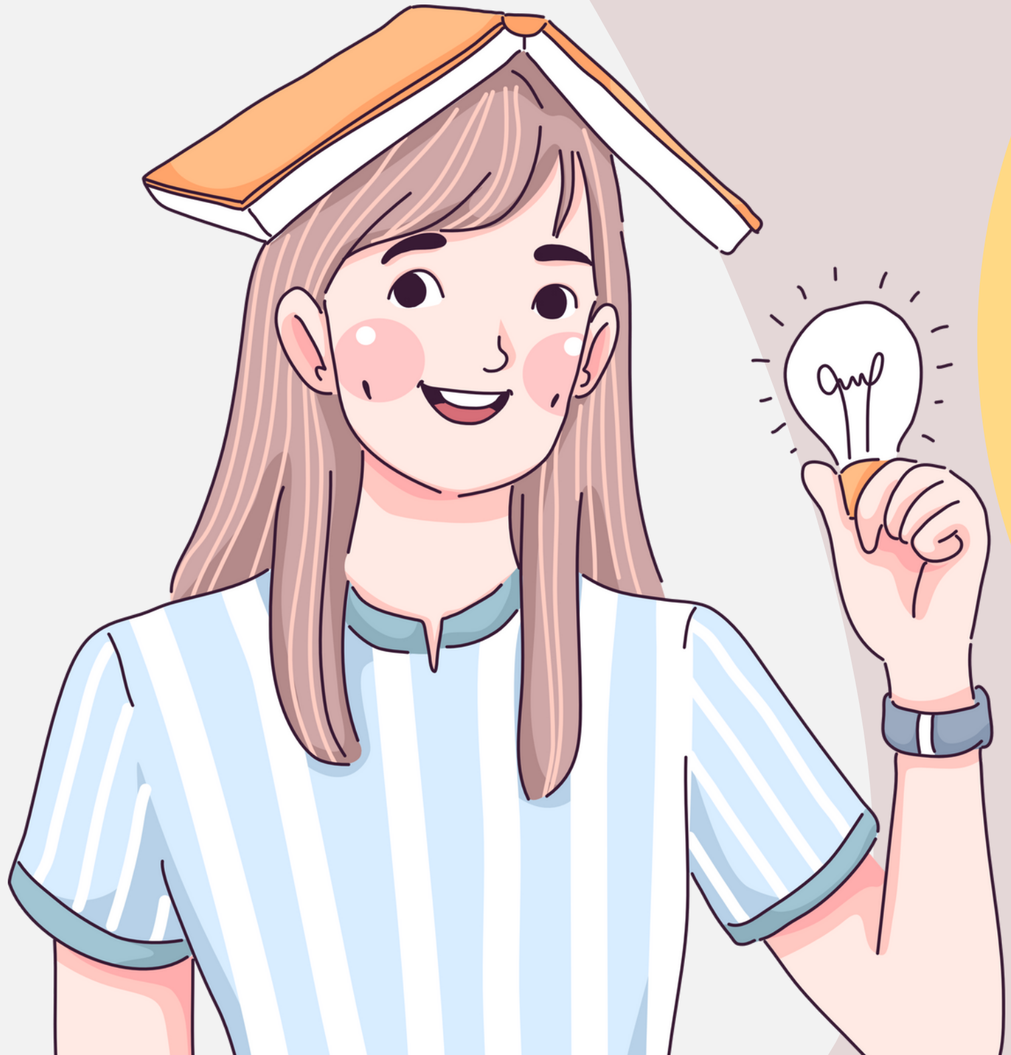
MyEduSolve

Kampus Merdeka
INDONESIA JAYA

STATISTICS FOR DATA SCIENCE

by Group 3



OUR TEAM

Dian Maulida
Dwi Fitriawati Fajrin
Axel Eldrian Hadiwibowo
Agung Prayogi
Yesaya Arya Danar Kristuadji

TABLE OF CONTENTS

INTRODUCTION TO STATISTICS & PROBABILITY


Statistics is the discipline that concerns the **collection, organization, analysis, interpretation,** and **presentation** of data.

TYPES OF STATISTICS

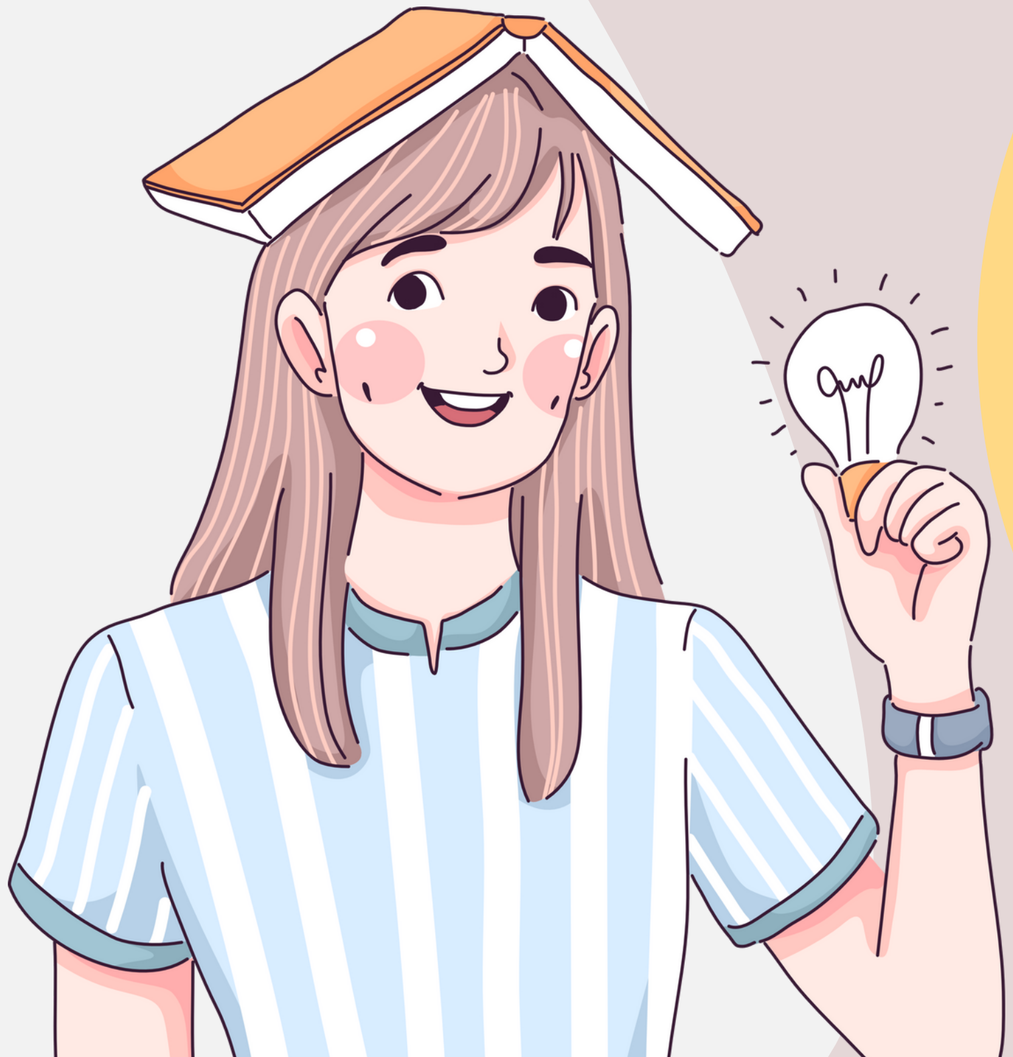
- Descriptive Statistics
- Inferential Statistics

The difference between

DESCRIPTIVE & INFERENTIAL STATISTICS

S. No	Descriptive Statistics	Inferential Statistics
1	Concerned with the describing the target population	Make inferences from the sample and generalize them to the population.
2	Organize, analyze and present the data in a meaningful manner	Compares, test and predicts future outcomes.
3	Final results are shown in form of charts, tables and Graphs	Final result is the probability scores.
4	Describes the data which is already known	Tries to make conclusions about the population that is beyond the data available.
5	Tools- Measures of central tendency (mean/median/ mode), Spread of data (range, standard deviation etc.)	Tools- hypothesis tests, Analysis of variance etc.

*will be explained further in their respective sub topics



WHAT IS DATA?

Data is a **collection of facts**, such as numbers, measurements, observation organized in data matrix, where each row represents an observation or a case and each column represents a variable.

POPULATION AND SAMPLE IN STATISTICS

Population is the entire group that you wish to draw data from (and subsequently draw conclusions about).

Sample is a representative group of a larger population.

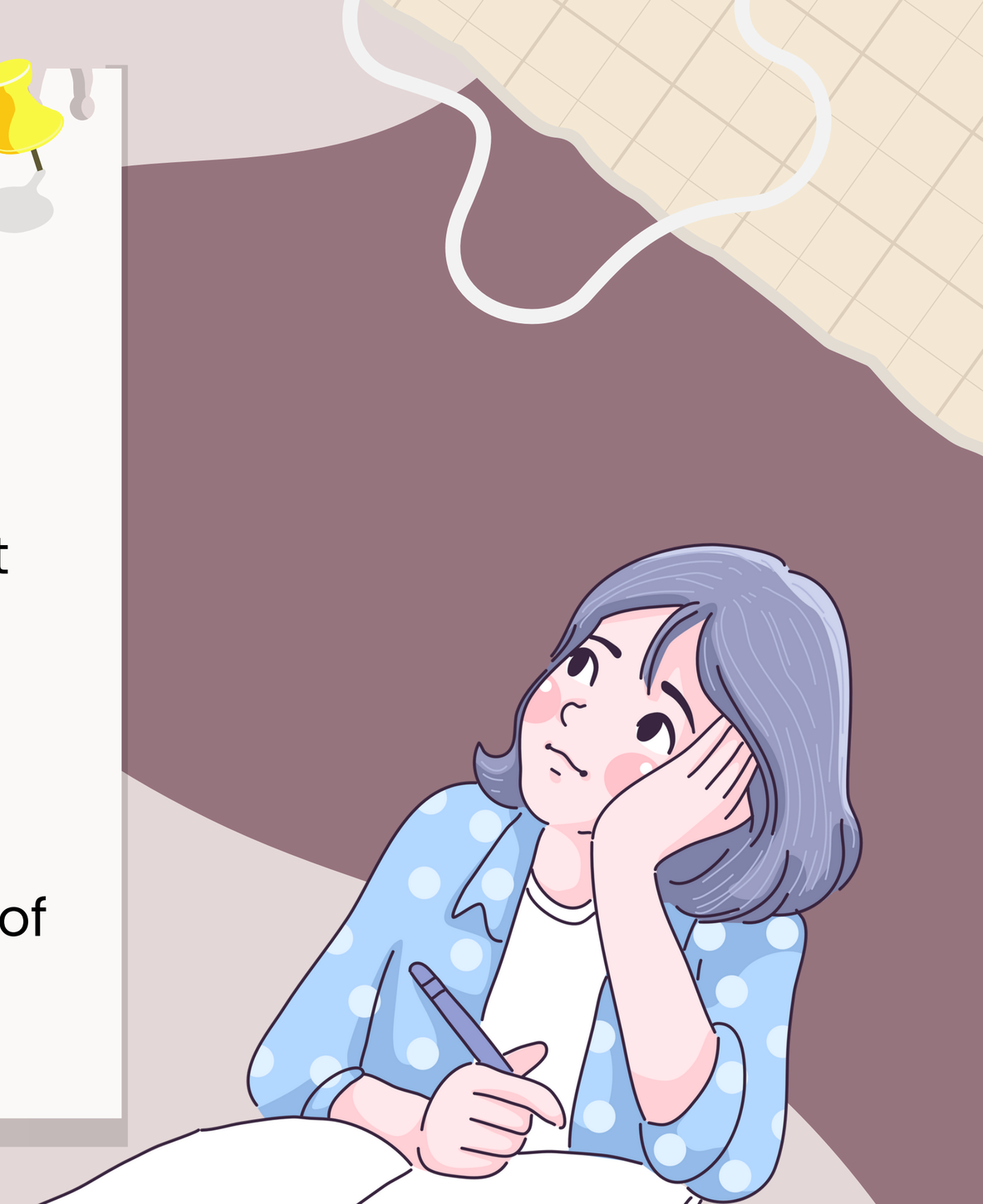
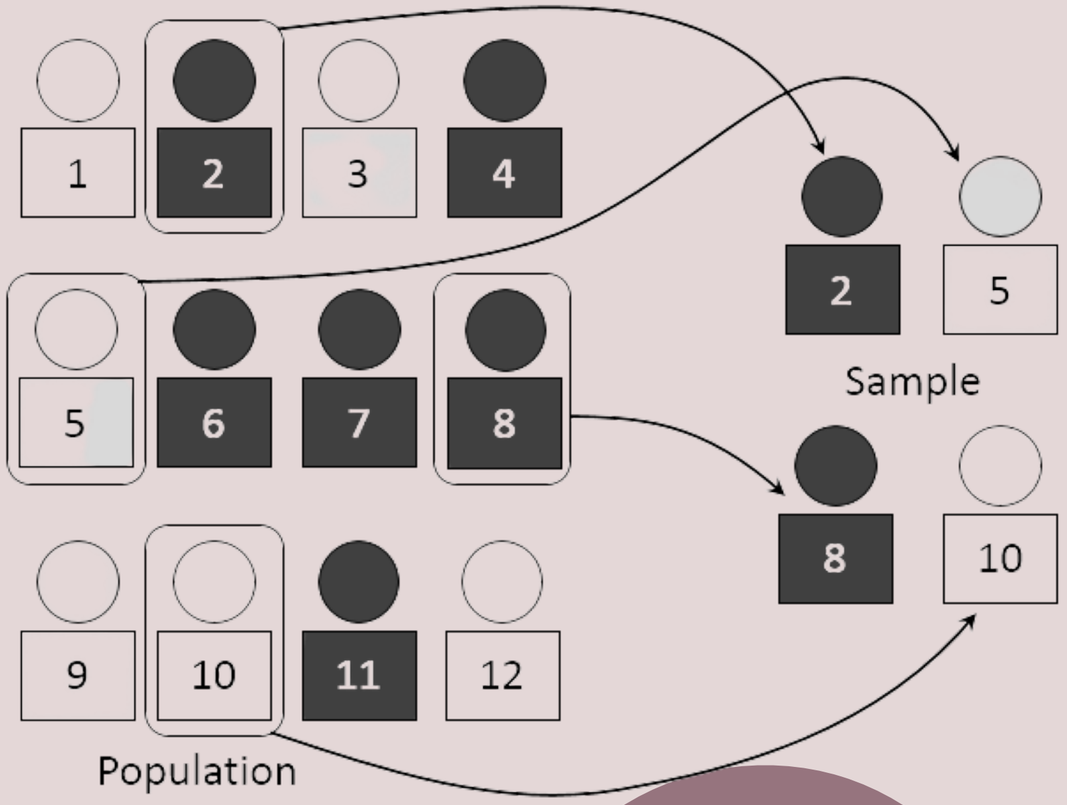


ILLUSTRATION OF POPULATION AND SAMPLE



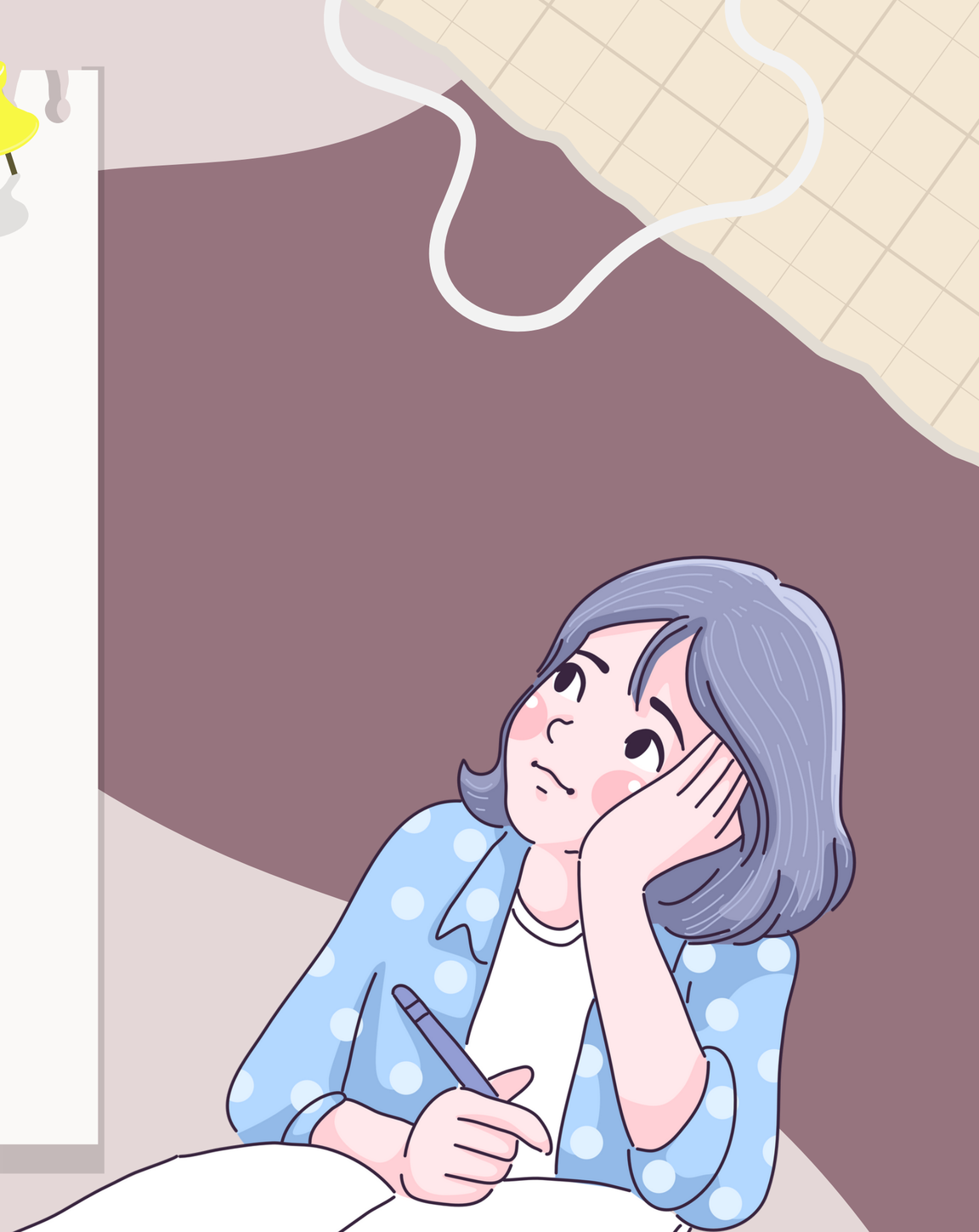
DATA TYPES IN STATISTICS

1. Numerical (Quantitative)

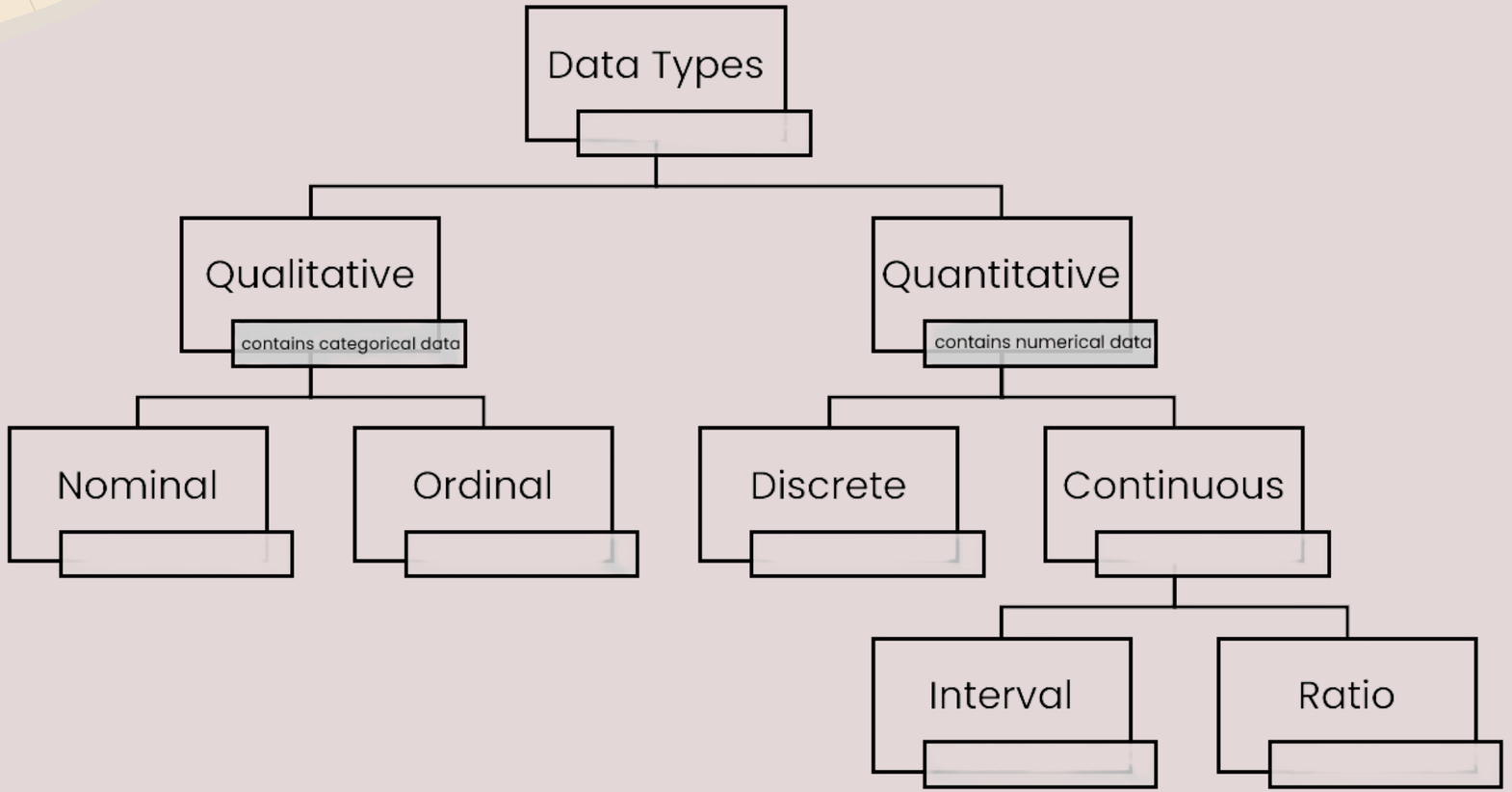
- Discrete
- Continuous
  - Interval
  - Ratio

2. Categorical (Qualitative)

- Nominal
- Ordinal



TYPES OF DATA



source

NUMERICAL

DISCRETE

the variable can only take point values and no values in between.

Example: Number of people in a bus.

CONTINUOUS

the variable can take any value within an interval.

Example: height of a person.

CONTINUOUS

INTERVAL

Data can be ordered as well as differences can be taken, but multiplication/division is not possible.

Example: categorizing as different years like 2011, 2012, etc.

RATIO

Ordering, difference and multiplication/division - all operations are possible.

Example: Age in years, temperature in degrees, etc.

CATEGORICAL

NOMINAL

Only labels data in different categories.

Example categorizing as: Male or Female

ORDINAL

Data can be arranged and ordered but difference doesn't make sense.

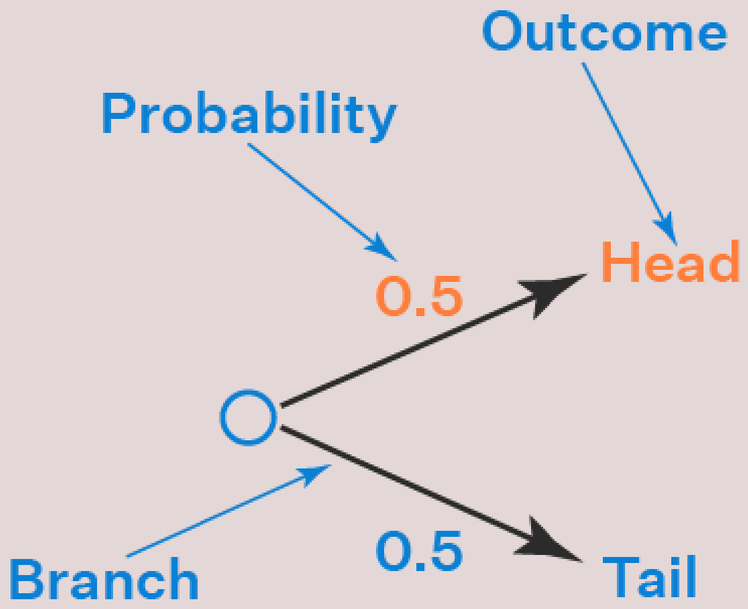
Example: ranking as 1st, second and 3rd.

PROBABILITY

Probability is the branch of mathematics concerning numerical descriptions of how likely an event is to occur, or how likely it is that a proposition is true.

$$P(A) = \frac{n(A)}{n(S)}$$

ILLUSTRATION OF PROBABILITY



Tree diagram for the toss of a coin

DATASET: TITANIC.CSV

Variable	Definition	Key
survival	Survival	0 = No, 1 = Yes
pclass	Ticket class	1 = 1st, 2 = 2nd, 3 = 3rd
sex	Sex	
Age	Age in years	
sibsp	# of siblings / spouses aboard the Titanic	
parch	# of parents / children aboard the Titanic	
ticket	Ticket number	
fare	Passenger fare	
cabin	Cabin number	
embarked	Port of Embarkation	C = Cherbourg, Q = Queenstown, S = Southampton

Sample Data

```
df.head()
```

	PassengerId	Survived	Pclass	Name	Sex	Age	SibSp	Parch	Ticket	Fare	Cabin	Embarked
0	1	0	3	Braund, Mr. Owen Harris	male	22.0	1	0	A/5 21171	7.2500	NaN	S
1	2	1	1	Cumings, Mrs. John Bradley (Florence Briggs Th...	female	38.0	1	0	PC 17599	71.2833	C85	C
2	3	1	3	Heikkinen, Miss. Laina	female	26.0	0	0	STON/O2. 3101282	7.9250	NaN	S
3	4	1	1	Futrelle, Mrs. Jacques Heath (Lily May Peel)	female	35.0	1	0	113803	53.1000	C123	S
4	5	0	3	Allen, Mr. William Henry	male	35.0	0	0	373450	8.0500	NaN	S

EXAMPLE

Probability of a living female passenger

```
na = df[(df['Sex'] == 'female')
        & (df['Survived'] == 1)].shape[0]
ns = df['PassengerId'].shape[0]

p = na/ns
print('Peluang:', p)
```

```
Peluang: 0.2615039281705948
```

n(A) = 233
n(S) = 891

Probability = n(A)/n(S)
= 233/891
= 0.261503928

EXAMPLE

Probability of surviving passengers in each passenger class

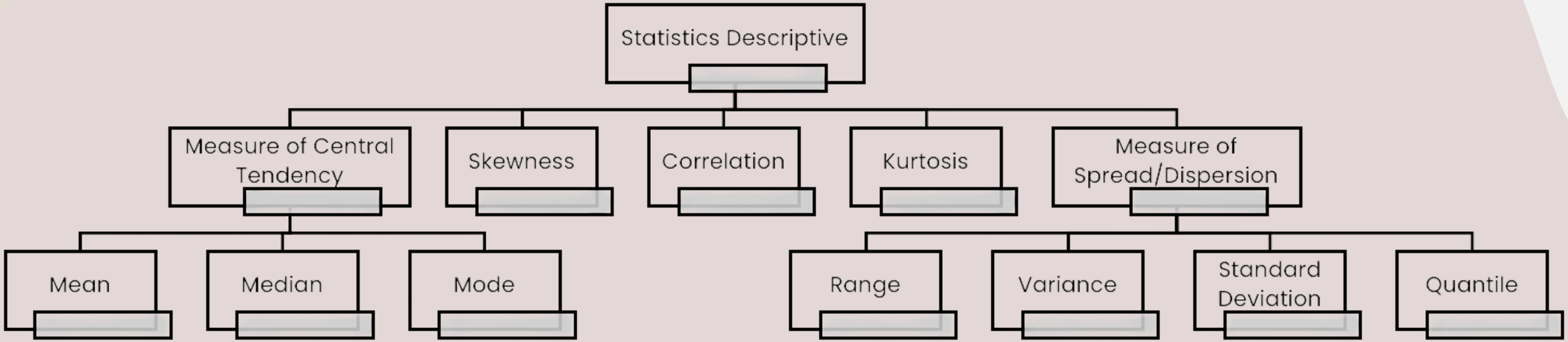
```
a = df[(df['Survived'] == 1)].groupby(
    ['Pclass']).agg({'PassengerId' : 'count'})
a = a.reset_index()
a.columns = ['Class', 'Total Passenger']
a['Probability'] = a['Total Passenger']/(a
                ['Total Passenger'].sum())
a
```

	Class	Total Passenger	Probability
0	1	136	0.397661
1	2	87	0.254386
2	3	119	0.347953

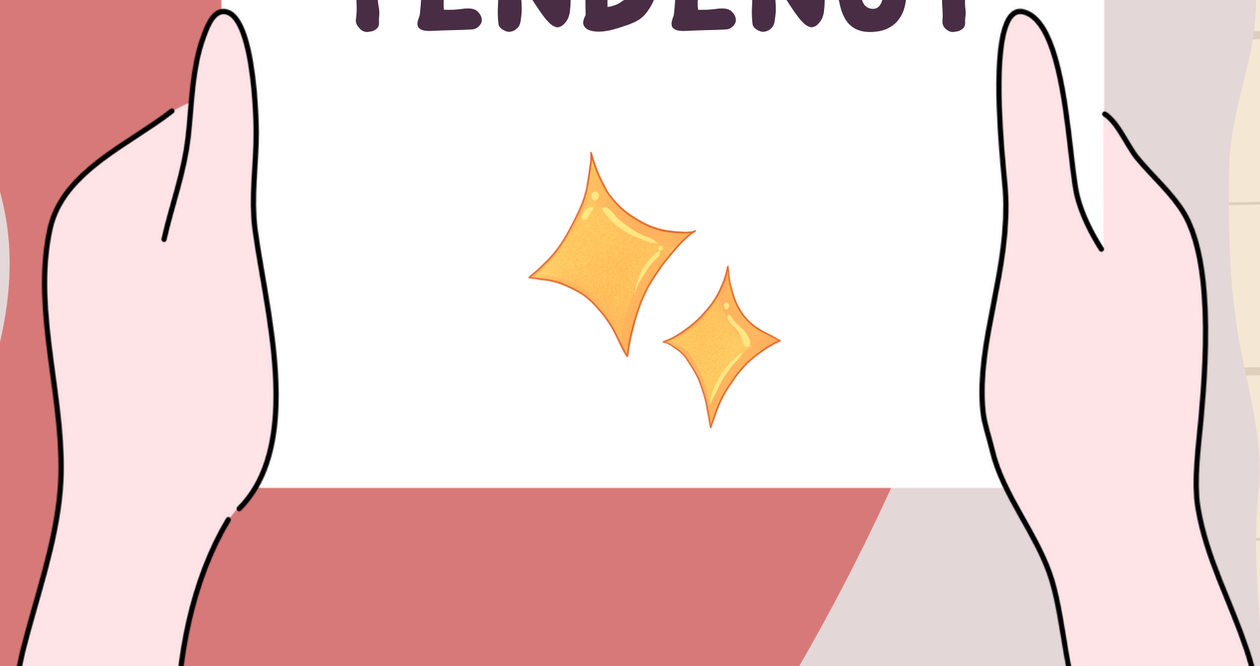
DESCRIPTIVE STATISTICS

Descriptive statistics are used to **describe** the **characteristics** or features of a dataset. It can be used to describe both individual quantitative observations as well as the overall process of obtaining insights from these data.

STATISTICS DESCRIPTIVE



MEASURE OF CENTRAL TENDENCY



- Mean
- Median
- Mode

MEAN

Mean is the average value. The mean is obtained by adding up all the data and then dividing by the number of data.

$$\text{Mean} = \frac{\text{Sum of Observations}}{\text{Total Number of Observations}}$$

DATASET

```
df = pd.read_csv('contoh_data.csv')
data_kelompok = df[['Nama', 'Umur']]
data_kelompok
```

✓ 0.1s

	Nama	Umur
0	Axel	20
1	Dian	20
2	Yogi	21
3	Yesaya	21
4	Dwi	21

PYTHON

```
umur = data_kelompok['Umur']
umur.mean() # get data mean
```

✓ 0.5s

```
20.6
```

MANUAL

mean
= 20 + 20 +21 + 21 + 21
= 103 / 5
= 20.6

MEDIAN

Median is the middle value of an ordered data set.

If ‘n’ is odd: $\text{Median} = \left(\frac{n+1}{2}\right)^{th} \text{term}$

If ‘n’ is even: $\text{Median} = \frac{\left(\frac{n}{2}\right)^{th}\text{term} + \left(\frac{n}{2}+1\right)^{th}\text{term}}{2}$

DATASET

```
df = pd.read_csv('contoh_data.csv')
data_kelompok = df[['nama', 'tb']]
data_kelompok
```

✓ 0.9s

	nama	tb
0	Axel	180
1	Yogi	182
2	Dian	170
3	Yesaya	178
4	Dwi	172

PYTHON

```
data_tinggi_badan = df['tb']
data_sort_tinggi_badan = data_tinggi_badan.sort_values()
data_sort_tinggi_badan.median()
```

✓ 0.3s

```
178.0
```

MANUAL

MEDIAN

[170, 172, **178**, 180, 182]

MODE

The mode is the value that appears most often in a data set. and if there is data with the same amount of data, the mode to be taken is the data with the smallest value.

$$\text{Mode} = L + h \frac{(f_m - f_1)}{(f_m - f_1) + (f_m - f_2)}$$

DATASET

```
df = pd.read_csv('contoh_data.csv')
data_kelompok = df[['Nama', 'Umur']]
data_kelompok
```

✓ 0.1s

	Nama	Umur
0	Axel	20
1	Dian	20
2	Yogi	21
3	Yesaya	21
4	Dwi	21

PYTHON

```
import statistics

modus_data_usia = df['usia']
statistics.mode(modus_data_usia)
```

✓ 0.3s

```
21
```

MANUAL

mode

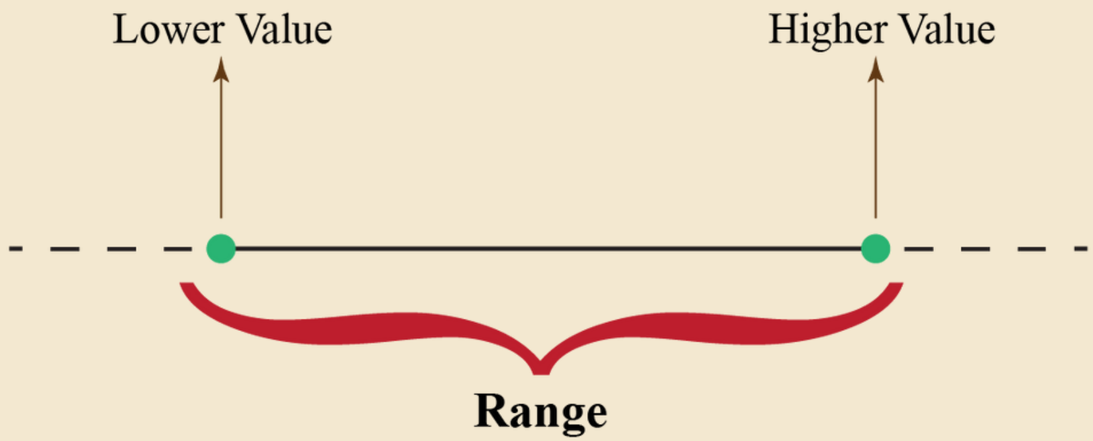
number 21 in the data
appears 3 times

MEASURE OF SPREAD/ DISPERSION

- Range
- Variance
- Standard Deviation
- Quantiles

RANGE

Calculate the difference by substracting the smallest from the largest



PYTHON CALCULATION

The **range** value of the 'Age' column can be calculated from the **maximum** value **minus** the **minimum** value from the 'Age' data

```
df_age_notnull = df['Age'][df['Age'].notnull()]
```

```
print(df_age_notnull)
```

```
0      22.0
1      38.0
2      26.0
3      35.0
4      35.0
       ...
885    39.0
886    27.0
887    19.0
889    26.0
890    32.0
Name: Age, Length: 714, dtype: float64
```

```
df_age_notnull.max()
```

```
80.0
```

```
df_age_notnull.min()
```

```
0.42
```

```
range_age = np.ptp(df_age_notnull)
print(range_age)
```

```
79.58
```

VARIANCE

Variance is a statistical measurement that is used to determine the spread of numbers in a data set with respect to the average value or the mean.

	Population	Sample
Ungrouped	$\sigma^2 = \frac{\sum_{i=1}^{N} (x_i - \bar{x})^2}{N}$	$\sigma^2 = \frac{\sum_{i=1}^{N} (x_i - \bar{x})^2}{N - 1}$
Grouped	$\sigma^2 = \sum_{i=1}^{N} \frac{f(M_i - \bar{X})^2}{N}$	$\sigma^2 = \sum_{i=1}^{N} \frac{f(M_i - \bar{X})^2}{N - 1}$

Xbar : stands for mean
Mi : the midpoint of the ith interval
Xi : the i th data point
N : the summation of all frequencies
n : the number of observations.

PYTHON CALCULATION

Manually, to calculate **variance**, first calculate the mean of a given set of values. Then each data is reduced by the calculated average and squared. After that the results are searched by calculating the average of this squared value

```
import statistics
```

```
variance_age = statistics.variance(df_age_notnull)
print(variance_age)
```

```
211.01912474630805
```

*Manual calculation will be performed on manual calculation of standard deviation

STANDARD DEVIATION

Standard deviation is the degree of dispersion or the scatter of the data points relative to its mean. It tells how the values are spread across the data sample and it is the measure of the variation of the data points from the mean.

Population	Sample
$\sigma = \sqrt{\frac{\sum(X - \mu)^2}{N}}$	$s = \sqrt{\frac{\sum(X - \bar{x})^2}{n - 1}}$
X - The Value in the data distribution μ - The population Mean N - Total Number of Observations	X - The Value in the data distribution $\bar{x}$ - The Sample Mean n - Total Number of Observations

MANUAL CALCULATION

data

	nama	usia
0	Axel	20
1	Yogi	21
2	Dian	20
3	Yesaya	21
4	Dwi	21

get (X - x)^2

usia	usia - mean(usia)	(usia - mean(usia))^2
20	-0.6	0.36
21	0.4	0.16
20	-0.6	0.36
21	0.4	0.16
21	0.4	0.16

get **sum** from **(X - x)^2**
= 0,36+0,16+0,36+0,16+0,16
= 1,2

get **variance** from **sum**
= sum / (n-1)
= 1,2 / (5-1)
= 1,2 / 4
= 0,3

Standart Deviation from Data usia is **0.5477225575** because that is value of square root from **variance**.

PYTHON CALCULATION

data

	nama	usia
0	Axel	20
1	Yogi	21
2	Dian	20
3	Yesaya	21
4	Dwi	21

```
SD_data_usia = data["usia"]
SD_data_usia
```

✓ 0.9s

```
0    20
1    21
2    20
3    21
4    21
```

```
print(statistics.stdev(SD_data_usia))
```

Standard Deviation result

```
0.5477225575051661
```

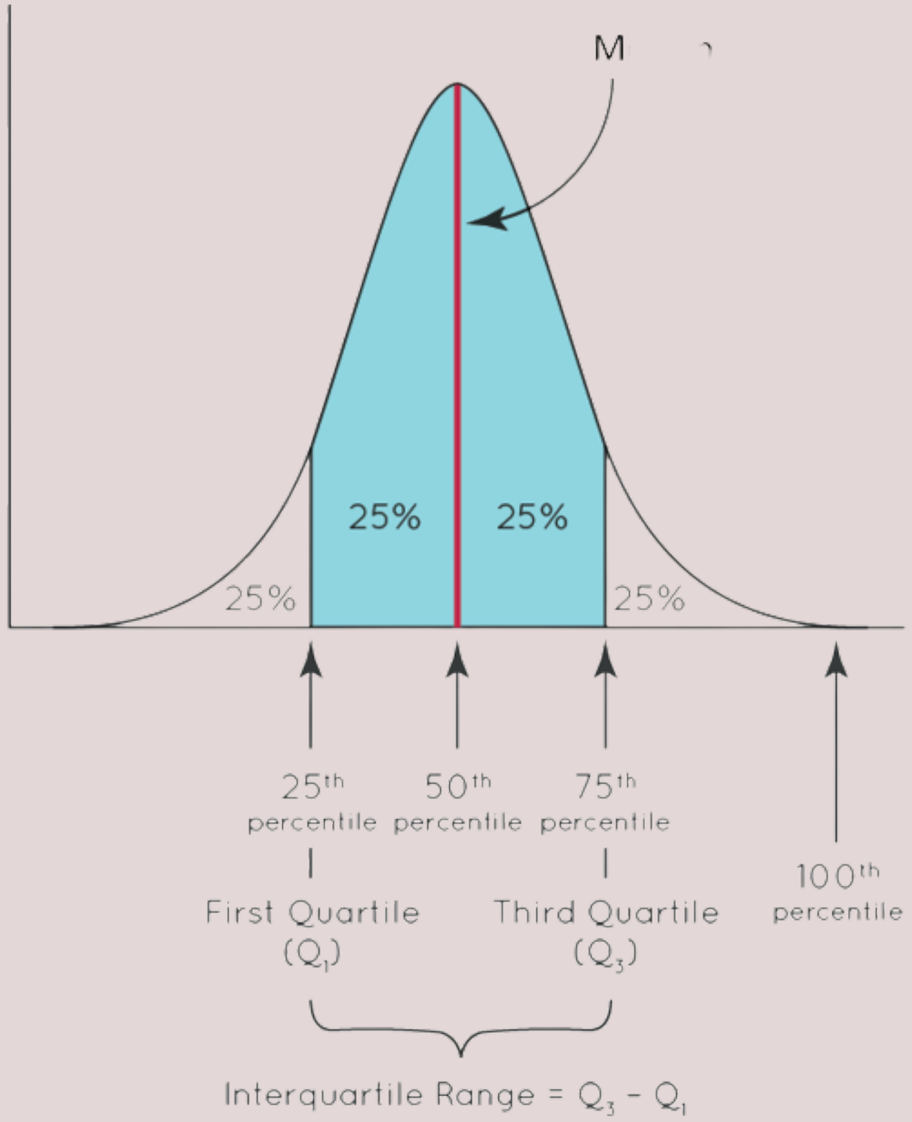
QUANTILE

quantiles are cut points dividing the range of a probability distribution into continuous intervals with equal probabilities, or dividing the observations in a sample in the same way

- Q0 is the smallest value in the data
- Q1 is the value separating the first quarter from the second quarter of the data
- Q2 is the middle value, separating the bottom from the top half
- Q3 is the value separating the third quarter from the fourth quarter
- Q4 is the largest value in the data

QUARTILE

Quantiles is a range from any value to any other value. Note That **percentiles and quartiles are simply types of quantiles.**



QUANTILE WITH PYTHON

DATASET

```
df = pd.read_csv('contoh_data.csv')
data_kelompok = df[['nama', 'tb']]
data_kelompok
```

✓ 0.9s

	nama	tb
0	Axel	180
1	Yogi	182
2	Dian	170
3	Yesaya	178
4	Dwi	172

```
q0 = np.quantile(data2, [0]) # get q0
print(q0)
```

✓ 0.1s

```
[170]
```

```
q1 = np.quantile(data2, [0.25]) # get q1
print(q1)
```

✓ 0.8s

```
[172.]
```

```
q2 = np.quantile(data2, [0.5]) # get q2
print(q2)
```

✓ 0.1s

```
[178.]
```

```
q4 = np.quantile(data2, [1]) # get q4
print(q4)
```

✓ 0.1s

```
[182]
```

```
q3 = np.quantile(data2, [0.75]) # get q3
print(q3)
```

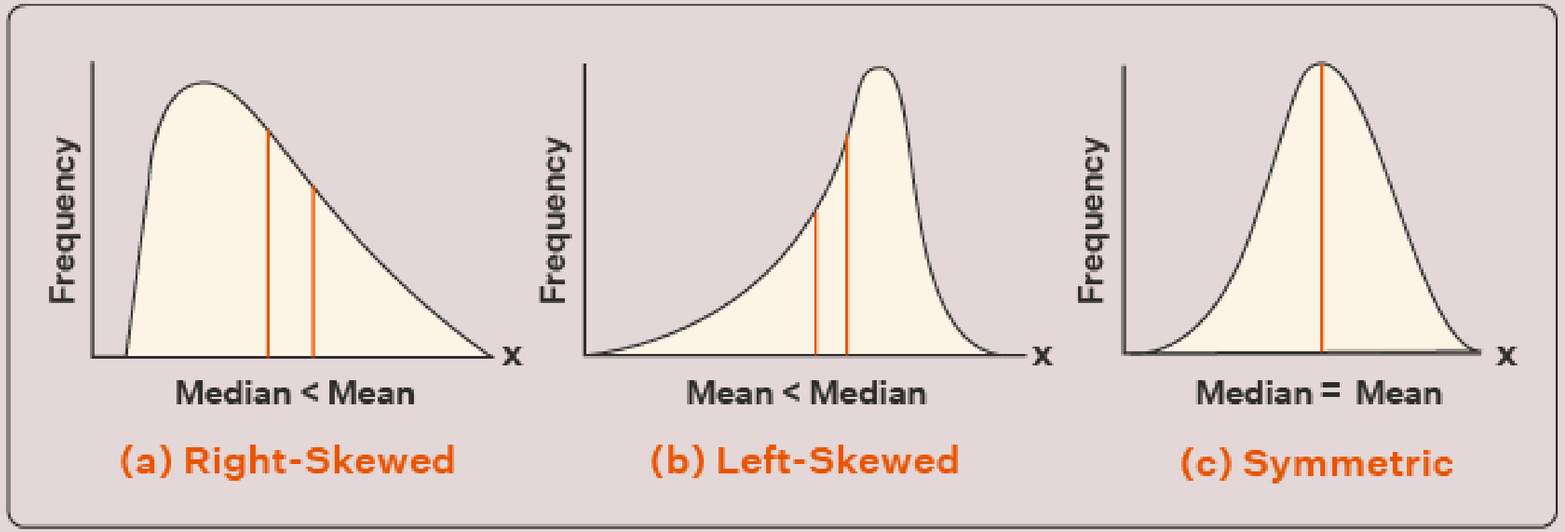
✓ 0.8s

```
[180.]
```

DISTRIBUTION

Distribution describes how values are distributed for a field. In other words, the statistical distribution shows which values are common and uncommon

SKEWNESS



Measure of the asymmetry of a distribution. A distribution is asymmetrical when its left and right side are not mirror images. A distribution can have right (or positive), left (or negative), or zero skewness. A right-skewed distribution is longer on the right side of its peak, and a left-skewed distribution is longer on the left side of its peak.

FORMULA

$$\text{skewness} = \frac{\sum_{i=1}^{N}(x_i - \bar{x})^3}{(N-1)s^3}$$

where:

- s is the standard deviation
- $\bar{x}$ is the mean of the distribution
- N is the number of observations of the sample

Coefficient of Skewness

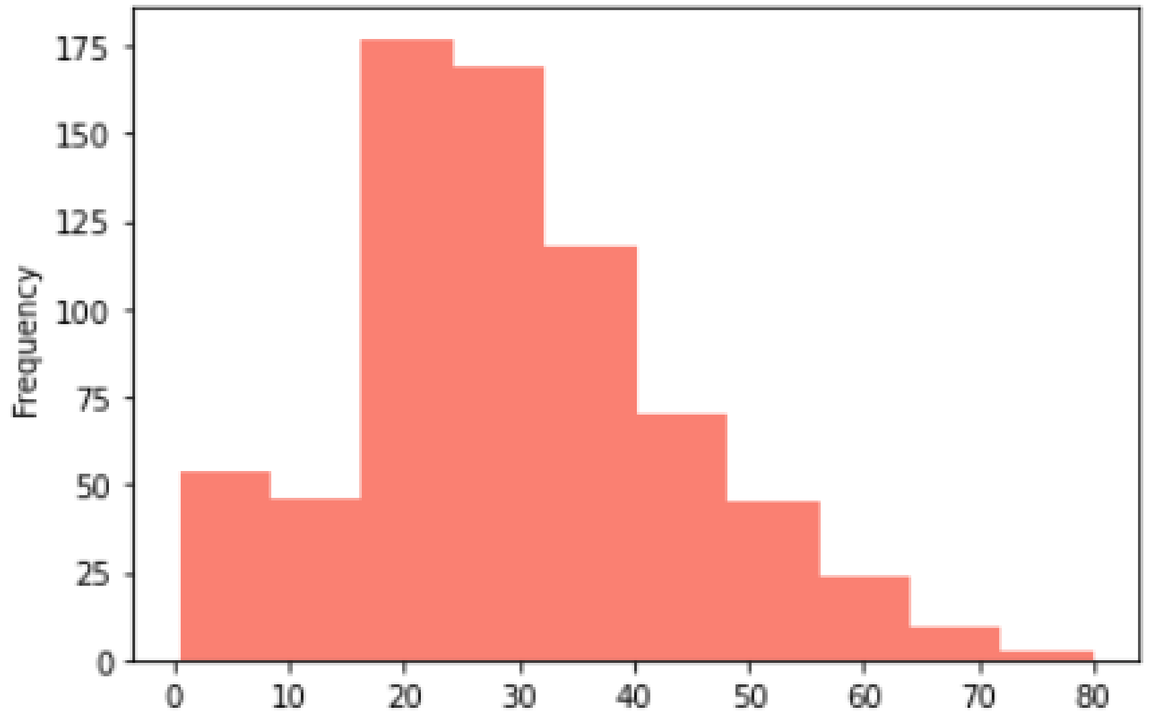
Using Mode: $\frac{\bar{x} - \text{Mode}}{s}$

Using Median: $\frac{3(\bar{x} - \text{Median})}{s}$

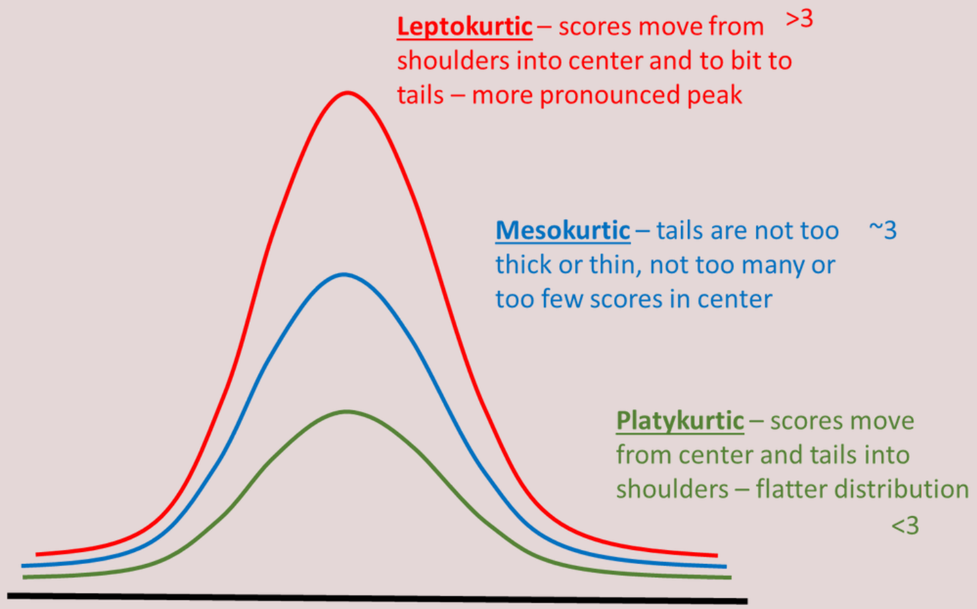
EXAMPLE

In this Age data from titanic dataset has a positive (right) skewness

```
df['Age'].plot(kind='hist', color="salmon");
```



KURTOSIS



$$\text{Kurtosis} = n * \frac{\sum_i^n (Y_i - \bar{Y})^4}{\sum_i^n (Y_i - \bar{Y}^2)^2}$$

Kurtosis: This measures whether or not the tails of a given distribution contain extreme values (also known as outliers). If a tail lacks outliers, we can say that it has low kurtosis. If a dataset has a lot of outliers, we can say it has high kurtosis.

CORRELATION

Correlation is a statistical measure that expresses the extent to which two variables are linearly related (meaning they change together at a constant rate). It’s a common tool for describing simple relationships without making a statement about cause and effect. Correlation is measured numerically using the correlation coefficient.

FORMULA

Population Correlation Coefficient

$$P_{xy} = \frac{\sigma_{xy}}{\sigma_x \ \sigma_y} = \frac{\sum (xi - \bar{x})(yi - \bar{y})}{\sqrt{\left(\sum (xi - \bar{x})^2\right)\left(\sum (yi - \bar{y})^2\right)}}$$

Where,

- $\sigma_x, \sigma_y \rightarrow$ Population Standard Deviation
- $\sigma_{xy} \rightarrow$ Population Covariance
- $\bar{X}, \bar{Y} \rightarrow$ Population Mean

Sample Correction, coefficient between x and y

$$r_{xy} = \frac{S_{xy}}{S_x \ S_y} = \frac{\sum (xi - \bar{x})(yi - \bar{y})}{\sqrt{\left(\sum (xi - \bar{x})^2\right)\left(\sum (yi - \bar{y})^2\right)}}$$

Where,

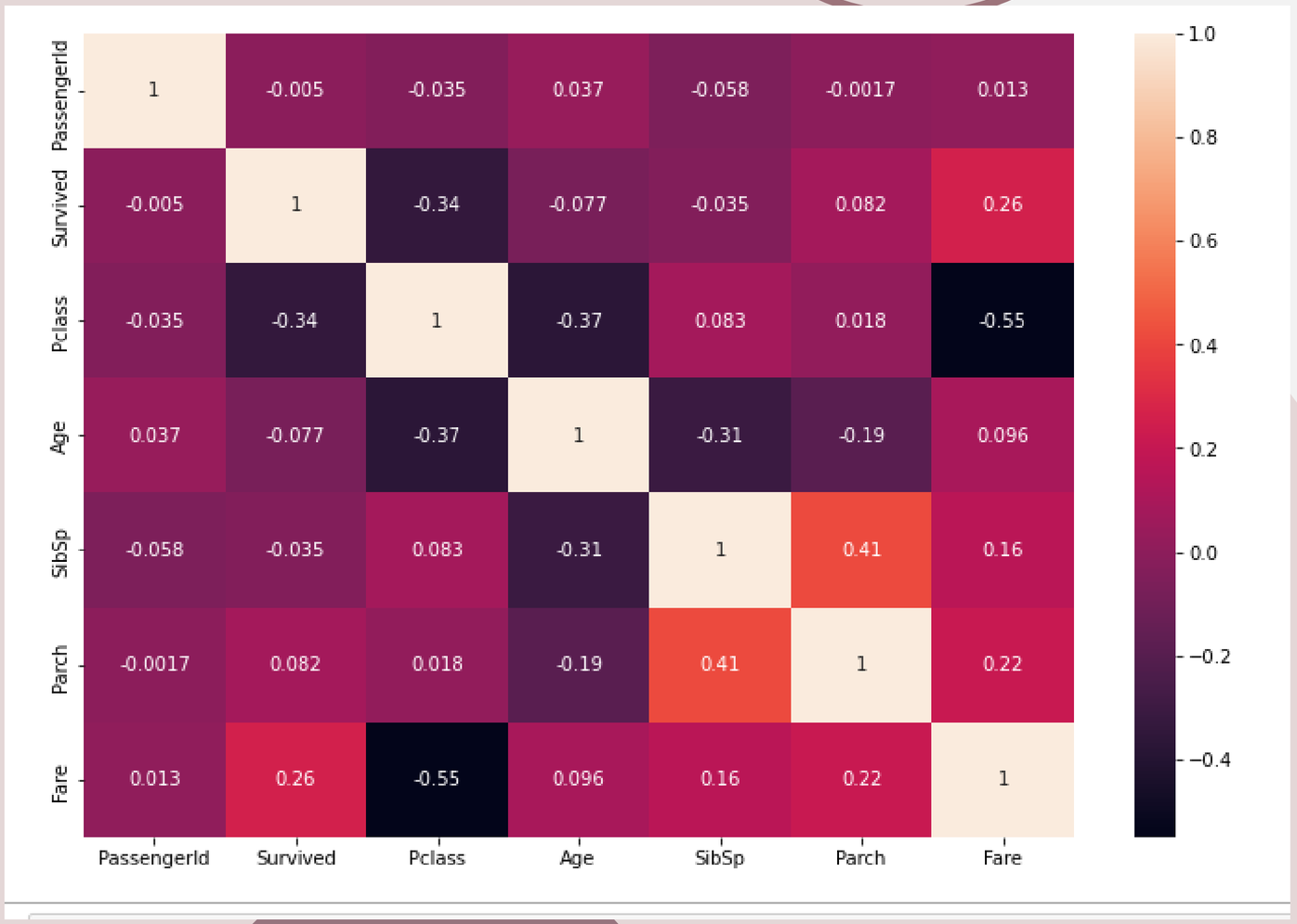
- $S_x, S_y \rightarrow$ Sample Standard Deviation
- $S_{xy} \rightarrow$ Sample Covariance
- $\bar{X}, \bar{Y} \rightarrow$ SampleMean

CORRELATION TITANIC DATASET USING HEATMAP

```
import matplotlib.pyplot as plt
```

```
# Define correlation titanic dataset using heatmap

plt.figure(figsize = (12,9))
sns.heatmap(df.corr(), annot=True);
```

Correlation heatmaps are a type of plot that visualize the strength of relationships between variables. The color-coding of the cells makes it easy to identify relationships between variables at a glance.



CORRELATION TITANIC DATASET

```
df[['Age', 'Survived']].corr()
```

	Age	Survived
Age	1.000000	-0.077221
Survived	-0.077221	1.000000

In this case, the correlation between age and survived has a negative value, it's means that two variables tend to move in opposite directions.

CORRELATION USING GROUP 3 DATA SAMPLE

```
# Correlation group 3 data sample

kel3 = pd.read_csv('kel3_korelasi.csv')
```

```
kel3.head()
```

	Nama	Usia	TB	BB
0	Axel	20	180	60
1	Yogi	21	182	66
2	Dian	20	170	58
3	Yesaya	21	178	70
4	Dwi	21	172	50

In this sample data contain Nama, Usia TB as Tinggi Badan and BB as Berat Badan

```
plt.figure(figsize = (6,4))
sns.heatmap(kel3.corr(), annot=True);
```

Correlation among variable in data sample

CORRELATION USING GROUP 3 DATA SAMPLE

```
kel3[['Usia','TB', 'BB']].corr()
```

	Usia	TB	BB
Usia	1.000000	0.246871	0.213561
TB	0.246871	1.000000	0.667811
BB	0.213561	0.667811	1.000000

Correlation value among data in data sample

DETERMINE CORRELATION MANUALLY USING GOOGLE SHEET

Correlation Coefficient

Nama	Usia	TB	BB
Axel	20	180	60
Yogi	21	182	66
Dian	20	170	58
Yesaya	21	178	70
Dwi	21	172	50

x	TB
y	BB
xbar	176.4
ybar	60.8

x	y	(x-xbar)	(y-ybar)	(x-xbar)(y-ybar)	(x-xbar)^2	(y-ybar)^2
180	60	3.6	-0.8	-2.88	12.96	0.64
182	66	5.6	5.2	29.12	31.36	27.04
170	58	-6.4	-2.8	17.92	40.96	7.84
178	70	1.6	9.2	14.72	2.56	84.64
172	50	-4.4	-10.8	47.52	19.36	116.64
176.4	60.8			106.4	107.2	236.8

sigma((x-xbar)(y-ybar))	106.4
sigma((x-xbar)^2)*sigma((y-ybar)^2))	25384.96
akar(sigma((x-xbar)^2)*sigma((y-ybar)^2)))	159.3265828
cc	**0.66781072**

The correlation between TB and BB from the data sample calculating manually using Google Sheet, as you can see, the value is the same as in the Jupiter notebook

```
kel3[['Usia','TB', 'BB']].corr()
```

	Usia	TB	BB
Usia	1.000000	0.246871	0.213561
TB	0.246871	1.000000	0.667811
BB	0.213561	0.667811	1.000000

DETERMINE CORRELATION MANUALLY USING GOOGLE SHEET

USIA DAN BB						
x	Usia					
y	BB					
xbar	20.6					
ybar	60.8					
x	**y**	**(x-xbar)**	**(y-ybar)**	**(x-xbar)(y-ybar)**	**(x-xbar)^2**	**(y-ybar)^2**
20	60	-0.6	-0.8	0.48	0.36	0.64
21	66	0.4	5.2	2.08	0.16	27.04
20	58	-0.6	-2.8	1.68	0.36	7.84
21	70	0.4	9.2	3.68	0.16	84.64
21	50	0.4	-10.8	-4.32	0.16	116.64
20.6	60.8			3.6	1.2	236.8
sigma((x-xbar)(y-ybar))			3.6			
sigma((x-xbar)^2)*sigma((y-ybar)^2)			284.16			
akar(sigma((x-xbar)^2)*sigma((y-ybar)^2)))			16.857046			
cc			**0.213560549**			

The correlation between Usia and BB from the data sample calculating manually using Google Sheet, as you can see, the value is the same as in the Jupiter notebook

```
kel3[['Usia','TB', 'BB']].corr()
```

	Usia	TB	BB
Usia	1.000000	0.246871	0.213561
TB	0.246871	1.000000	0.667811
BB	0.213561	0.667811	1.000000

DETERMINE CORRELATION MANUALLY USING GOOGLE SHEET

The correlation between TB and Usia from the data sample calculating manually using Google Sheet, as you can see, the value is the same as in the Jupiter notebook

USIA DAN TB

x	Usia
y	TB
xbar	20.6
ybar	176.4

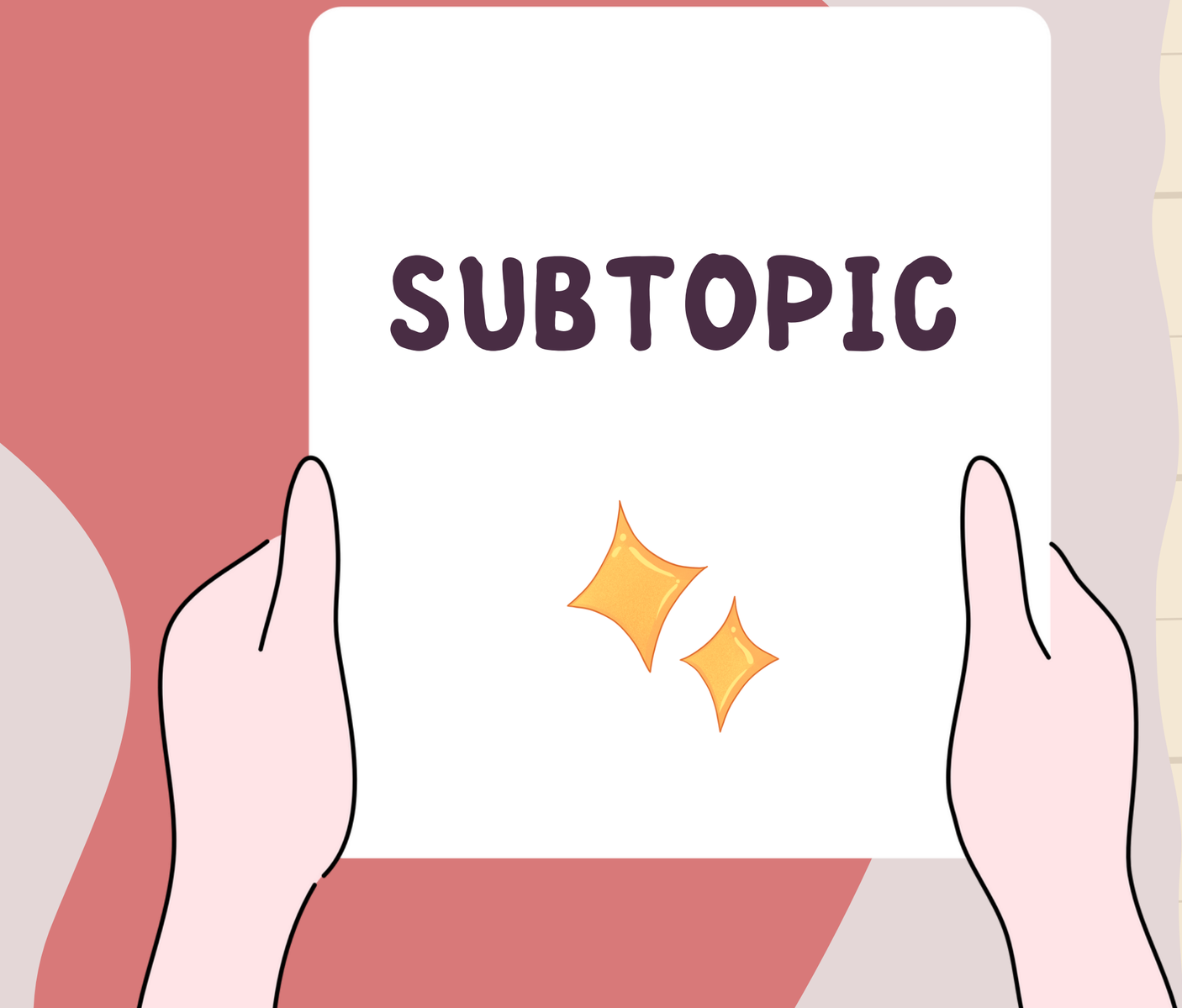
x	y	(x-xbar)	(y-ybar)	(x-xbar)(y-ybar)	(x-xbar)^2	(y-ybar)^2
20	180	-0.6	3.6	-2.16	0.36	12.96
21	182	0.4	5.6	2.24	0.16	31.36
20	170	-0.6	-6.4	3.84	0.36	40.96
21	178	0.4	1.6	0.64	0.16	2.56
21	172	0.4	-4.4	-1.76	0.16	19.36
20.6	176.4			2.8	1.2	107.2

sigma((x-xbar)(y-ybar))	2.8
sigma((x-xbar)^2)*sigma((y-ybar)^2))	128.64
akar(sigma((x-xbar)^2)*sigma((y-ybar)^2)))	11.3419575
cc	**0.2468709656**

```
kel3[['Usia','TB', 'BB']].corr()
```

	Usia	TB	BB
Usia	1.000000	0.246871	0.213561
TB	0.246871	1.000000	0.667811
BB	0.213561	0.667811	1.000000

OUTLIERS & ANOMALIES



- Difference Outliers and Anomalies
- Example Outliers and Anomalies
- Checking Outliers and Anomalies
- Handling Outliers and Anomalies

DIFFERENCE

Outliers are observations that are distant from the mean or location of a distribution. However, they don't necessarily represent abnormal behavior or behavior generated by a different process. On the other hand, **anomalies** are data patterns that are generated by different processes.

OUTLIERS

When data has an average age of 28 years. However, the data has value grandfather is 70 years old and the baby is 1 year old. Data with numbers 70 and 1 are Outliers values. Because it is beyond average.

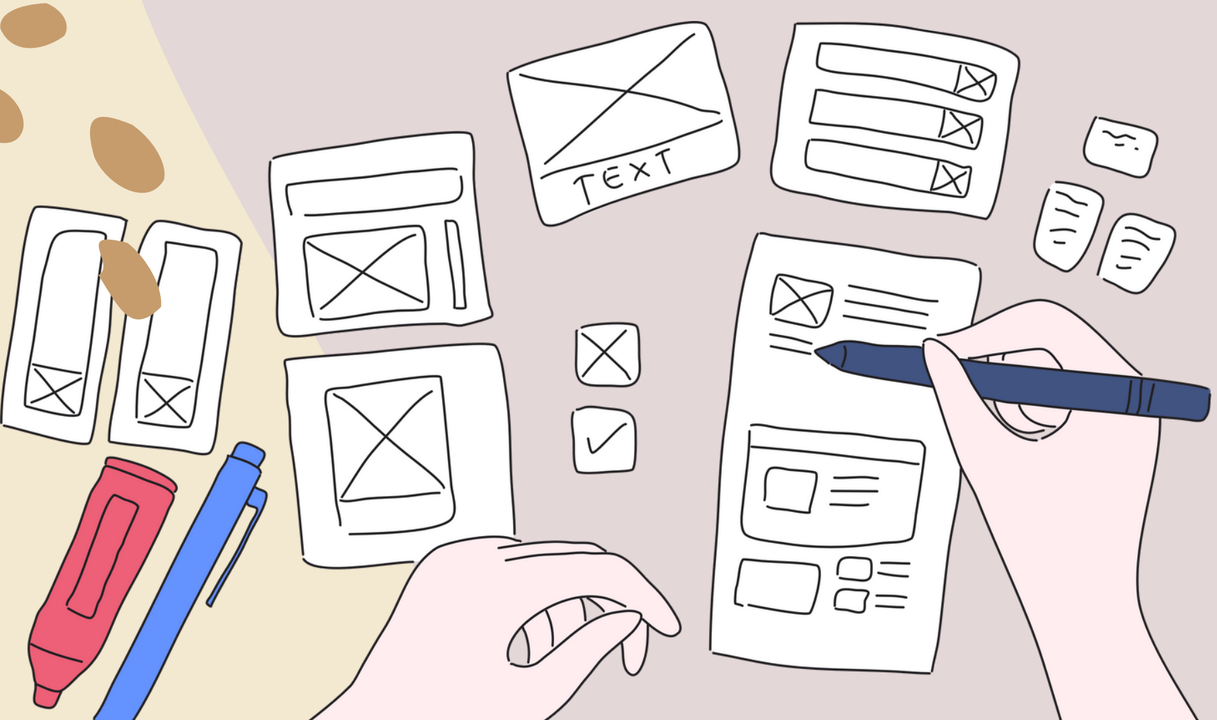
ANOMALIES

If a data input enters a person's age of -20 or 10,000 years, it can be called an anomaly. Because it doesn't make sense that age is minus 10,000 years.

CHECK ANOMALIES

We can use function describe() in python

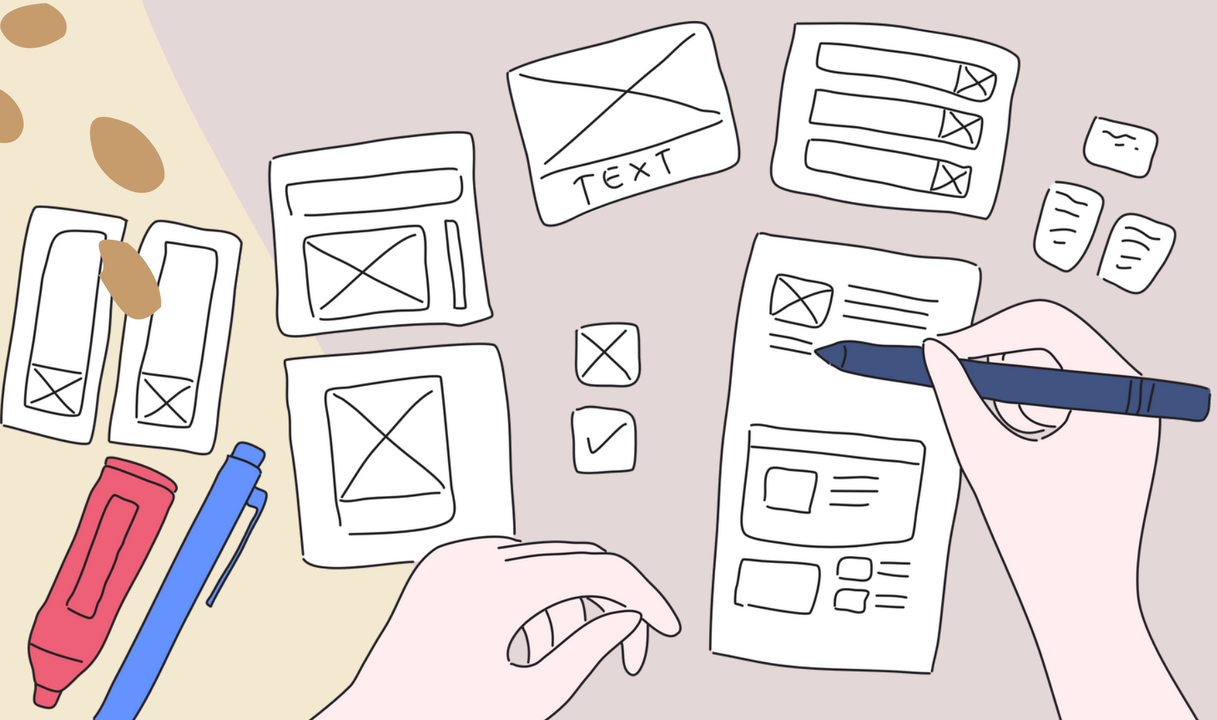
```
In [6]: # checking anomalies

        df['Age'].describe()

Out[6]: count    714.000000
        mean      29.699118
        std       14.526497
        min        0.420000
        25%       20.125000
        50%       28.000000
        75%       38.000000
        max       80.000000
        Name: Age, dtype: float64
```



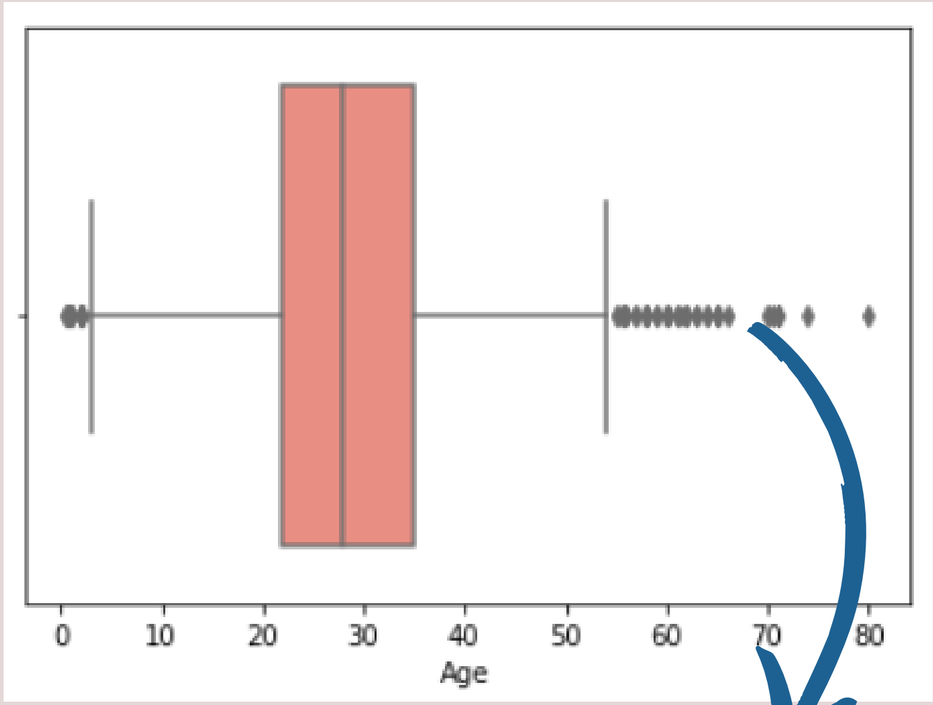
From the data above, we know that there are no anomalies because the max-age of 80 years is still normal.

CHECK OUTLIERS

The first way to check for outliers can use a boxplot.




```
In [5]: # Boxplot

        sns.boxplot(df['Age']);
```



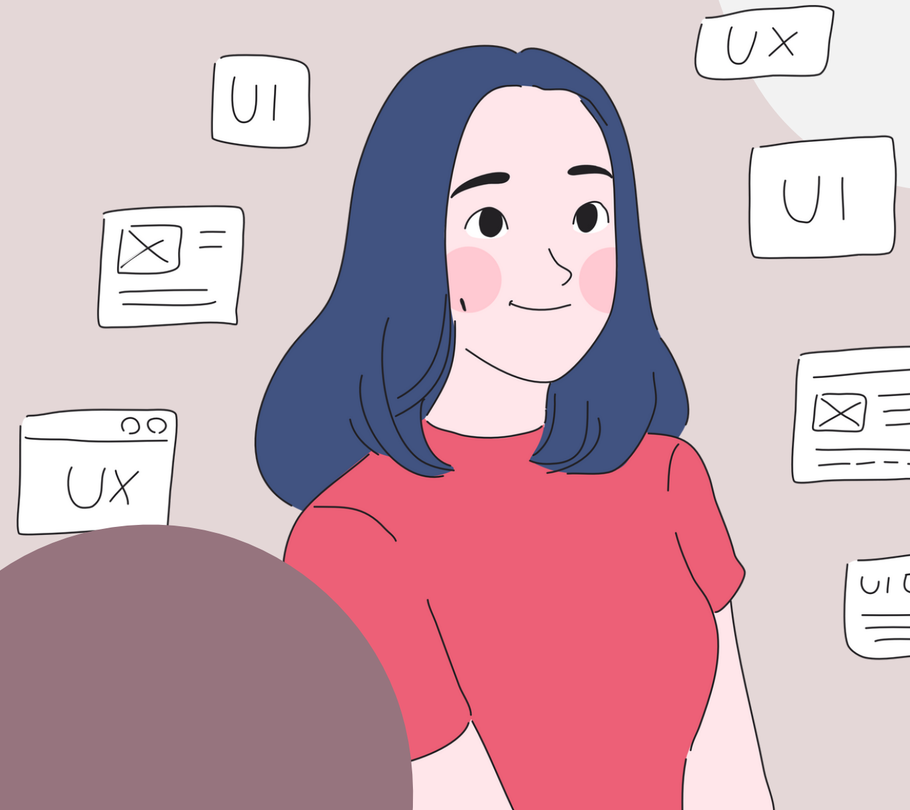
OUTLIERS

CHECK OUTLIERS (2)

We can use IQR:

IQR tells how spread the middle values are. It can tell when a value is too far from the middle.

An outlier is a point which falls more than 1,5 times the interquartile range above the third quartile or below the first quartile.



STEP CHECK OUTLIERS

1

```
In [6]: # Choose the data to be checked for outliers

        df_age_notnull = df['Age'][df['Age'].notnull()]
```

2

```
In [8]: # Finding first quartile and third quartile

        q1, q3 = np.percentile(df_age_notnull, [25,75])

        # Print Value First Quartile and Third Quartile

        print(q1, q3)
```

```
20.125 38.0
```

STEP CHECK OUTLIERS

3

```
In [10]: # Finding the IQR which is the difference between third and first quartile

         iqr = q3 - q1

         # Print Value IQR

         print(iqr)
```

```
17.875
```

4

```
In [11]: # Find lower and upper bound

         lower_bound = q1 - (1.5*iqr)
         upper_bound = q3 + (1.5*iqr)

         # Print Value lower_bound and upper_bound
         print(lower_bound)
         print(upper_bound)
```

```
-6.6875
64.8125
```

STEP CHECK OUTLIERS

5 VALUES BELOW LOWER_BOUND AND ABOVE UPPER_BOUND ARE OUTLIERS.

GROUP 3 DATA CHECK OUTLIERS

nama	usia	tb	bb
Axel	20	180	60
Yogi	21	182	66
Dian	20	170	58
Yesaya	21	178	70
Dwi	21	172	50

Are there value outliers in column 'TB' in the data above?

GROUP 3 DATA CHECK OUTLIERS

q1	172
q3	180
iqr	8
lower bound	160
upper bound	192

q1 = PERCENTILE(Start_value:End_value,0.25)

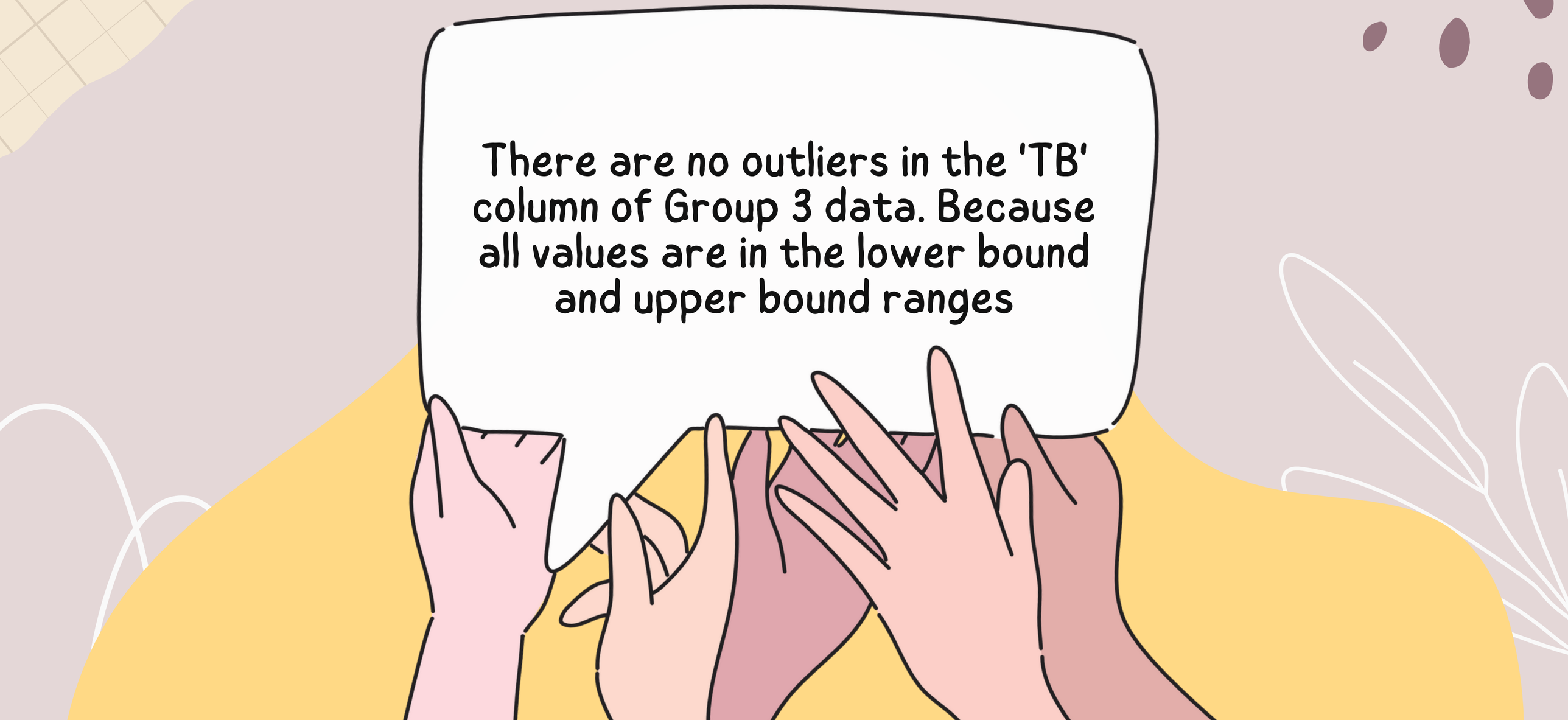
q1 = PERCENTILE(Start_value:End_value,0.75)

iqr = q3 - q1

lower_bound = q1 - (1.5*iqr)

lower_bound =q3 + (1.5*iqr)

Conclusion !!


There are no outliers in the 'TB' column of Group 3 data. Because all values are in the lower bound and upper bound ranges

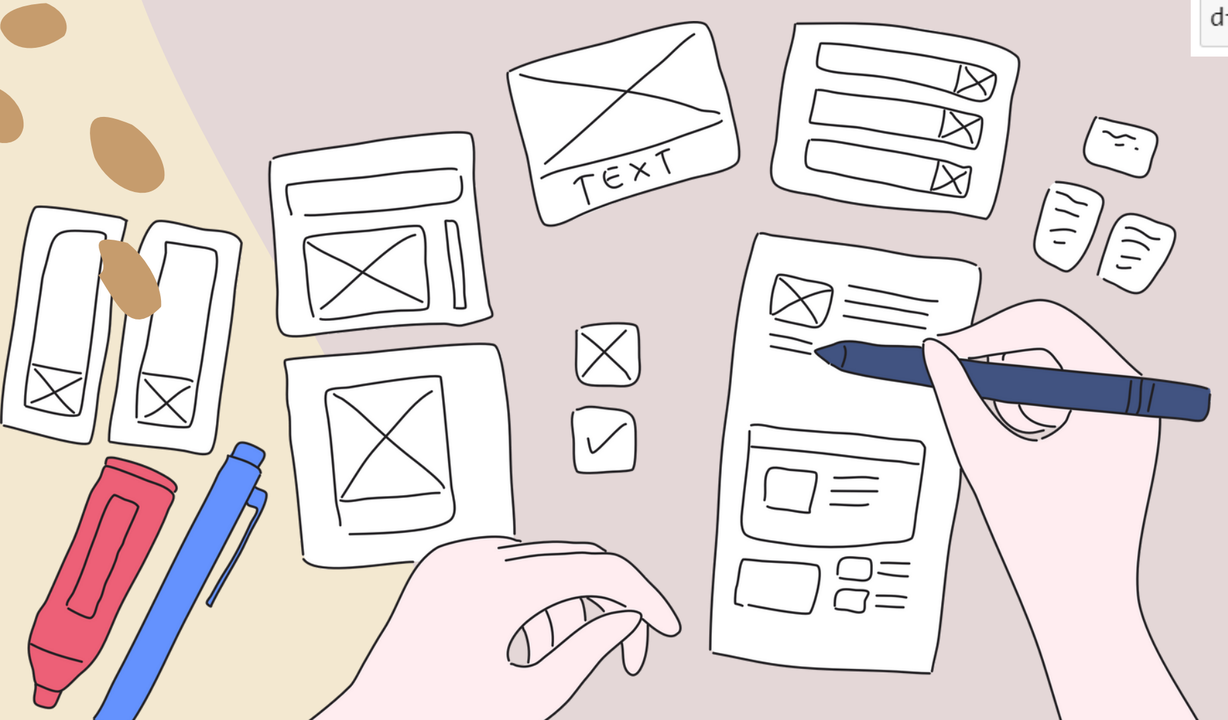
HANDLING OUTLIERS

To handle outliers, we can select data above the lower bound and below the upper bound. In this way, outlier data will delete.

HANDLING OUTLIERS

Selecting data 'Age' that is between the lower bound and upper bound.

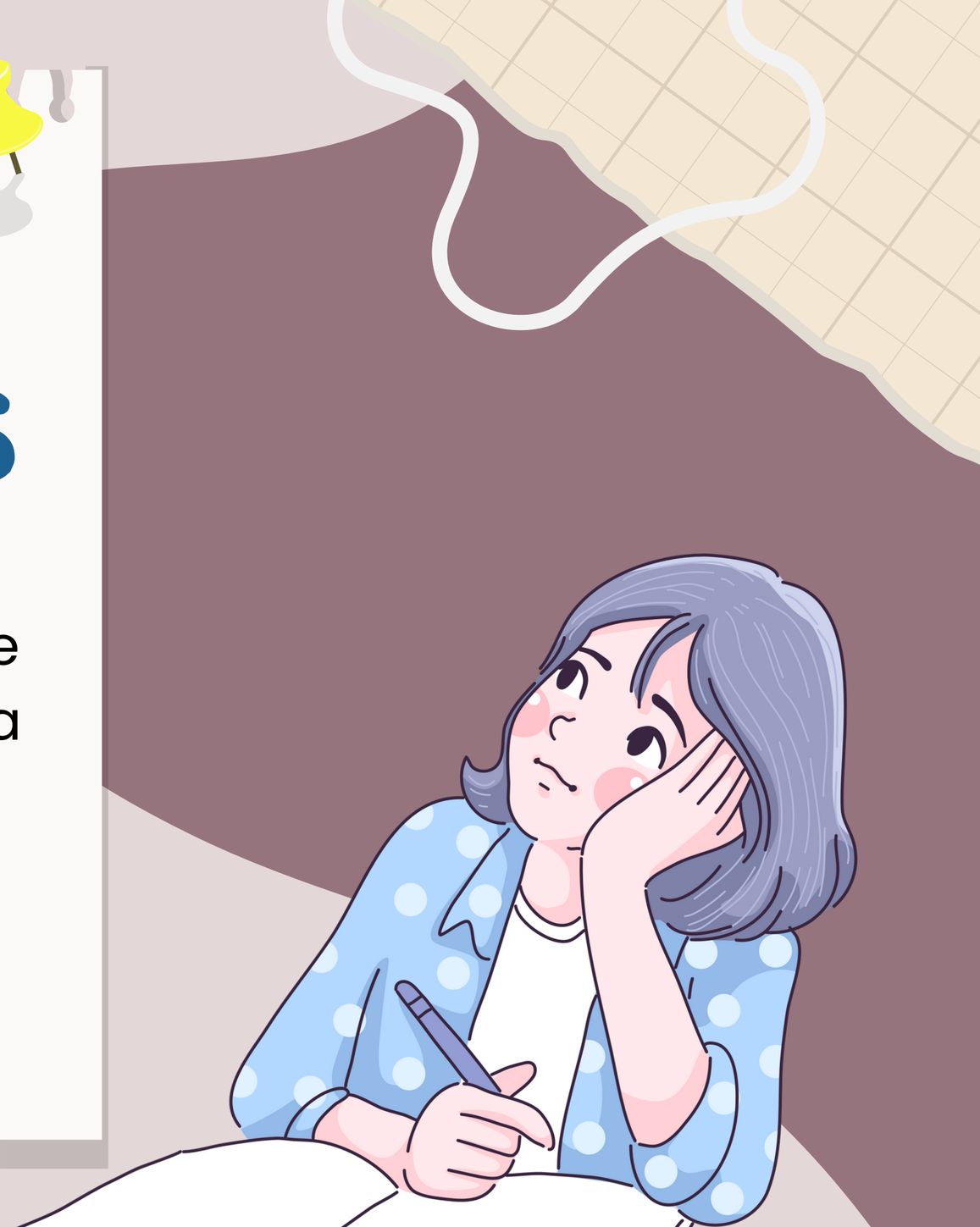
```
# Handling Outliers

# Delete outliers

df_age_new = df['Age'][(df['Age'] > lower_bound) & (df['Age'] < upper_bound)]
```



Data 'Age' has been cleared from outliers.

HANDLING MISSING VALUES

Before handling missing values, we need to determine the type of data to handle. Including numerical or categorical.



EXAMPLE NUMERICAL MISSING VALUES

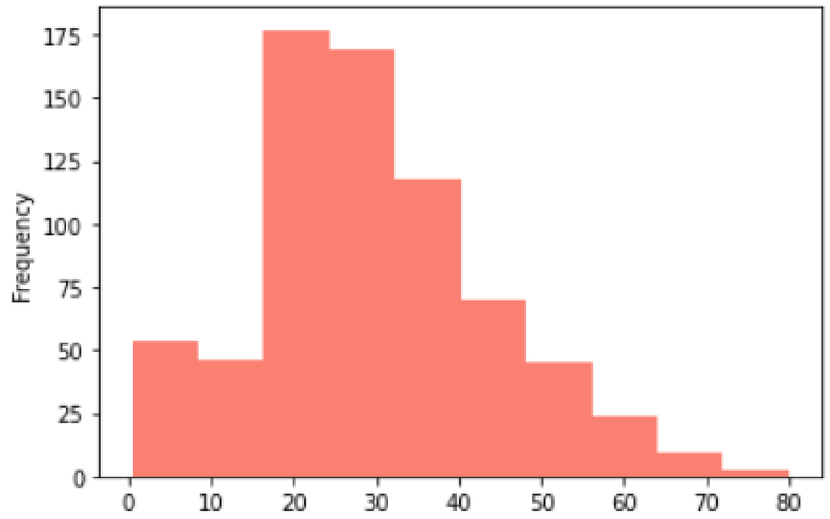
Case:
We will handle missing values in the 'Age' column of the Titanic data.

1.

```
In [19]: # handling missing value

         # numerical

         # Total missing value
         df['Age'].isnull().sum()

Out[19]: 177
```

2.

```
df['Age'].plot(kind='hist', color="salmon");
```



NOTES:

- If Normal Distribution, handling value using mean
- If curtosis or skewness, handling value using median

EXAMPLE NUMERICAL MISSING VALUES

Case:
We will handle missing values in the 'Age' column of the Titanic data.

3.

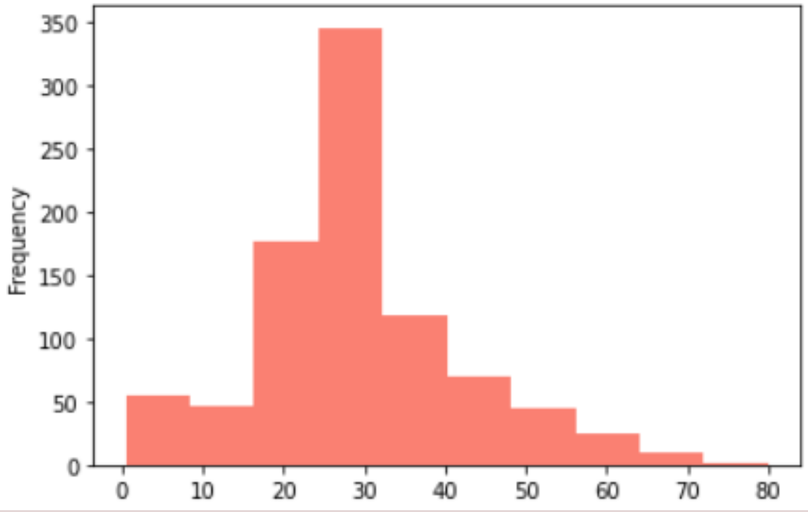
```
In [21]: # Find Median use function median()

         val = df['Age'].median()

         # Then insert the median value into the missing value.

         df['Age'] = df['Age'].fillna(val)
```

4.

```
In [23]: # Check Again missing values in column 'Age'

         df['Age'].isna().sum()

Out[23]: 0
```

5.

```
df['Age'].plot(kind='hist', color="salmon");
```



EXAMPLE CATEGORICAL MISSING VALUES

Case:
We will handle missing values in the 'Embarked' column of the Titanic data.

1.

```
In [25]: # categorical

         # See values in column 'Embarked'

         df['Embarked'].value_counts()

Out[25]: S    644
         C    168
         Q     77
         Name: Embarked, dtype: int64
```

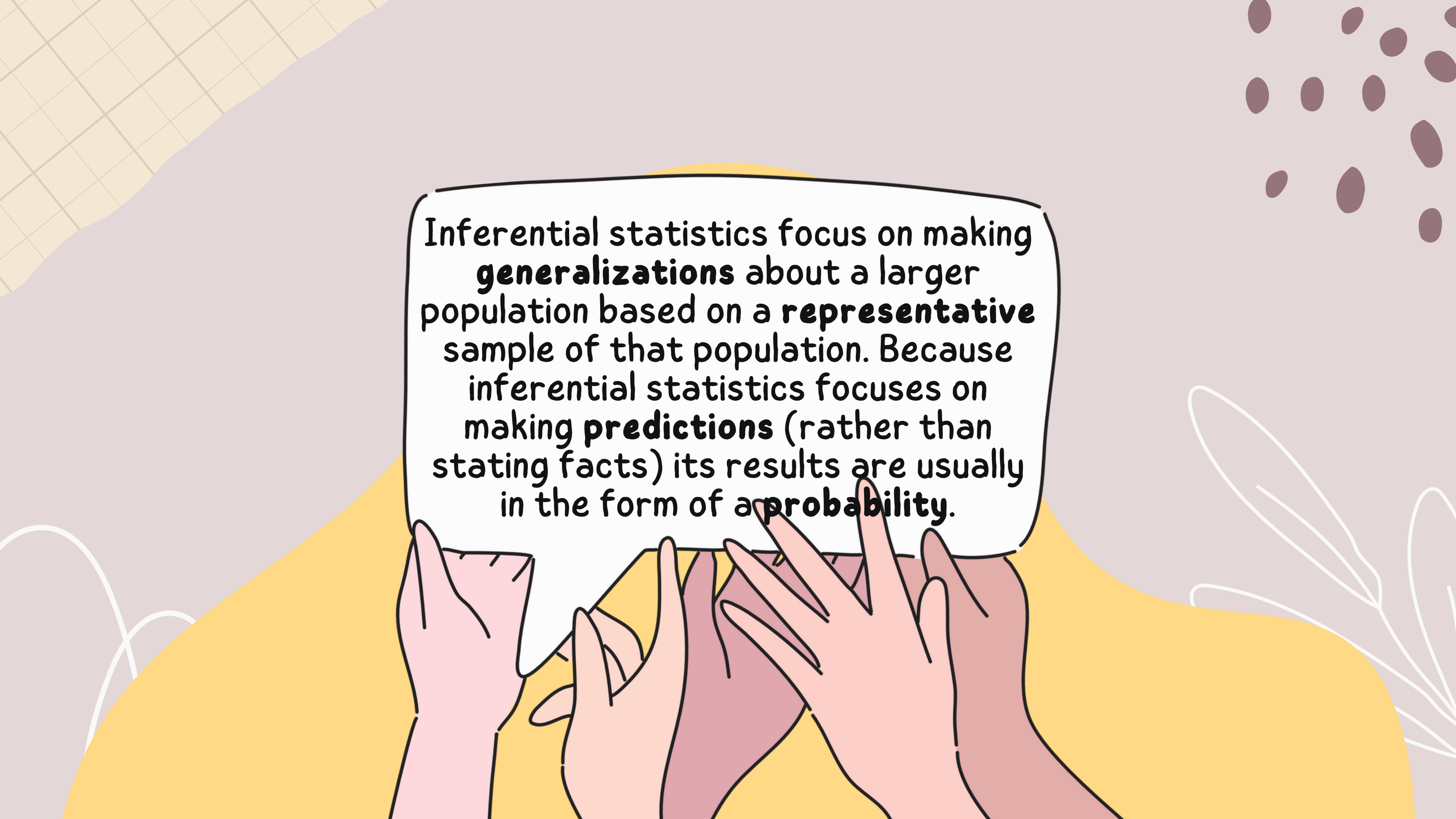
2.

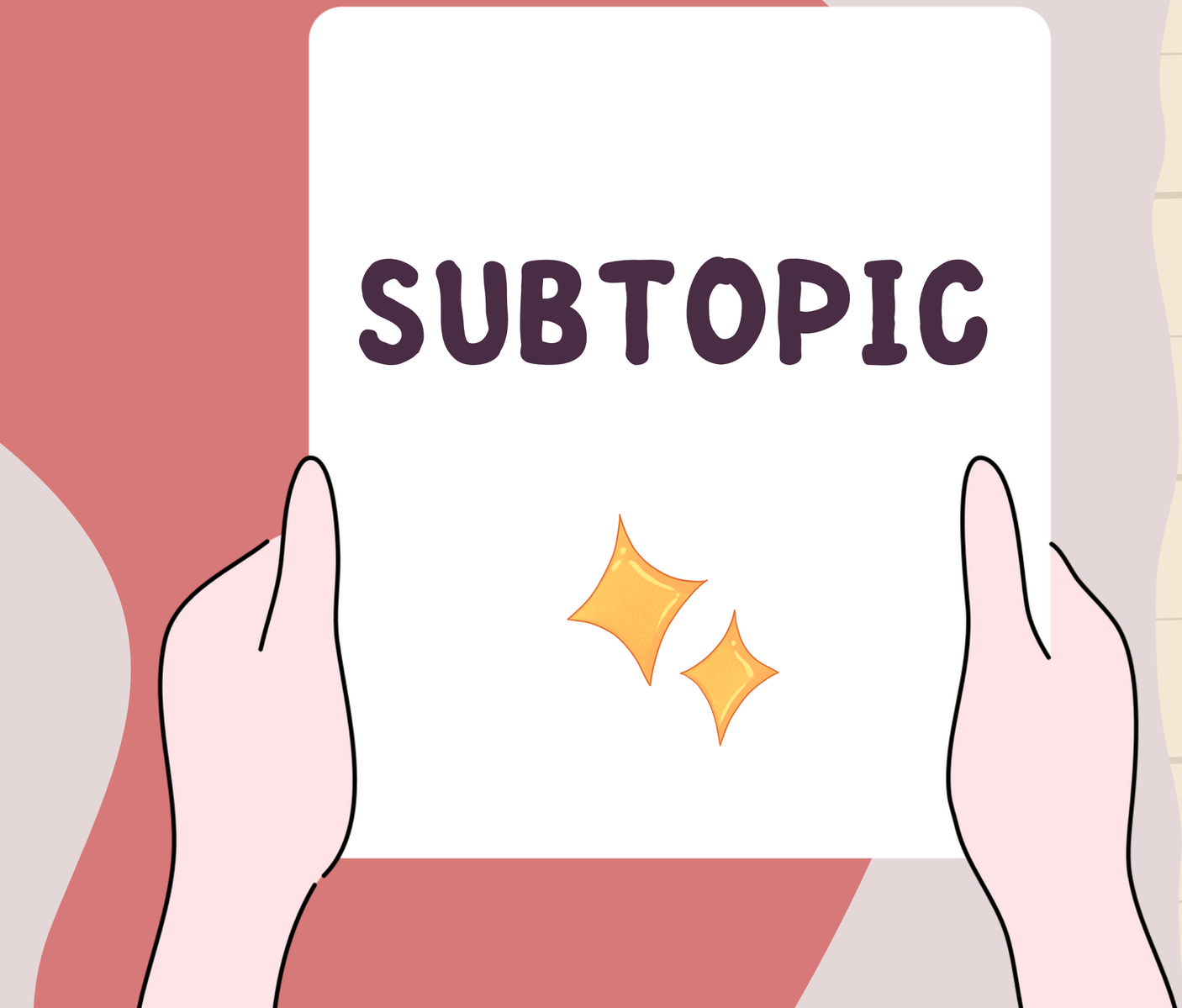
```
In [27]: # Find Mode use function mode()

         val = df['Embarked'].mode().values[0]

         # Then insert the mode value into the missing value

         df['Embarked'] = df['Embarked'].fillna(val)
```

3.

```
In [28]: # Check Again missing values in column 'Embarked'

         df['Embarked'].isna().sum()

Out[28]: 0
```

INFERENTIAL STATISTICS


Inferential statistics focus on making **generalizations** about a larger population based on a **representative** sample of that population. Because inferential statistics focuses on making **predictions** (rather than stating facts) its results are usually in the form of a **probability**.



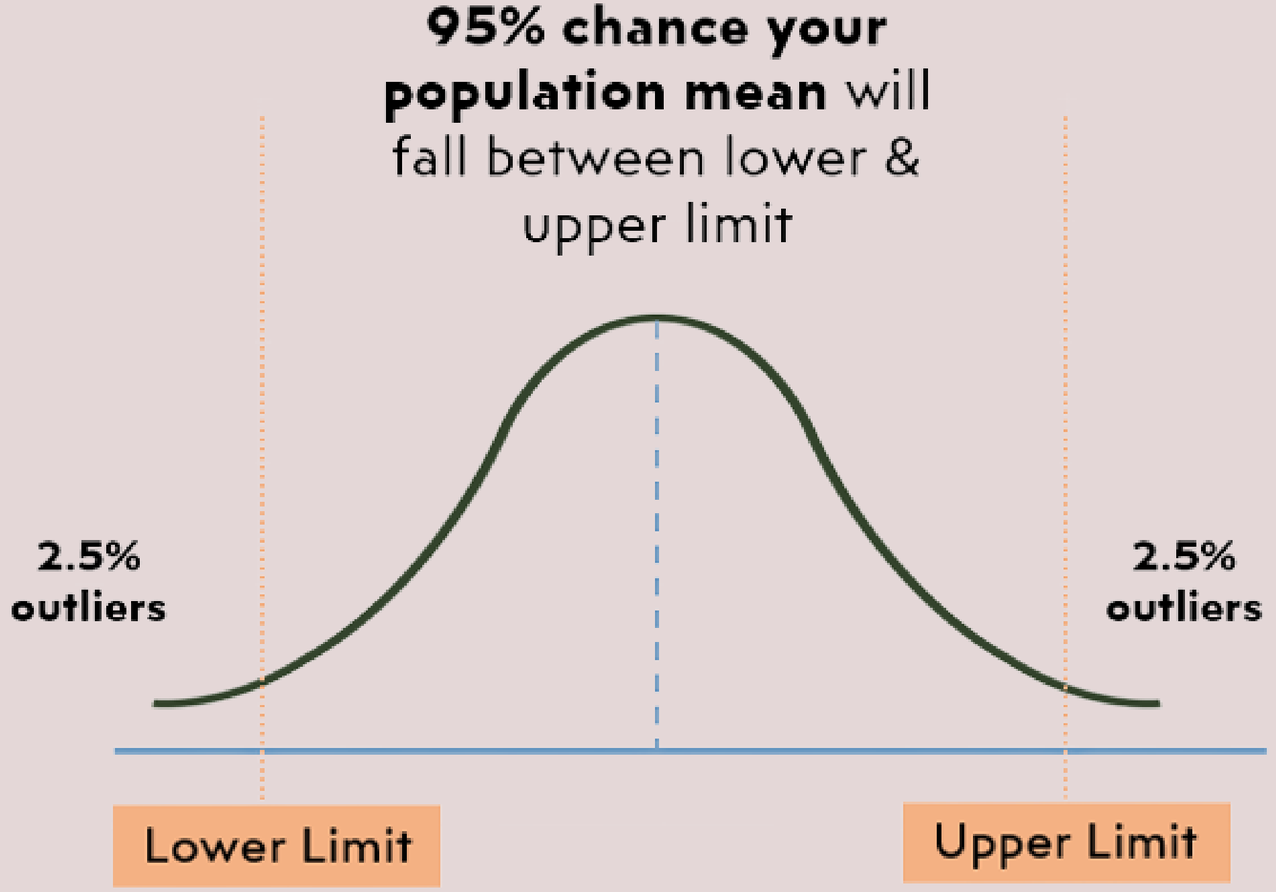
Confidents Interval

Hypothesis Testing

CONFIDENCE INTERVAL

a range of estimates for an unknown parameter. A confidence interval is computed at a designated confidence level; the 95% confidence level is most common

ILLUSTRATION OF 95% CONFIDENCE INTERVAL



STEP :

- Get random sample from our data
- Calculate mean from our data
- Make mean bootstrap from sample
- Iterate like 1000 times to look mean for every bootstrap
- Visualize our mean bootstrap
- Get 95% confidence level by used percentile 2.5 and 97.5

STEP IN PYTHON :

Case:
Calculate from confidence level from coffee drinkers data

1.

```
coffee_full = pd.read_csv('coffee_dataset.csv')
coffee_red = coffee_full.sample(200)
coffee_red
```

2.

```
#proportion of coffee drinkers in sample
mean_drinker = coffee_red['drinks_coffee'].mean()
mean_non_drinker = 1 - mean_drinker
```

```
print(mean_drinker)
print(mean_non_drinker)
```

3.

```
#bootstrap sample of 200 draws
bootsamp = coffee_red.sample(200, replace = True)
bootsamp['drinks_coffee'].mean()
```

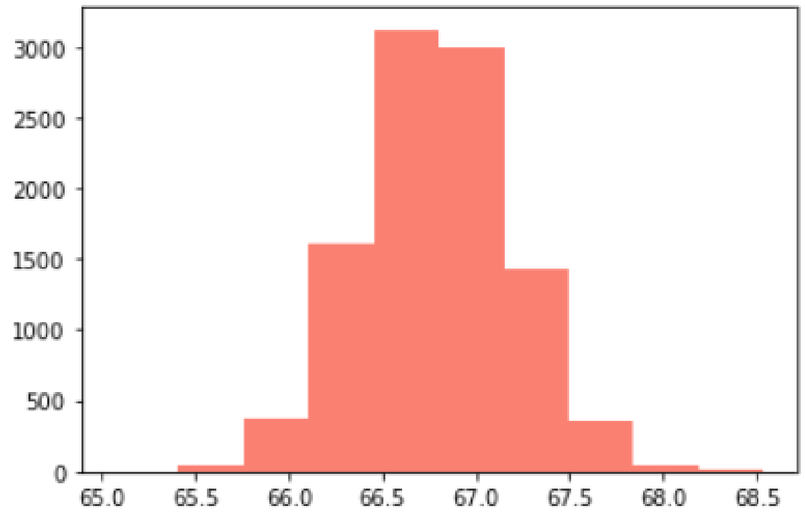
```
0.605
```

STEP IN PYTHON :

4.

```
# Bootstrap sample of 10,000 times
boot_means = []
for _ in range(10000):
    bootsamp = coffee_sample.sample(200, replace=True)
    mean = bootsamp[bootsamp['drinks_coffee'] == False]['height'].mean()
    boot_means.append(mean)

plt.hist(boot_means, color='salmon')
```

```
(array([3.000e+00, 4.900e+01, 3.720e+02, 1.608e+03, 3.122e+03, 2.994e+03,
        1.436e+03, 3.630e+02, 4.600e+01, 7.000e+00]),
 array([65.0634378 , 65.41077637, 65.75811493, 66.1054535 , 66.45279207,
        66.80013064, 67.14746921, 67.49480777, 67.84214634, 68.18948491,
        68.53682348]),
 <BarContainer object of 10 artists>)
```



STEP IN PYTHON :

5.

```
#obtain 95% confidence interval
np.percentile(boot_means, 2.5), np.percentile(boot_means, 97.5)
```

```
(65.99291328157521, 67.58402738281573)
```

```
#compare between full data and sample data mean
coffee_full[coffee_full['drinks_coffee']==False]['height'].mean()
```

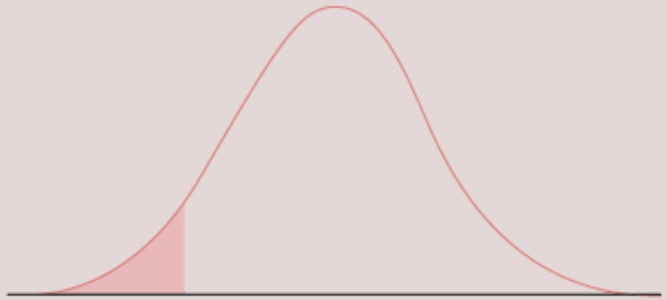
```
66.44340776214703
```

HYPOTHESIS TESTING

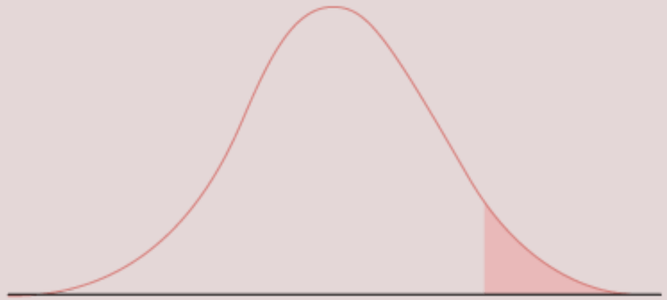
Hypothesis testing is an act in statistics whereby an analyst tests an assumption regarding a population parameter.

HYPOTHESIS TESTING

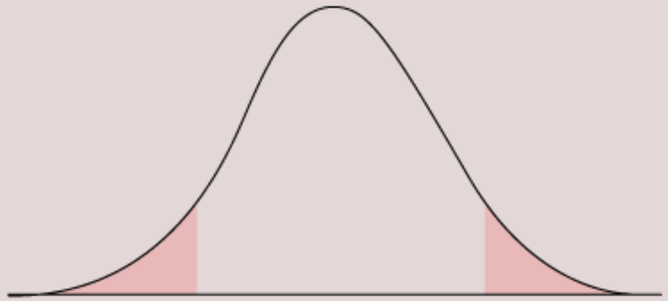
$H_o : \mu = claim$
$H_1 : \mu < claim$



$H_o : p = claim$
$H_1 : p > claim$



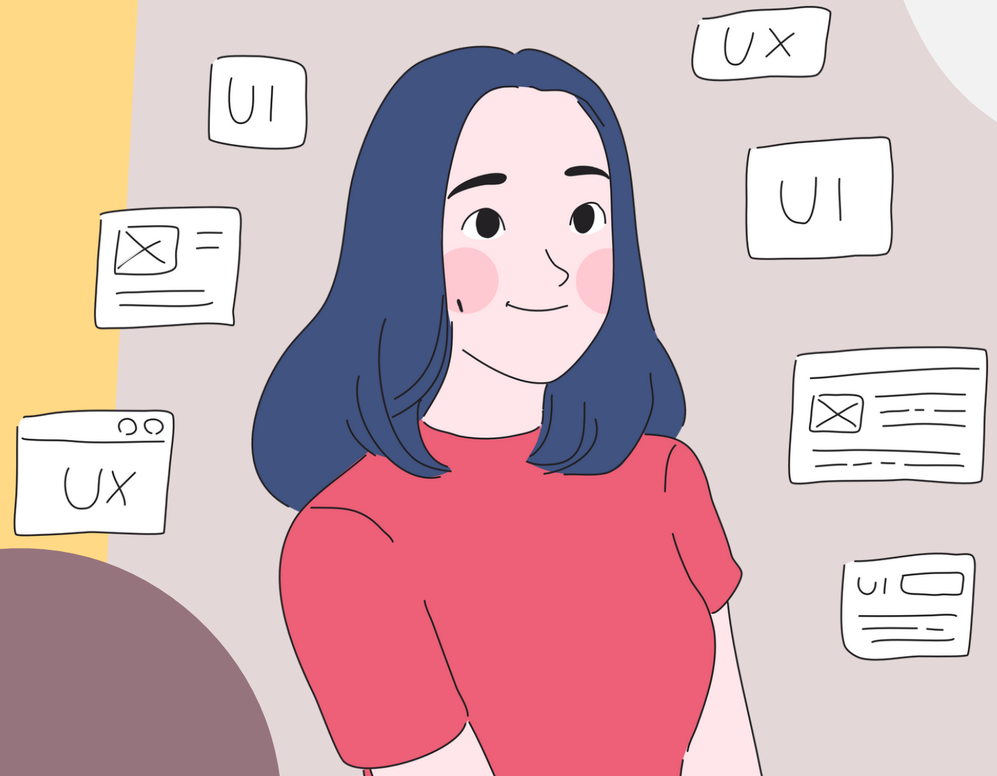
$H_o : \beta = claim$
$H_1 : \beta \neq claim$



T-TEST STATISTICS

The t-test formula is applied to the sample population. The t-test formula depends on the mean, variance, and standard deviation of the data being compared. There are 3 types of t-tests:

- One-sample test
- Independent sample t-test
- Paired samples t-test



ONE-SAMPLE TEST

$$t = \frac{\bar{x} - \mu}{\frac{\sigma}{\sqrt{n}}}$$

where,

'x' bar is the mean of the sample,

μ is the assumed mean,

σ is the standard deviation

and n is the number of observations

INDEPENDENT SAMPLE T-TEST

$$d = \frac{\bar{x}_1 - \bar{x}_2}{\sqrt{\left(\frac{s_1^2}{n_1} + \frac{s_2^2}{n_2}\right)}}$$

$$df = n_1 + n_2 - 1$$

$$s_1^2 = \frac{\sum_{i=1}^{n_1} (x_i - \bar{x}_1)^2}{n_1 - 1}$$

$$s_2^2 = \frac{\sum_{j=1}^{n_2} (x_j - \bar{x}_2)^2}{n_2 - 1}$$

PAIRED SAMPLES T-TEST

$$t = \frac{\Sigma(x_1 - x_2)}{\frac{s}{\sqrt{n}}}$$

where

t = Student's t-test

$x_1 - x_2$ = Difference mean of the pairs

s= standard deviation

n = sample size

T-TEST EXAMPLE

1.

```
import pandas as pd
from scipy import stats
from statsmodels.stats import weightstats as stests
```

```
df = pd.read_csv('blood_pressure.csv')
```

```
df.head()
```

2.

```
df[['bp_before', 'bp_after']].describe()
```

	bp_before	bp_after
count	120.000000	120.000000
mean	156.450000	151.358333
std	11.389845	14.177622

T-TEST EXAMPLE

3.

```
df.shape
```

```
(120, 5)
```

4.

```
ttest, pval = stats.ttest_rel(df['bp_before'], df['bp_after'])
print(pval)
```

```
0.0011297914644840823
```

T-TEST EXAMPLE

5.

```
if pval < 0.05 :
    print('reject null hypothesis')
else :
    print('accept null hypothesis')
```

```
reject null hypothesis
```

THANK YOU