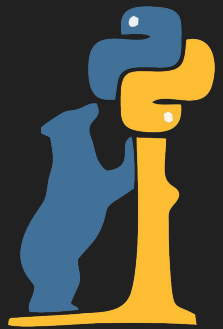
ลบแถวบางส่วนที่มีค่าว่าง

```
df.dropna(subset=['Age','Job'],inplace=True)

df
```



ลบคอลัมน์ที่ข้อมูลไม่ครบออกไป

```
df.dropna(axis='columns',inplace=True)

df
```



ลบแถวที่มีค่าว่างทุกคอลัมน์

1.เช็คว่าว่างก่อน

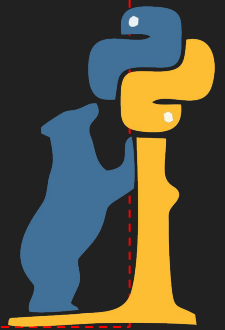
```
df[df.isnull().any(axis=1)]
```

2.ดูว่ามีแถวใดว่างบ้าง

```
df[df.isnull().all(axis=1)]
```

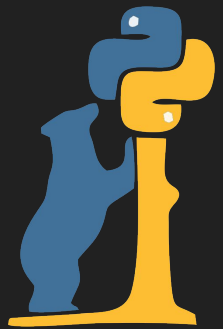
3.ลบข้อมูล

```
df.dropna(how='all',inplace=True)
```



ลบทั้งคอลัมน์ที่มีค่าว่าง

```
df.dropna(axis=1,how='all',inplace=True)

df
```



ลบข้อมูลซ้ำ

เช็คค่าซ้ำ

```
df[df.duplicated()]
```

```
df.drop_duplicates(inplace=True)
```