In [4]:

```
import numpy as np
import pandas as pd
import seaborn as sns
```

In [5]:

```
import scipy.stats as scs
import matplotlib.pyplot as plt
%matplotlib inline
```

In [6]:

```
df = pd.read_excel(r"C:/Users/22jha/OneDrive/Desktop/SA Assignment/SA dataset.xlsx")
df.head()
```

Out[6]:

	Gender	AGE	CASTE	RELIGN	MTONGUE	OCCU	INCOME	AREA	WRITE	READ	MATH
0	T	16	SC	H	T	C	4000	0.0	6.0	21.0	5.0
1	F	35	SC	H	T	C	4000	0.0	17.0	18.0	7.0
2	T	15	SC	H	T	C	4500	2.0	22.0	31.0	25.0
3	F	20	OT	H	T	C	4500	0.0	17.0	19.0	17.0
4	F	29	SC	H	U	C	3000	0.0	16.0	24.0	12.0

In [8]:

```
(df['READ']+df['WRITE']+df['MATH']==df['TOTAL']).value_counts()
```

Out[8]:

```
True    1000
dtype: int64
```

In [9]:

```
(df['READ']+df['WRITE']+df['MATH']==df['TOTAL'])
```

Out[9]:

```
0       True
1       True
2       True
3       True
4       True
        ...
995     True
996     True
997     True
998     True
999     True
Length: 1000, dtype: bool
```

In [10]:

```
df.isnull().sum()
```

Out[10]:

```
Gender     0
AGE        0
CASTE      0
RELIGN     0
MTONGUE    0
OCCU       0
INCOME     0
AREA       0
WRITE      0
READ       0
MATH       0
TOTAL      0
dtype: int64
```

In [ ]:

```
##Q 1 Testing the null hypothesis - Average of the total is 56.
## H0: Average Total Marks = 56

## H1: Average Total Marks != 56
```

In [11]:

```
μ = 56
totalmean = df['TOTAL'].mean()
tstddev = df['TOTAL'].std()
n = 1000
dof = n - 1
α = 0.05

print('Mean = ',totalmean)
print('Standard Deviation = ',tstddev)
```

```
Mean =  56.063770000000005
Standard Deviation =  15.682784855628404
```

In [13]:

```
totalvalue = (totalmean - μ)/(tstddev/n**0.5)
print('totalvalue =',totalvalue)
```

```
totalvalue = 0.12858586548586154
```

In [14]:

```
t = scs.t.sf(totalvalue,dof)*2
t
```

Out[14]:

```
0.8977112323608465
```

In [ ]:

```
## Observation - We observe t>α, so H0 cannot be rejected. Hence, Average of the total mark
```

In [ ]:

```
## Q2. Test the null hypothesis that the average marks in Math is less than or equal to 15

## H0: Average Maths Marks <= 15

## H1: Average Maths Marks > 15
```

In [16]:

```
μ_math = 15
mathmean = df['MATH'].mean()
mathstddev = df['MATH'].std()
n_math = 1000
dof_math = n_math - 1
α_math = 0.05
print('Mean of Maths Marks = ',mathmean)
print('Standard Deviation for Maths Marks = ',mathstddev)
```

```
Mean of Maths Marks =  16.49527
Standard Deviation for Maths Marks =  7.033626890469411
```

In [18]:

```
tval_math = (mathmean - μ_math)/(mathstddev/n_math**0.5)
print('TotalValue =',tval_math)
```

TotalValue = 6.722646780321907

In [19]:

```
t_maths = scs.t.sf(tval_math,dof_math)*2
t_maths
```

Out[19]:

2.996032120322512e-11

In [ ]:

```
ervation - We observe that t_math < a_math, so We reject the Null Hypothesis. Hence the mean
```

In [ ]:

```
## Q3. The State Authorities feel that the women performed better than men.
##     In order to test this, they wanted to carry out a one-sided hypothesis test.
##     First calculate the average Total Score for all the 1000 beneficiaries in your sample
##     Then calculate the proportion of women who scored more than this average.
##     Consider this as the sample proportion p. using this value of p, test the null hypoth
```

In [20]:

```
totalmean = df.TOTAL.mean()
print('Average total score =',round(totalmean,2))
```

Average total score = 56.06

In [22]:

```
df['Performance']=np.where(df['TOTAL']>totalmean,'HIGH','LOW')
df['Performance'].value_counts()
HL = df.groupby(['Performance','Gender']).size().unstack()
print(HL)
```

```
Gender         F    T
Performance
HIGH         360  154
LOW          347  139
```

In [ ]:

```
## Solution 3 - Taking women population as the total base for calculating High Women Perfor
```

In [49]:

```
totalwomen = 360/(360+347)
print('p (Proportion of women who scored more than the average) = ',round(totalwomen,2))
```

```
p (Proportion of women who scored more than the average) =  0.51
```

In [50]:

```
lowwomen = 1-totalwomen
print('Population other than high performing woman = ',round(lowwomen,2))
```

```
Population other than high performing woman =  0.49
```

In [51]:

```
se_women = np.sqrt(totalwomen*lowwomen/(360+347))
print('Standard Error of High performing woman = ',round(se_women,4))
```

```
Standard Error of High performing woman =  0.0188
```

In [52]:

```
zscore = (totalwomen-0.5)/se_high
print('Calculated score value of z = ',round(zscore,2))
```

```
Calculated score value of z =  0.46
```

In [53]:

```
zvalue = np.abs(scs.norm.ppf(0.05))
print('Tabulated z-score at 5% level of Significance = ',round(zvalue,2))
```

```
Tabulated z-score at 5% level of Significance =  1.64
```

In [54]:

```
pvaluess = scs.norm.cdf(zval)
print(pvaluess)
```

```
0.9500000000000001
```

In [ ]:

```
## Solution 3 - Taking 'High' population as the total base for calculating High Women Perfo
```

In [26]:

```
highperfw = 360/(360+154)
print('p (Proportion of women who scored more than the average) = ',round(highperfw,2))
```

```
p (Proportion of women who scored more than the average) =  0.7
```

In [27]:

```
lowperf = 1-highperfw
print('Population other than high performing woman = ',round(lowperf,2))
```

Population other than high performing woman = 0.3

In [29]:

```
se_high = np.sqrt(highperfw*lowperf/(360+154))
print('Standard Error of High performing woman = ',round(se_high,4))
```

Standard Error of High performing woman = 0.0202

In [31]:

```
zscr = (highperfw-0.5)/se_high
print('Calculated score value of z = ',round(zscr,2))
```

Calculated score value of z = 9.92

In [32]:

```
zval = np.abs(scs.norm.ppf(0.05))
print('Tabulated z-score at 5% level of Significance = ',round(zval,2))
```

Tabulated z-score at 5% level of Significance = 1.64

In [33]:

```
pvalue = scs.norm.cdf(zval)
print(pvalue)
```

0.9500000000000001

In [ ]:

```
## Observation - We can observe that pvalue> 0.05. Hence, we fail to reject the Null Hypoth
```

In [ ]:

```
## Q4. It is always claimed that those who are taught in their mother tongue perform better
##     The teaching in the class is in Tamil language.
##     Test the null hypothesis that there is no difference in the performance of those
##     beneficiaries whose mother tongue is Tamil and those whose mother tongue is NOT Tamil
##     Use Total score as a measure for performance.
```

In [ ]:

```
## H0: Average Total Score of Tamil speaking student  = Average Total score of Non Tamil sp

## H1: Average Total Score of Tamil speaking student != Average Total score of Non Tamil sp
```

In [34]:

```
tamillang = df[df['MTONGUE'] == 'T']['TOTAL']
notamillang = df[df['MTONGUE'] !='T']['TOTAL']
α_tamil = 0.05


print('Tamil Score is ',tamillang)
print('Non Tamil Score is',notamillang)
```

```
Tamil Score is  0      32.0
1      42.0
2      78.0
3      53.0
5      56.0
       ...
993    56.0
994     7.0
995    61.0
996    73.0
999    68.0
Name: TOTAL, Length: 826, dtype: float64
Non Tamil Score is 4      52.0
8      33.0
10     86.0
17     45.0
22     42.0
       ...
984    77.0
987    66.0
990    62.0
997    85.0
998    63.0
Name: TOTAL, Length: 174, dtype: float64
```

In [35]:

```
pvalt = scs.ttest_ind(tamillang, notamillang)
pvalt
print('Pvalue = ',pvalt.pvalue)
print('Statistics = ',pvalt.statistic)
```

```
Pvalue =  0.038281607571874356
Statistics =  -2.074576994737671
```

In [ ]:

```
nces hence we can say that mother tongue is a changing factor in the students' performances
```

In [ ]:

```
## Q5. We would like to test if the two variables Age and Performance (measured by Total Sc
##   In order to test this hypothesis, divide the beneficiaries into
##   4 categories namely Low, Good, Very Good and Excellent based on the Total Score.
##   The cut-offs for each of the categories is left to you.
##   Similarly, divide the beneficiaries into 3 groups (Grp 1, Grp 2 and Grp 3) based on th
##   Again, the cut-offs for each group is left to you.
##   Test the null hypothesis that the two attributes, Performance and Age are independent
##   using the Performance categories and Age groups created by you.
```

In [ ]:

```
## H0: Performance of Group A = Performance of Group B = Performance of Group C

## H1: Performance of Group A != Performance of Group B != Performance of Group C
```

In [37]:

```
performance = []
for x in df ['TOTAL'].values:
    if x < df['TOTAL'].quantile(.25):
        performance.append('LOW')
    elif x >= df['TOTAL'].quantile(.25) and x < df['TOTAL'].quantile(.50):
        performance.append('GOOD')
    elif x >= df['TOTAL'].quantile(.50) and x < df['TOTAL'].quantile(.75):
        performance.append('VERY GOOD')
    else:
        performance.append('EXCELLENT')
df['Performance'] = performance
```

In [39]:

```
agegroup = []
for x in df['AGE'].values:
    if x < df['AGE'].quantile(0.33):
        agegroup.append('Group1')
    elif x >= df ['AGE'].quantile(0.33) and x < df ['AGE'].quantile(0.67):
        agegroup.append('Group2')
    else:
        agegroup.append('Group3')
df['AGE_Group'] = agegroup
```

In [47]:

```
independent = pd.crosstab(df['Performance'], df['AGE_Group'])
independent
```

Out[47]:

AGE_Group	Group1	Group2	Group3
Performance			
EXCELLENT	100	86	90
GOOD	91	83	89
LOW	48	77	110
VERY GOOD	66	74	86

In [42]:

```
stat,Pvalue,ddof,array = scs.chi2_contingency(independent)
Pvalue
```

Out[42]:

```
0.0020978757866773902
```

In [43]:

```
stat
```

Out[43]:

```
20.675397756622452
```

In [44]:

```
ddof
```

Out[44]:

```
6
```

In [46]:

```
array
```

Out[46]:

```
array([[ 84.18 ,  88.32 , 103.5  ],
       [ 80.215,  84.16 ,  98.625],
       [ 71.675,  75.2  ,  88.125],
       [ 68.93 ,  72.32 ,  84.75 ]])
```

In [45]:

```
scs.chi2_contingency(independent)
```

Out[45]:

```
(20.675397756622452,
 0.0020978757866773902,
 6,
 array([[ 84.18 ,  88.32 , 103.5  ],
        [ 80.215,  84.16 ,  98.625],
        [ 71.675,  75.2  ,  88.125],
        [ 68.93 ,  72.32 ,  84.75 ]]))
```

In [ ]:

```
ue < 0.05, so we can reject the Null Hypothesis. Hence, this implies Performance and Age ar
```