In [ ]:
```
# Add the Pandas dependency

import pandas as pd
```

In [ ]:
```
# Files to load

school_data_to_load = "Resources 4/schools_complete.csv"
student_data_to_load = "Resources 4/students_complete.csv"
```

In [2]:
```
# Read the school data file and store it in a Pandas DataFrame.

school_data_df = pd.read_csv(school_data_to_load)
school_data_df
```

```
---------------------------------------------------------------------------
NameError                                 Traceback (most recent call last)
Input In [2], in <cell line: 3>()
      1 # Read the school data file and store it in a Pandas DataFrame.
----> 3 school_data_df = read_csv(school_data_to_load)
      4 school_data_df

NameError: name 'read_csv' is not defined
```

In [ ]:
```
# Determine if there are any missing values in the student data.
student_data_df.count()

# Determine if there are any missing values in the school data.
school_data_df.isnull()

# Determine if there are any missing values in the student data.
student_data_df.isnull()

# Determine if there are not any missing values in the school data.
school_data_df.notnull()
```

In [ ]:
```
# Determine if there are not any missing values in the school data.
school_data_df.notnull()
```

In [ ]:
```
# Files to load
file_to_load = "Resources/missing_grades.csv"

# Read the CSV into a DataFrame
missing_grade_df = pd.read_csv(file_to_load)
missing_grade_df

# Drop the NaNs.
missing_grade_df.dropna()

# Fill in the empty rows with "85".
missing_grade_df.fillna(85)
```

In [ ]:
```
# Determine data types for the school DataFrame.
school_data_df.dtypes

# Determine data types for the student DataFrame.
student_data_df.dtypes
```

```
# Put the student names in a list.
student_names = student_data_df["student_name"].tolist()
student_names
```

In [ ]:
```
# Split the student name and determine the length of the split name.
for name in student_names:
    print(name.split(), len(name.split()))
```

In [ ]:
```
# Create a new list and use it for the for loop to iterate through the list.
students_to_fix = []

# Use an if statement to check the length of the name.
# If the name is greater than or equal to "3", add the name to the list.

for name in student_names:
    if len(name.split()) >= 3:
        students_to_fix.append(name)

# Get the length of the students whose names are greater than or equal to "3".
len(students_to_fix)
```

In [ ]:
```
# Add the prefixes less than or equal to 4 to a new list.
prefixes = []
for name in students_to_fix:
    if len(name.split()[0]) <= 4:
        prefixes.append(name.split()[0])

print(prefixes)


# Add the suffixes less than or equal to 3 to a new list.
suffixes = []
for name in students_to_fix:
    if len(name.split()[-1]) <= 3:
        suffixes.append(name.split()[-1])

print(suffixes)
```

In [ ]:
```
# Get the unique items in the "prefixes" list.
set(prefixes)

# Get the unique items in the "suffixes" list.
set(suffixes)
```

In [ ]:
```
# Strip "Mrs." from the student names
for name in students_to_fix:
    print(name.strip("Mrs."))

# Replace "Dr." with an empty string.
name = "Dr. Linda Santiago"
name.replace("Dr.", "")
```

In [3]:
```
# Add each prefix and suffix to remove to a list.
prefixes_suffixes = ["Dr. ", "Mr. ","Ms. ", "Mrs. ", "Miss ", " MD", " DDS", " DVM", '
```

In [ ]:
```
# Iterate through the "prefixes_suffixes" list and replace them with an empty space, '
```

```
for word in prefixes_suffixes:
    student_data_df["student_name"] = student_data_df["student_name"].str.replace(word
```

In [ ]:
```
# Put the cleaned students' names in another list.
student_names = student_data_df["student_name"].tolist()
student_names

# Create a new list and use it for the for loop to iterate through the list.
students_fixed = []
```

In [ ]:
```
# Create a new list and use it for the for loop to iterate through the list.
students_fixed = []

# Use an if statement to check the length of the name.

# If the name is greater than or equal to 3, add the name to the list.

for name in student_names:
    if len(name.split()) >= 3:
        students_fixed.append(name)

# Get the length of the students' names that are greater than or equal to 3.
len(students_fixed)
```

In [ ]:
```
# Add each prefix and suffix to remove to a list.
prefixes_suffixes = ["Dr. ", "Mr. ","Ms. ", "Mrs. ", "Miss ", " MD", " DDS", " DVM", '

# Iterate through the words in the "prefixes_suffixes" list and replace them with an e
for word in prefixes_suffixes:
    student_data_df["student_name"] = student_data_df["student_name"].str.replace(word
```