To import necessary modules

In [1]:
```
import pandas as pd
import numpy as np
import warnings
warnings.filterwarnings("ignore")
```

In [2]:
```
test=pd.read_csv(r"K:\Desktop\NIIT\tables\DS1_C4_S4_Test_Scores_Data_Practice.csv")
non_unique=test.columns[test.columns!="student_id"]
```

In [3]:
```
school=test[non_unique]
```

Task1: To help find all the duplicate records after thier first occurence

In [4]:
```
school[school.duplicated(keep="first")]
```

Out[4]:

	school	school_setting	school_type	classroom	teaching_method	gender	lunch	pretest	posttest
4	ANKYI	Urban	Non-public	6OL	Standard	Male	Does not qualify	64.0	76.0
19	ANKYI	Urban	Non-public	6OL	Standard	Male	Does not qualify	64.0	73.0
38	ANKYI	Urban	Non-public	ZNS	Standard	Female	Does not qualify	59.0	69.0
50	CCAAW	Suburban	Non-public	2B1	Experimental	Male	Qualifies for reduced/free lunch	59.0	74.0
106	CCAAW	Suburban	Non-public	PGK	Standard	Female	Does not qualify	73.0	80.0
...	...	...	...	...	...	...	...	...	...
2058	ZOWMK	Urban	Public	Q0E	Experimental	Female	Qualifies for reduced/free lunch	38.0	53.0
2067	ZOWMK	Urban	Public	Q0E	Experimental	Male	Qualifies for reduced/free lunch	37.0	53.0
2070	ZOWMK	Urban	Public	Q0E	Experimental	Female	Qualifies for reduced/free lunch	38.0	53.0
2080	ZOWMK	Urban	Public	QA2	Standard	Female	Qualifies for reduced/free lunch	45.0	51.0
2128	ZOWMK	Urban	Public	ZBH	Standard	Female	Does not qualify	39.0	55.0

79 rows × 9 columns

Task2: To find all duplicates of school type teaching method pre test and post test parameters

In [5]:
```
school[school.duplicated(["school_type","teaching_method","pretest","posttest"],keep="first")]
```

Out[5]:

	school	school_setting	school_type	classroom	teaching_method	gender	lunch	pretest	posttest
4	ANKYI	Urban	Non-public	6OL	Standard	Male	Does not qualify	64.0	76.0
14	ANKYI	Urban	Non-public	6OL	Standard	Female	Does not qualify	64.0	77.0
19	ANKYI	Urban	Non-public	6OL	Standard	Male	Does not qualify	64.0	73.0
26	ANKYI	Urban	Non-public	ZNS	Standard	Female	Does not qualify	60.0	70.0
35	ANKYI	Urban	Non-public	ZNS	Standard	Female	Does not qualify	60.0	68.0
...	...	...	...	...	...	...	...	...	...
2128	ZOWMK	Urban	Public	ZBH	Standard	Female	Does not qualify	39.0	55.0
2129	ZOWMK	Urban	Public	ZBH	Standard	Female	Qualifies for reduced/free lunch	38.0	46.0
2130	ZOWMK	Urban	Public	ZBH	Standard	Female	Qualifies for reduced/free lunch	45.0	51.0
2131	ZOWMK	Urban	Public	ZBH	Standard	Male	Qualifies for reduced/free lunch	46.0	53.0
2132	ZOWMK	Urban	Public	ZBH	Standard	Male	Qualifies for reduced/free lunch	41.0	48.0

903 rows × 9 columns

Task3: To find average pretest score for all schools and highest pretest score

In [6]:
```
avg=school.groupby(["school"]).mean()["pretest"]
print(avg)
print()
print("The school with the highest average pretest is = ",avg[avg==avg.max()])
```

```
school
ANKYI    61.341463
CCAAW    64.623853
CIMBB    65.067568
CUQAM    53.925234
DNQDD    54.327869
FBUMG    62.891304
GJJHK    53.194915
GOKXL    50.796875
GOOBU    38.248408
IDGFP    75.202128
KFZMY    41.865385
KZKKE    37.261261
LAYPA    62.035088
OJOBU    56.197531
QOQTS    52.527027
UAGPU    62.390805
UKPGS    78.453125
UUUQX    67.253012
VHDHF    52.666667
VKWQH    52.060000
VVTVA    35.955752
ZMNYA    68.130435
ZOWMK    41.572650
Name: pretest, dtype: float64

The school with the highest average pretest is =  school
UKPGS    78.453125
Name: pretest, dtype: float64
```

Task4: To display the location wise and gender wise maximum

In [7]:
```
school.loc[:,["school_setting","gender"]].value_counts()[:2:]
```

Out[7]:
```
school_setting  gender
Urban           Female    456
                Male      444
dtype: int64
```

Task5: To display the minimum and maximum post test scores for all the schools and also show the top five post test scores Top 5

In [8]:
```
posttest_sc=pd.pivot_table(school,index="school",values=["posttest"],aggfunc=["min","max"])
print("The minimum and maximum posttest scores by all schools")
posttest_sc
```

```
The minimum and maximum posttest scores by all schools
```

Out[8]:

school	min posttest	max posttest
ANKYI	63.0	79.0
CCAAW	67.0	91.0
CIMBB	64.0	88.0
CUQAM	56.0	76.0
DNQDD	49.0	79.0
FBUMG	68.0	88.0
GJJHK	49.0	85.0
GOKXL	48.0	77.0
GOOBU	32.0	64.0
IDGFP	74.0	100.0
KFZMY	44.0	67.0
KZKKE	36.0	62.0
LAYPA	63.0	84.0
OJOBU	50.0	84.0
QOQTS	51.0	85.0
UAGPU	62.0	83.0
UKPGS	82.0	99.0
UUUQX	62.0	91.0
VHDHF	52.0	77.0
VKWQH	48.0	82.0
VVTVA	39.0	62.0
ZMNYA	66.0	95.0
ZOWMK	43.0	63.0

In [9]:
```
print("The top 5 posttest scores ")
m=pd.pivot_table(school,index="school",values=["posttest"],aggfunc=["max"]).sort_values(by=[('max', 'posttest')],ascending=False).iloc[0:5,:]
m
```

```
The top 5 posttest scores
```

Out[9]:

school	max posttest
IDGFP	100.0
UKPGS	99.0
ZMNYA	95.0
UUUQX	91.0
CCAAW	91.0

Task6: To display school wise averages and sum of pre test and top 5 pre test scores

In [10]:
```
print("The school wise average of pretest and postest")
print(school.groupby("school").mean())
```

```
The school wise average of pretest and postest
          pretest   posttest
school
ANKYI   61.341463  71.390244
CCAAW   64.623853  78.110092
CIMBB   65.067568  76.945946
CUQAM   53.925234  65.560748
DNQDD   54.327869  66.565574
FBUMG   62.891304  78.608696
GJJHK   53.194915  65.025424
GOKXL   50.796875  64.953125
GOOBU   38.248408  49.643312
IDGFP   75.202128  87.223404
KFZMY   41.865385  54.576923
KZKKE   37.261261  47.918919
LAYPA   62.035088  73.571429
OJOBU   56.197531  67.814815
QOQTS   52.527027  64.671141
UAGPU   62.390805  71.873563
UKPGS   78.453125  91.188976
UUUQX   67.253012  79.261905
VHDHF   52.666667  66.843137
VKWQH   52.060000  64.820000
VVTVA   35.955752  49.203540
ZMNYA   68.130435  81.608696
ZOWMK   41.572650  52.905983
```

In [11]:
```
print("The sum of pretest scores : ")
print(school.groupby("school").sum()["pretest"])
print()
print("The top 5 average pretest scores = ")
print(school.groupby("school")["pretest"].mean().sort_values(ascending=False).head())
```

```
The sum of pretest scores :
school
ANKYI     2515.0
CCAAW     7044.0
CIMBB     4815.0
CUQAM     5770.0
DNQDD     6628.0
FBUMG     2893.0
GJJHK     6277.0
GOKXL     3251.0
GOOBU     6005.0
IDGFP     7069.0
KFZMY     2177.0
KZKKE     4136.0
LAYPA     3536.0
OJOBU     4552.0
QOQTS     7774.0
UAGPU     5428.0
UKPGS    10042.0
UUUQX     5582.0
VHDHF     2686.0
VKWQH     5206.0
VVTVA     4063.0
ZMNYA     4701.0
ZOWMK     4864.0
Name: pretest, dtype: float64

The top 5 average pretest scores =
school
UKPGS    78.453125
IDGFP    75.202128
ZMNYA    68.130435
UUUQX    67.253012
CIMBB    65.067568
Name: pretest, dtype: float64
```

Task7: The school FBUMG with type of 5LQ classroom wishes to offer scholarships

In [12]:
```
FBU=test[(test.school=="FBUMG") & (test.classroom=="5LQ")].loc[:,["student_id","posttest"]]
FBU["Scholarship"]=(FBU.posttest/100)*10000
FBU
```

Out[12]:

	student_id	posttest	Scholarship
467	04DG5	84.0	8400.0
468	20M2D	85.0	8500.0
469	39KCW	81.0	8100.0
470	5Z1B6	88.0	8800.0
471	6TLU8	81.0	8100.0
472	ATQQJ	82.0	8200.0
473	AYEU1	79.0	7900.0
474	B9FSU	83.0	8300.0
475	EV13K	83.0	8300.0
476	I5H37	83.0	8300.0
477	JC5I9	80.0	8000.0
478	JPE2J	81.0	8100.0
479	MFBYU	82.0	8200.0
480	O144X	86.0	8600.0
481	OGKP3	83.0	8300.0
482	UMFI7	73.0	7300.0
483	V1DNJ	76.0	7600.0
484	YRN9S	87.0	8700.0

Task8: To replace does not qualify and qualifies with reduced free lunch with NO and YES

In [13]:
```
school.lunch.replace(to_replace=["Does not qualify","Qualifies for reduced/free lunch"],value=["No","Yes"],inplace=True)
school.lunch
```

Out[13]:
```
0        No
1        No
2        No
3        No
4        No
       ...
2128     No
2129    Yes
2130    Yes
2131    Yes
2132    Yes
Name: lunch, Length: 2133, dtype: object
```

Task9: To identify count of teaching methods by locations

In [14]:
```
pd.crosstab(school.school_setting,school.teaching_method)
```

Out[14]:

teaching_method	Experimental	Standard
school_setting		
Rural	201	309
Suburban	284	433
Urban	275	631

In [15]:
```
school.value_counts().groupby(["school_setting","teaching_method"]).sum()
```

Out[15]:
```
school_setting  teaching_method
Rural           Experimental       199
                Standard           307
Suburban        Experimental       283
                Standard           431
Urban           Experimental       269
                Standard           626
dtype: int64
```

Task10: To replace the pre test scores of male students with new value of 135 for male students who have post test less than 39

In [16]:
```
index=school[(school.gender=="Male")&(school.posttest<39)]["pretest"].index
school.pretest[index.values]=135
school.pretest[index.values]
school[(school.gender=="Male")&(school.posttest<39)]
```

Out[16]:

	school	school_setting	school_type	classroom	teaching_method	gender	lunch	pretest	posttest
742	GOOBU	Urban	Public	HKF	Standard	Male	Yes	135.0	38.0
745	GOOBU	Urban	Public	HKF	Standard	Male	Yes	135.0	38.0
751	GOOBU	Urban	Public	HKF	Standard	Male	Yes	135.0	34.0
1023	KZKKE	Rural	Public	3D0	Standard	Male	Yes	135.0	37.0
1081	KZKKE	Rural	Public	QTU	Standard	Male	Yes	135.0	38.0
1089	KZKKE	Rural	Public	QTU	Standard	Male	Yes	135.0	37.0

Task11: To compare average posttest scores with sub urban and urban male students

In [17]:
```
comp=pd.pivot_table(school,index="school_setting",values=["posttest"],aggfunc=["mean"]).iloc[0:2,:]
print(comp)
print()
print("relative percentage = ",round(((comp.values[1][0]-comp.values[0][0])/comp.values[0][0])*100,2),"%")
```

```
                   mean
               posttest
school_setting
Rural         64.039293
Suburban      76.020950

relative percentage =  18.71 %
```

Task12: To help replace missing values with most recurring values

In [18]:
```
print(school.isnull().sum())

school.gender=school.gender.fillna("Male")
school.pretest=school.pretest.fillna(school.pretest.mean())
school.posttest=school.posttest.fillna(school.posttest.mean())
print("After replacing ")
print()
print(school.isnull().sum())
```

```
school              0
school_setting      0
school_type         0
classroom           0
teaching_method     0
gender             10
lunch               0
pretest             4
posttest            4
dtype: int64
After replacing

school             0
school_setting     0
school_type        0
classroom          0
teaching_method    0
gender             0
lunch              0
pretest            0
posttest           0
dtype: int64
```