

P4DS Summative Assignment 2

Data Analysis Project

Developing Education Equity: Analysing Positive Outlier Schools' Performance at Keystage 4 for Disadvantaged Pupils in the UK - 2022/23

Student ID: 201901718

Name: Saqib Safdar

Project Plan

1.1 Sources of the dataset

a) Department for Education (DfE)

The multiple datasets are sourced from the Department for Education's (DfE) website [1][2]. The academic year 2022-23 is the most recent data and published on 1st February 2024. Five datasets from the DfE website were used in this analysis. For each of the data sets a separate file containing the metadata is also provided. The data sets were merged based on the Unique Reference Number (URN) column for each school. Progress 8 scores are used to evaluate school performance; this is a measure of the value-added by each school based on the progress made across 8 qualifications of each pupil, using their key stage 2 results from year 6 as a baseline. The attainment 8 score (total points across 8 subjects) of each pupil is similar key stage 2 results, is compared to the national average attainment 8; the difference indicates a level of progress. A progress 8 score of 1, would indicate the student has done better by 1 grade than the national average etc. Subjects included in progress 8 include:

- English and Mathematics - both double weighted due to importance

- EBacc Subjects - three slots from subjects such as sciences, computer science, history, geography and languages
- Open Group - remaining three from other academic, arts or vocational subjects

The DfE has data of the progress 8 score and funding for disadvantaged and non-disadvantaged students, which makes it very convenient to analyse.

[1] Department for Education. (n.d.). Explore education statistics: Data tables. Retrieved November 1, 2024, from <https://explore-education-statistics.service.gov.uk/data-tables>

[2] Department for Education. (n.d.). Compare the performance of schools and colleges in England. Retrieved November 10, 2024, from <https://www.gov.uk/school-performance-tables>

b) Index of Multiple Deprivation (IMD)

In addition to the four data sets from the DfE, rather than use funding for schools or number of disadvantaged pupils, the deprivation index for each area in the UK was downloaded from the Ministry of Housing, Communities and Local Government (MHCLG) website [3] and merged with the school information data set using the school postcode. This allows for a more detailed analysis of the relationship between school performance and socioeconomic factors which may affect the performance of disadvantaged students.

[3] Ministry of Housing, Communities & Local Government. (2019). English Indices of Deprivation 2019: Postcode Lookup. Retrieved from <https://imd-by-postcode.opendatacommunities.org/imd/2019>

1.2 Accuracy and Reliability of Data

The data is sourced from the Department for Education's (DfE) website and the Ministry of Housing, Communities and Local Government website. The data is accurate and reliable as it is sourced from official government sources. For the DfE, provisional and final KS4 results are provided. The key differences are the final results are quality assured for:

- a) Completeness of data: results are verified
- b) Accuracy of data: results are corrected for any errors or omissions
- c) Usage: results are approved for use in official publications and are publicly available.

To categorise each school's socioeconomic status, the Index of Multiple Deprivation Decile (IMD) is used, which ranks each postcode in England between 1 and 10. The IMD is a composite measure of deprivation based on several other domains of deprivation including income, employment, education and health. The data is from an official government source and is therefore accurate and reliable.

1.3 Data quality, usability, and presentation

Considerations: 1. The IMD data is from 2019 and is the nearest year to the academic year 2022-23 of school performance data. When evaluating the relationship between school performance and socioeconomic factors, the socioeconomic factors may have changed in some cases since 2019. However, I will treat these as negligible changes as the three-year period between 2019 and 2022 is relatively short. 2. As the analysis is based on school performance on a national level, including thousands of schools, I will use ‘inner’ joins to merge the datasets to ensure the analysis is not affected by schools which are not recognised. I will also drop any rows with missing values in key columns used for analysis.

Project Aim and Objectives

2.1 Context and motivation

Context

I have been working in education for two decades now. More recently, I have worked in MATs that are high performing and data-driven. The efficiency of a school/MAT in using its funds, together with the impact of its pedagogical framework can be seen using progress 8 scores. It has been shown that by five years of age, only 57% of disadvantaged pupils achieve a good level of development compared to 74% from better off households[4]. The gap continues throughout education; in 2022 -2023, 29% of free-school mean (FSM) pupils went to university which 49.8% of non FSM pupils progressed to university. [5].

Motivation

Several motivations underpin this analysis:

1. In a recent letter from the secretary of state for education, five priorities were set out for higher education providers, to top of which is : “Play a stronger role in expanding access and improving outcomes for disadvantaged students. The gap in outcomes from higher education between disadvantaged students and others is unacceptably large and is widening, with participation from disadvantaged students in decline for the first time in two decades.” [6]
2. Enhancing Education Practice: Some secondary schools are able to close the gap and give students from disadvantaged backgrounds better opportunities to progress to university. This data science investigations aim to identify outlier schools who outperform what is expected from them.

3. Justifying School Funding: Given the various avenues of funding data available, e.g. pupil premium for disadvantaged pupils, school-led tutoring funding, and the results for FSM and non FSM students, progress 8 and Eng - Maths, the efficiency of schools in using their funds can be evaluated. I can also examine if there is a correlation between progress 8 of disadvantaged and the level of funding schools receive to support them.
4. Understand demographic factors: Analysis of school demographics, e.g. gender, school type, local authority, can help to understand their influence on school performance.
5. Socioeconomic factors: The relationship between school performance and socioeconomic factors such as deprivation can be explored by merging the school performance data with the deprivation index for each area in the UK. Other factors such as percentage of disadvantaged students, percentage of non-disadvantaged students, pupil premium funding, percentage of disadvantaged students achieving grades 9-5 in English and Maths, can also be explored.
6. Impact of MAT: Group level management, collaboration and performance, particularly on outlier schools, can be explored to determine if there is a correlation between school performance and the type of MAT they belong to.

[4] Institute for Fiscal Studies. (2024, May). *The past and future of UK health spending*. Retrieved from <https://www.ifs.org.uk/publications/health-spending-report>

[5] Busby, E. (2024, October 24). Gap between private and state school pupils going to top universities widens. *The Independent*. Retrieved from <https://www.independent.co.uk/news/uk/gap-england-department-for-education-government-data-b2634966.html>

[6] Phillipson, B. (2024, November 4). *Letter from the Secretary of State for Education*. Department for Education.

2.2 Specific Objective(s)

1. Evaluate National Disparities in Educational Performance Between Advantaged and Disadvantaged Pupils

Using comprehensive datasets from the Department for Education (DfE) and the Ministry of Housing, Communities, and Local Government (MHCLG), conduct a detailed national-level analysis of the performance gap in key metrics, including Progress 8, Attainment 8, and English and Mathematics scores. This objective will involve merging, cleaning and validating data, before statistical analysis is conducted to determine the level of gap between disadvantaged and advantaged pupils

2. Identify and analyse outlier schools nationally for progress 8 scores for disadvantaged pupils and investigate contributing factors.

This objective will conduct more in depth statistical analysis to identify positive outlier schools with progress-8 scores for disadvantaged pupils. Further analysis on quantitative and categorical factors will be conducted to determine the influence of socio-economic indicators, such as the Index of Multiple Deprivation and demographics of the school.

3. Identify and evaluate the top performing multi-academy trusts in supporting disadvantaged pupils.

This objective will conduct statistical analysis to identify top performing multi-academy trusts and their success in closing the disadvantage gap. Hypothesis testing and regression analysis will be conducted to determine the level of impact of potential factors.

System Design

Architecture

Key Components: Descriptions, Purpose and Challenges

The following data sets will be downloaded and used from the DfE website.

1. DfE data set 1: KS4 school performance 2022-23 - Purpose: This provides information on the academic performance of each school and provides categories relating to advantage and disadvantage pupils in progress 8, attainment-8 and in EBACC subjects English and Mathematics. The description of each field is given below.

- Key fields used for analysis:
 - URN (Unique Reference Number)
 - Average Attainment 8 score
 - Average Progress 8 score
 - Percentage of disadvantaged students
 - Percentage of non-disadvantaged students
 - Percentage of disadvantaged students achieving grades 9-5 in English and Maths
 - Percentage of non-disadvantaged students achieving grades 9-5 in English and Maths
 - Attainment 8 score for non-disadvantaged students
 - Progress 8 score for non-disadvantaged students
 - Attainment 8 score for disadvantaged students
 - Progress 8 score for disadvantaged students
 - Progress 8 score in Maths for disadvantaged students
 - Progress 8 score in English for disadvantaged students
 - Progress 8 score in Maths for non-disadvantaged students
 - Progress 8 score in English for non-disadvantaged students

2. Data set 2: School information - provides information on the demographics of each school.

Purpose: The purpose of this data set is to determine school demographics such as gender, Ofsted rating etc, and other such categorical columns which can be used to determine potential impact on students' progress.

Key fields used in analysis:

- URN - Unique Reference Number for the school
- Local Authority Name (LANAME) - Name of the local authority the school belongs to
- Local Authority Code (LA) - Numeric code identifying the local authority
- School Type - Type of school (e.g. Academy, Community School, etc.)
- Minor Group - More detailed classification of school type
- Gender - Whether the school is mixed, boys only or girls only
- Ofsted Rating - Latest Ofsted inspection rating for the school

3. Data set 3: School funding

Purpose: Provides information on the various types of funding for each school.

Key fields used in analysis:

- School UKPRN: Unique ID number for each school provider
- School URN: Another unique ID number for each school
- Time Period: The academic year the funding is for
- FSM Funding: Money given to schools for students eligible for free school meals
- FSM6 Funding: Money given for students who were eligible for free school meals in the previous 6 years
- Pupil Premium: Extra funding given to help disadvantaged students
- Pupil Premium Pupils: Number of students who qualify for pupil premium funding
- School-led Tutoring Funding: Money given to schools to provide extra tutoring
- Total Funding: The total amount of funding received by the school

4. DfE data set 4: Multi Academy Trust (MAT) performance

Purpose: provides information of performance for each Multi-Academy Trust (MAT)

Key fields used from MAT performance data:

- Trust Name: Name of the Multi-Academy Trust
- Trust UID: Unique identification number for the trust
- Trust ID: Alternative ID code for the trust
- Number of Institutions: Number of schools in the trust
- Total Pupils: Total number of pupils across all schools in the trust

- Average Attainment 8 Score: Average attainment score across 8 subjects for the trust
- Average Progress 8 Score: Average progress score showing value added by trust
- Time Period: Academic year the data is from

5. Data set 5: Academies membership

Purpose: provides information on which MAT each school belongs to allowing external data such as to be linked to schools through their postcode and then to URN.

Key fields used in analysis:

- URN - Unique Reference Number for the school
- Group UID - Unique identifier for school group/trust
- Group ID - Alternative identifier for school group/trust
- Establishment Name - Official name of the school
- Group Name - Name of the school group/trust
- Postcode - Postcode of the school

6. MHCLG Data - Index of Multiple Deprivation (IMD)

Purpose: In addition to the five data sets from the DfE, the deprivation index for each area in the UK will be downloaded from the Ministry of Housing, Communities and Local Government (MHCLG) website and merged with the school information data set using the school postcode. This allows for a more detailed analysis of the relationship between school performance and socioeconomic factors which may affect the performance of disadvantaged students, as compared to say relying solely on funding data or percentage of disadvantaged pupils.

Key columns used for analysis:

- Postcode
- Index of Multiple Deprivation Decile

7. Metadata

Purpose: To identify the appropriate columns for analysis from the DfE data sets, the metadata will be used. Each of the DfE data sets lists above will have a corresponding meta-data file.

8. Classes

Purpose: To optimise the processes above, functions will be organised in classes

Challenges: Key challenges will be selecting and identify the appropriate columns from the DfE data sets as the data set a very large number of fields. The meta data file will be needed to be used to identify the code and description for each field. The code used would then need to be re-written in most cases so it is clear to the non-technical reader what the field stands

for, while retaining a format suitable for a data column in python. Another challenge will be ensuring data types are in the correct format for quantitative analysis. Where needed, feature engineering would need to be employed for new fields which may be required such as pupil premium funding per pupil. Another challenge will be in connecting the index of multiple deprivation IMD with each school, as the MHCLG is independent to the DfE, and will not include the school URN which is what will be used to combined the DfE data.

Pipeline and Workflow

The pipeline starts by setting up necessary functions and classes for data loading, wrangling and cleaning.

- Determine necessary functions and classes needed for the project
- Data Collection: Collect 2022-23 school and MAT performance data from the Department for Education (DfE); this includes the five data sets listed above and their meta files.
- Data Collection: Collect data from the Ministry of Housing, Communities and Local Government (MHCLG) website; Index of Multiple Deprivation Decile (IMD)
- Meta Data: Read the metadata for each data set to understand the data and variables. Create a dictionary of code and description.
- Using the meta-data fields extract the key columns for analysis from the data files.
- Data Integration: Merge the data sets based on the Unique Reference Number (URN) column for each school.
- Data Cleaning: Clean the data to remove any missing values and inconsistencies. Convert data to appropriate data types.
- Nomenclature: Determine new naming convention using meta-data dictionary and assign this to the data files.
- Feature Engineering: Create algorithms to define new features e.g pupil premium funding per pupil, key stage4_maths_gaps, keystage4 English gap and progress 8 gap between advantaged and disadvantaged pupils.
- Data Integration: Socioeconomic Indexing - incorporate the Index of Multiple Deprivation Decile (IMD) for each postcode to the school information data set.
- Statistical Analysis and Modelling: Conduct statistical analysis to determine advantage - disadvantage gap, identify outlier schools and top 10 performing MATs. Evaluate the impact of socioeconomic and other factors on school performance
- Visualisation: Create visualisations to present the findings.
- Conclusion: Summarise the findings and relate them to the original objectives.

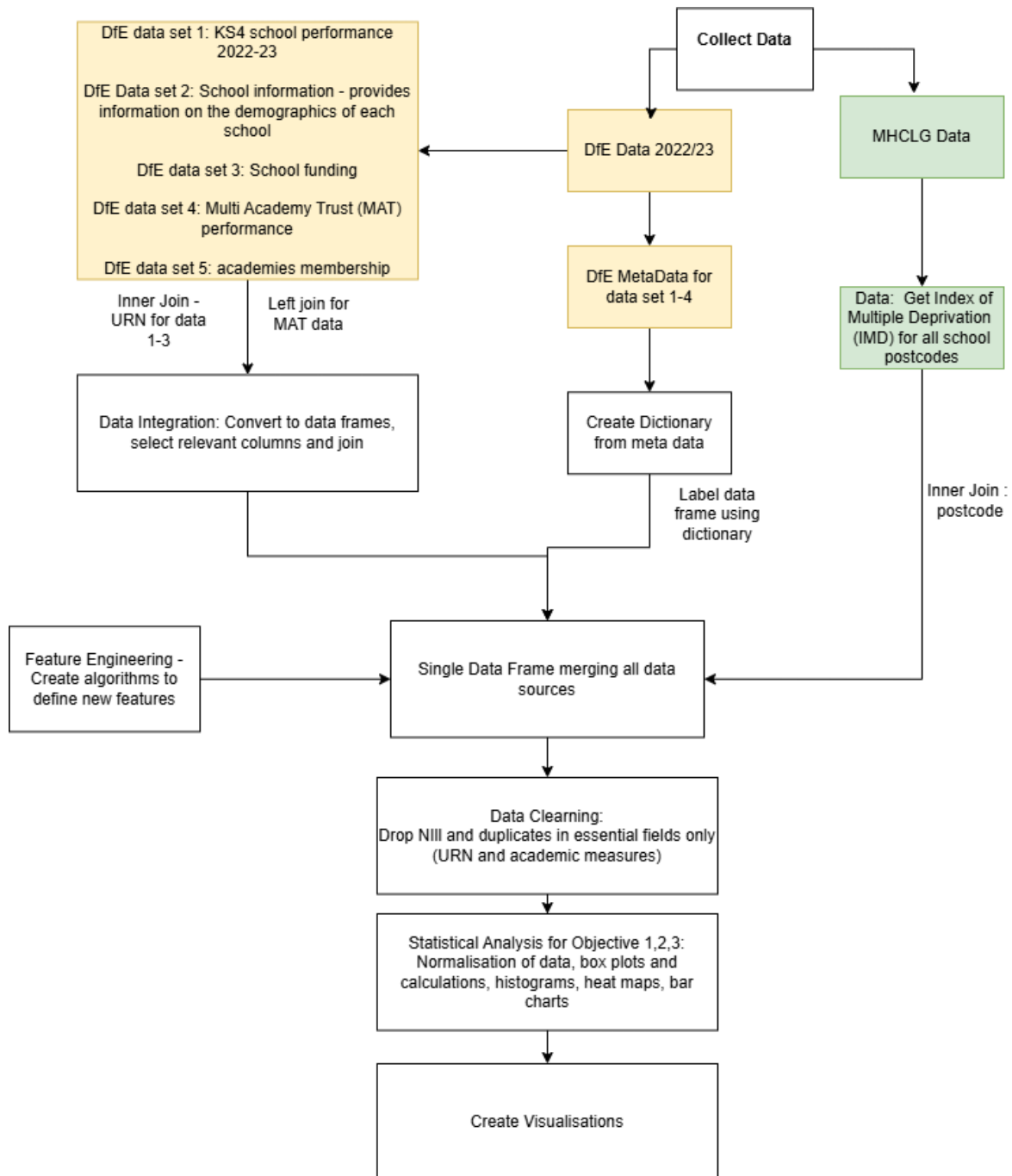


Figure 1: Pipeline.drawio.png

For a more dynamics view, workflow diagram can also be viewed [here](#)

Processing Modules and Algorithms

The following modules and algorithms will be required in a number of instances and therefore defined and written within a class:

- **Class: DataWrangler** - load data from CSV, Excel file or existing pandas data frame

Methods: - Load a csv file into a pandas dataframe using load_csv method - Load an excel file into a pandas dataframe using load_excel method - Create a dictionary from a dataframe using make_dictionary method - Rename columns in a dataframe using a dictionary using column_rename method - substitute original column names with descriptive names in a dictionary or list - Convert percentage strings in specified columns to float values using convert_percentage_columns method - Retrieve specific columns from a given dataframe using a set of URNs using get_school_details method

- Plot boxplots, histograms, heatmaps and scatter plots to visualise the data
- Write code to generate summary statistics of the boxplots

Program Code

Libraries

I will begin by importing the needed libraries for converting data to dataframes, conducting calculations and visualisations

```
import pandas as pd
import numpy as np
import statsmodels.api as sm
import matplotlib.pyplot as plt
import seaborn as sns
from scipy import stats
import os
from sklearn.preprocessing import StandardScaler
```

Classes

A class called Dataloader will be created to manage all core functions related to data loading and wrangling. This includes:

- load_csv
- load_excel

- make_dictionary
- column_rename
- convert_percentages_column

Details of the functions purpose, parameters and return value can be read in the docstrings below the function definition

```
class DataWrangler:
    def __init__(self, file_path=None, dataframe=None):
        """
        Initialise the DataWrangler with a file path or an existing DataFrame.

        Parameters:
        - file_path (str): The path to the data file (CSV or Excel).
        - dataframe (pd.DataFrame): An existing DataFrame to work with.
        """
        if dataframe is not None:
            self.df = dataframe.copy()
            print("DataWrangler initialised with the provided DataFrame.")
        elif file_path is not None:
            self.file_path = file_path
            self.df = None

            if self.file_path.endswith("csv"):
                self.load_csv()
            elif self.file_path.endswith(".xlsx"):
                self.load_excel()
            else:
                raise ValueError("Unsupported file format. Please provide a CSV or Excel file.")
        else:
            raise ValueError("Either file_path or dataframe must be provided.")

    def load_csv(self):
        """
        Load a CSV file into a pandas DataFrame.
        """
        try:
            self.df = pd.read_csv(self.file_path, encoding='latin1')
            print(f"CSV file loaded successfully from {self.file_path}")
        except FileNotFoundError as e:
            print(f"Error loading CSV file: {e}")
```

```

def load_excel(self):
    """
    Load an Excel file into a pandas DataFrame.
    """
    try:
        self.df = pd.read_excel(self.file_path)
        print(f"Excel file loaded successfully from {self.file_path}")
    except FileNotFoundError as e:
        print(f"Error loading Excel file: {e}")
        self.df = None

def make_dictionary(self, key_column: str, value_column: str):
    """
    Create a dictionary from two columns of the DataFrame.

    Parameters:
    - key_column (str): The column to use as the dictionary key.
    - value_column (str): The column to use as the dictionary value.

    Returns:
    - dict: A dictionary mapping keys to values.
    """
    try:
        return dict(zip(self.df[key_column], self.df[value_column]))
    except KeyError as e:
        print(f"Error: Key column not found in DataFrame: {e}")
        return None

def column_rename(self, column_dict: dict):
    """
    Rename columns in the DataFrame using a provided dictionary.

    Parameters:
    - column_dict (dict): A dictionary mapping original column names to new names.

    Returns:
    - pd.DataFrame: The DataFrame with renamed columns.
    """
    self.df = self.df.rename(columns=column_dict)
    print("Columns renamed successfully.")
    return self.df

```

```

def convert_percentage_columns(self, columns):
    """
    Remove % sign form columns .

    Parameters:
    - columns (list): List of column names to convert.

    Returns:
    - pd.DataFrame: The DataFrame with converted columns.
    """
    for col in columns:
        # Remove '%' and convert to float
        self.df[col] = self.df[col].astype(str).str.replace('%', '')
        print(f"Column '{col}' converted")
    return self.df

def get_school_details(self, urn_set, columns):
    """
    Retrieve essential school details for specified URNs and columns.

    Parameters:
    - urn_set (set): A set of URNs (Unique Reference Numbers) for schools.
    - columns (list): List of columns to include in the output.

    Returns:
    - pd.DataFrame: A DataFrame containing the specified details.
    """
    return self.df[self.df['URN'].isin(urn_set)][columns]

```

Load Data

I will now load and examine the five data files from the DfE as pandas data frames and do a quick inspection using `.head()`, `.info()`, `.describe()`. To avoid repetition, I will do a more thorough analyse of data types and missing values later, once all the data is combined.

```

# Beginning with MAT data:
ks4_mat_performance = DataWrangler('data/2022-2023_england_ks4-mats-performance.csv')
ks4_mat_performance.df.head()

```

CSV file loaded successfully from data/2022-2023_england_ks4-mats-performance.csv

	TIME_PERIOD	TIME_IDENTIFIER	TRUST_GROUP_TYPE	TRUST_NAME	TRUST_UID
0	202223	AcademicYear	Multi-academy trusts	ACTIVATE LEARNING EDUCATION	
1	202223	AcademicYear	Multi-academy trusts	ACER TRUST	
2	202223	AcademicYear	Multi-academy trusts	RED KITE LEARNING TRUST	
3	202223	AcademicYear	Multi-academy trusts	CONSILIUM ACADEMIES	
4	202223	AcademicYear	Multi-academy trusts	BATLEY MULTI ACADEMY TR	

```
# Keystage 4 school performance data:
ks4_school_performance = DataWrangler('data/2022-2023_england_ks4final.csv')
ks4_school_performance.df.head()
```

CSV file loaded successfully from data/2022-2023_england_ks4final.csv

C:\Users\saqib\AppData\Local\Temp\ipykernel_37860\3754428246.py:32: DtypeWarning: Columns (5) have mixed types. Specify dtype option on import or setting with 'astype' before pd.read_csv(self.file_path, encoding='latin1')

	RECTYPE	LEA	ESTAB	URN	SCHNAME	SCHNAME_AC	ADDRESS1	ADDRESS2	AD
0	1	201.0	6007.0	100003.0	City of London School		NaN		107 Que
1	1	201.0	6005.0	100001.0	City of London School for Girls		NaN		St Giles
2	1	201.0	6000.0	100544.0	David Game College		NaN		31 Jewr
3	4	201.0	NaN	NaN	NaN		NaN		NaN
4	1	202.0	4285.0	100053.0	Acland Burghley School		NaN		Burghle

```
#School demographics data:
school_demographics = DataWrangler('data/2022-2023_england_school_information.csv')
school_demographics.df.rename(columns={'URN': 'URN'}, inplace=True) #correction to URN col
school_demographics.df.head()
```

CSV file loaded successfully from data/2022-2023_england_school_information.csv

	URN	LANAME	LA	ESTAB	LAESTAB	SCHNAME	STREET	LOCALITY	ADDRESS3	TO
0	100000	City of London	201	3614	2013614	The Aldgate School			St James's Passa	
1	100001	City of London	201	6005	2016005	City of London School for Girls			St Giles' Terrace	

	URN	LANAME	LA	ESTAB	LAESTAB	SCHNAME	STREET	LOCALITY	ADDRESS3	TO
2	100002	City of London	201	6006	2016006	St Paul's Cathedral School			2 New Change	
3	100003	City of London	201	6007	2016007	City of London School			107 Queen Victo	
4	100008	Camden	202	2019	2022019	Argyle Primary School			Tonbridge Street	

```
# School funding data:
school_funding = DataWrangler('data/20230126_school_level_data_csv.csv')
school_funding.df.rename(columns={'i»;time_period': 'time_period'}, inplace=True) #correction
school_funding.df.head()
```

CSV file loaded successfully from data/20230126_school_level_data_csv.csv

	time_period	time_identifier	geographic_level	country_code	country_name	old_la_code	new_la
0	202223	Financial year	School	E92000001	England	301	E09000
1	202223	Financial year	School	E92000001	England	301	E09000
2	202223	Financial year	School	E92000001	England	301	E09000
3	202223	Financial year	School	E92000001	England	301	E09000
4	202223	Financial year	School	E92000001	England	301	E09000

```
#Academies data which connect URN code to postcode
academies_membership = DataWrangler('data/academiesmatmembership20220901.csv')
academies_membership.df.head()
```

CSV file loaded successfully from data/academiesmatmembership20220901.csv

	URN	DfE Number	EstablishmentNumber	Establishment UKPRN	LA (code)	LA (name)	Group U
0	136683.0	840/4054	4054.0	10033436.0	840.0	County Durham	
1	140594.0	936/2341	2341.0	10044809.0	936.0	Surrey	
2	136354.0	925/3510	3510.0	10032221.0	925.0	Lincolnshire	
3	137036.0	381/5404	5404.0	10034739.0	381.0	Calderdale	
4	140214.0	925/2016	2016.0	10043499.0	925.0	Lincolnshire	

Load Metadata and Make Dictionaries

I will now load the meta-data for each data file. To determine what each column in the data files means, I will create a dictionary using the make_dictionary function defined as part of

the DataWrangler class. The meta data is labeled after each associated data file with the addition of 'meta' at the end.

```
ks4_mat_performance_meta = DataWrangler('data/ks4-mats-performance_meta.csv')
ks4_mat_performance_dict = DataWrangler.make_dictionary(ks4_mat_performance_meta, 'Metafile 1')

ks4_mat_performance_dict
```

CSV file loaded successfully from data/ks4-mats-performance_meta.csv

```
{'TIME_PERIOD': nan,
 'TIME_IDENTIFIER': nan,
 'TRUST_GROUP_TYPE': 'Trust type',
 'TRUST_NAME': 'Trust name',
 'TRUST_UID': 'Trust Unique identifier',
 'TRUST_ID': 'Trust Identifier',
 'TRUST_COMPANIES_HOUSE_NUMBER': 'Trust companies house number',
 'TRUST_UKPRN': 'Trust UK provider reference number',
 'TRUST_LEADREGION': 'Trust lead region',
 'INSTITUTIONS_MATPTINC': 'URNs, included in performance measures',
 'NUMINST_MATPTINC': 'Number of academies in the trust, included in performance measures',
 'NUMINST_CONVERTER_MATPTINC': 'Number of converter academies, included in performance measures',
 'NUMINST_SPONSOR_MATPTINC': 'Number of sponsor-led academies, included in performance measures',
 'NUMINST_FREE_MATPTINC': 'Number of free school - mainstream academies, included in performance measures',
 'NUMINST_STUDIO_MATPTINC': 'Number of free school - studio schools, included in performance measures',
 'NUMINST.UTC_MATPTINC': 'Number of free school - UTCs, included in performance measures',
 'NUMINST_FSM6CLA1A_MATPTINC': 'Number of academies with disadvantaged pupils, included in performance measures',
 'NUMINST_3_MATPTINC': 'Number of academies that have been in the trust for 3 years, included in performance measures',
 'NUMINST_4_MATPTINC': 'Number of academies that have been in the trust for 4 years, included in performance measures',
 'NUMINST_5PLUS_MATPTINC': 'Number of academies that have been in the trust for 5 years or more, included in performance measures',
 'TPUP_MATPTINC': 'Number of pupils at the end of ks4, included in performance measures',
 'KS2ASS_MATPTINC': 'KS4 cohort average KS2 Scaled Score (average of English reading and maths)',
 'PFSM6CLA1A_MATPTINC': '% of pupils at the end of ks4 who are disadvantaged, included in performance measures',
 'PNOTFSM6CLA1A_MATPTINC': '% of pupils at the end of ks4 who are not disadvantaged, included in performance measures',
 'PEALGRP2_MATPTINC': '% of pupils at the end of ks4 with English as additional language (EAL) included in performance measures',
 'PSEN_ALL4_MATPTINC': '% of pupils at the end of ks4 with special educational needs (SEN) included in performance measures',
 'ATT8SCR_WGTAVG': 'Average Attainment 8 score per pupil at the end of KS4, weighted average',
 'P8MEACOV': '% of pupils at the end of ks4 included in Progress 8 measure',
 'P8MEA_WGTAVG': 'Progress 8 measure after adjustment for extreme scores, weighted average',
 'P8CILOW': 'Progress 8 lower 95% confidence interval for adjusted average',
 'P8CIUPP': 'Progress 8 upper 95% confidence interval for adjusted average',
```


'PTL2BASICS_95_WGTAVG': '% of pupils at the end of KS4 achieving strong 9-5 passes in both L
 'EBACCAPS_WGTAVG': 'Average EBacc APS score per pupil at the end of KS4, weighted average',
 'PTEBACC_95_WGTAVG': '% of pupils at the end of KS4 achieving the English Baccalaureate with
 'PTEBACC_94_WGTAVG': '% of pupils at the end of KS4 achieving the English Baccalaureate with
 'PTEBACC_E_PTQ_EE_WGTAVG': '% of pupils at the end of KS4 with entries in all English Bacc
 'ATT8SCR_WGTAVG_FSM6CLA1A': 'Average Attainment 8 score per disadvantaged pupil at the end o
 'P8MEACOV_FSM6CLA1A': '% of disadvantaged pupils at the end of ks4 included in Progress 8 me
 'P8MEA_WGTAVG_FSM6CLA1A': 'Progress 8 measure after adjustment for extreme scores for disadv
 'P8CILOW_FSM6CLA1A': 'Progress 8 lower 95% confidence interval for adjusted average for disa
 'P8CIUPP_FSM6CLA1A': 'Progress 8 upper 95% confidence interval for adjusted average for disa
 'PTL2BASICS_95_WGTAVG_FSM6CLA1A': '% of disadvantaged pupils at the end of KS4 achieving str
 'EBACCAPS_WGTAVG_FSM6CLA1A': 'Average EBacc APS score per disadvantaged pupil at the end of
 'PTEBACC_95_WGTAVG_FSM6CLA1A': '% of disadvantaged pupils at the end of KS4 achieving the E
 'PTEBACC_94_WGTAVG_FSM6CLA1A': '% of disadvantaged pupils at the end of KS4 achieving the E
 'PTEBACC_E_PTQ_EE_WGTAVG_FSM6CLA1A': '% of disadvantaged pupils at the end of KS4 with entr
 'ATT8SCR_WGTAVG_NFSM6CLA1A': 'Average Attainment 8 score per non-disadvantaged pupil at the
 'P8MEACOV_NFSM6CLA1A': '% of non-disadvantaged pupils at the end of ks4 included in Progress
 'P8MEA_WGTAVG_NFSM6CLA1A': 'Progress 8 measure after adjustment for extreme scores for non-c
 'P8CILOW_NFSM6CLA1A': 'Progress 8 lower 95% confidence interval for adjusted average for non
 'P8CIUPP_NFSM6CLA1A': 'Progress 8 upper 95% confidence interval for adjusted average for non
 'PTL2BASICS_95_WGTAVG_NFSM6CLA1A': '% of non-disadvantaged pupils at the end of KS4 achievin
 'EBACCAPS_WGTAVG_NFSM6CLA1A': 'Average EBacc APS score per non-disadvantaged pupil at the er
 'PTEBACC_95_WGTAVG_NFSM6CLA1A': '% of non-disadvantaged pupils at the end of KS4 achieving
 'PTEBACC_94_WGTAVG_NFSM6CLA1A': '% of non-disadvantaged pupils at the end of KS4 achieving
 'PTEBACC_E_PTQ_EE_WGTAVG_NFSM6CLA1A': '% of non-disadvantaged pupils at the end of KS4 with
 'P8_BANDING': 'Progress 8 banding shown on performance tables website',
 'INSTITUTIONS_INMAT': 'URNs, including mainstream academies not in performance measures',
 'NUMINST_INMAT': 'Number of academies in the trust, including those not in performance meas
 'NUMINST_CONVERTER_INMAT': 'Number of converter academies, including those not in performan
 'NUMINST_SPONSOR_INMAT': 'Number of sponsor-led academies, including those not in performan
 'NUMINST_FREE_INMAT': 'Number of free school - mainstream academies, including those not in
 'NUMINST_STUDIO_INMAT': 'Number of free school - studio schools, including those not in per
 'NUMINST.UTC_INMAT': 'Number of free school - UTCs, including those not in performance meas
 'TPUP_INMAT': 'Number of pupils at the end of KS4, including those not in performance measu
 'PFSM6CLA1A_INMAT': '% of pupils at the end of KS4 who are disadvantaged, including those n
 'PNOTFSM6CLA1A_INMAT': '% of pupils at the end of KS4 who are not disadvantaged, including t

```

school_demographics_meta = DataWrangler('data\school_information_meta.csv')
school_demographics_dict = DataWrangler.make_dictionary(school_demographics_meta,'Field Name')
school_demographics_dict

```

CSV file loaded successfully from data\school_information_meta.csv

```
<>:1: SyntaxWarning: invalid escape sequence '\s'
<>:1: SyntaxWarning: invalid escape sequence '\s'
C:\Users\saqib\AppData\Local\Temp\ipykernel_37860\3609157132.py:1: SyntaxWarning: invalid escape sequence '\s'
school_demographics_meta = DataWrangler('data\school_information_meta.csv')
```

```
{'URN': 'School unique reference number',
 'LANAME': 'Local authority name',
 'LA': 'Local authority number',
 'ESTAB': 'Establishment number',
 'LAESTAB': 'DfE number',
 'SCHNAME': 'School name',
 'STREET': 'School address (1)',
 'LOCALITY': 'School address (2)',
 'ADDRESS3': 'School address (3)',
 'TOWN': 'School town',
 'POSTCODE': 'School postcode',
 'SCHSTATUS': 'School open / closed status',
 'OPENDATE': 'Open date of school (if opened on or after 1st September 2022)',
 'CLOSEDATE': 'Date the school closed',
 'MINORGROUP': 'Type of school / college eg maintained school',
 'SCHOOLTYPE': 'School Type eg Voluntary Aided school',
 'ISPRIMARY': 'Does the school provide primary education? ( 0 = No, 1 = Yes)',
 'ISSECONDARY': 'Does the school provide secondary education? ( 0 = No, 1 = Yes)',
 'ISPOST16': 'Does the school provide post 16 education? ( 0 = No, 1 = Yes)',
 'AGELOW': 'Lowest age of entry',
 'AGEHIGH': 'Highest age of entry',
 'GENDER': "Indicates whether it's a mixed or single sex school",
 'RELCHAR': 'Religious character',
 'ADMPOL': 'Admissions Policy',
 'OFSTEDRATING': 'Ofsted rating',
 'OFSTEDLASTINSP': 'Ofsted last inspection date'}
```

```
ks4_school_performance_meta = DataWrangler('data/ks4_meta.xlsx') # this is originally in .xls
school_performance_dict = DataWrangler.make_dictionary(ks4_school_performance_meta, 'Metafile')
#school_performance_dict['URN'] = 'URN' # keep the URN column as it is as this will be used
school_performance_dict
```

Excel file loaded successfully from data/ks4_meta.xlsx

```
{'RECTYPE': 'Record type (1=mainstream school; 2=special school; 4=local authority; 5=National)',
 'LEA': 'Local authority code (see separate list of local authorities and their codes)',
```

'ESTAB': 'Establishment number',
 'URN': 'School Unique Reference Number',
 'SCHNAME': 'School name',
 'SCHNAME_AC': 'School now known as (used if the school has converted to an academy on or after 1 April 2011)',
 'ADDRESS1': 'School address (1)',
 'ADDRESS2': 'School address (2)',
 'ADDRESS3': 'School address (3)',
 'TOWN': 'School town',
 'PCODE': 'School postcode',
 'TELNUM': 'School telephone number',
 'PCON_CODE': 'Parliamentary constituency code',
 'PCON_NAME': 'Parliamentary constituency name',
 'CONTFAG': 'Contingency flag - school results 'significantly affected'. This field is zero if not significantly affected',
 'ICLOSE': 'Closed school flag (0=open; 1=closed; 2=pending closure)',
 'NFTYPE': 'School type (see separate list of abbreviations used in the tables)',
 'RELDENOM': 'School religious character',
 'ADMPOL': 'School admissions policy (self-declared by schools on Edubase)',
 'ADMPOL_PT': 'School admissions policy - new definition from 2019',
 'EGENDER': 'School gender of entry',
 'FEEDER': 'Indicates whether school is a feeder school for sixth form centre/consortia (0=not a feeder school; 1=feeder school)',
 'TABKS2': 'Indicates whether school is published in the primary school (key stage 2) performance table',
 'TAB1618': 'Indicates whether school is published in the school and college (16-18) performance table',
 'AGERANGE': 'Age range',
 'TOTPUPS': 'Number of pupils on roll (all ages)',
 'NUMBOYS': 'Total boys on roll (including part-time pupils)',
 'NUMGIRLS': 'Total girls on roll (including part-time pupils)',
 'TPUP': 'Number of pupils at the end of key stage 4',
 'BPUP': 'Number of boys at the end of key stage 4',
 'PBPUP': '% of pupils at the end of key stage 4 who are boys',
 'GPUP': 'Number of girls at the end of key stage 4',
 'PGPUP': '% of pupils at the end key stage 4 who are girls',
 'KS2ASS': 'KS4 cohort average KS2 Scaled Score (average of English reading and maths)',
 'TPRIORLO': 'Number of pupils at the end of key stage 4 with low prior attainment at the end of key stage 3',
 'PTPRIORLO': '% of pupils at the end of key stage 4 with low prior attainment at the end of key stage 3',
 'TPRIORAV': 'Number of pupils at the end of key stage 4 with middle prior attainment at the end of key stage 3',
 'PTPRIORAV': '% of pupils at the end of key stage 4 with middle prior attainment at the end of key stage 3',
 'TPRIORHI': 'Number of pupils at the end of key stage 4 with high prior attainment at the end of key stage 3',
 'PTPRIORHI': '% of pupils at the end of key stage 4 with high prior attainment at the end of key stage 3',
 'TFSM6CLA1A': 'Number of disadvantaged pupils at the end of key stage 4',
 'PTFSM6CLA1A': '% of pupils at the end of key stage 4 who are disadvantaged',
 'TNOTFSM6CLA1A': 'Number of non-disadvantaged pupils at the end of key stage 4',
 'PTNOTFSM6CLA1A': '% of pupils at the end of key stage 4 who are not disadvantaged',
 'TEALGRP2': 'Number of pupils at the end of key stage 4 with English as additional language'

'PTEALGRP2': '% of pupils at the end of key stage 4 with English as additional language (EAL)',
 'TEALGRP1': 'Number of pupils at the end of key stage 4 with English as their first language',
 'PTEALGRP1': '% of pupils at the end of key stage 4 with English as their first language',
 'TEALGRP3': 'Number of pupils at the end of key stage 4 whose first language is unclassified',
 'PTEALGRP3': '% of pupils at the end of key stage 4 whose first language is unclassified',
 'TNMOB': 'Number of pupils at the end of key stage 4 who are non-mobile',
 'PTNMOB': '% of pupils at the end of key stage 4 who are non-mobile',
 'SENE4': 'Number of pupils at the end of key stage 4 with special educational needs (SEN) with a statement',
 'PSENE4': '% of pupils at the end of key stage 4 with special educational needs (SEN) with a statement',
 'SEN_ALL4': 'Number of pupils at the end of key stage 4 with special educational needs (SEN) in any category',
 'PSEN_ALL4': '% of pupils at the end of key stage 4 with special educational needs (SEN) in any category',
 'SENK4': 'Number of pupils at the end of key stage 4 with special educational needs (SEN) without a statement',
 'PSENK4': '% of pupils at the end of key stage 4 with special educational needs (SEN) without a statement',
 'TOTATT8': 'Total sum of Attainment 8 scores',
 'ATT8SCR': 'Average Attainment 8 score per pupil',
 'TOTATT8ENG': 'Total sum of Attainment 8 scores for English element',
 'ATT8SCRENG': 'Average Attainment 8 score per pupil for English element',
 'TOTATT8MAT': 'Total sum of Attainment 8 scores for mathematics element',
 'ATT8SCRMAT': 'Average Attainment 8 score per pupil for mathematics element',
 'TOTATT8EBAC': 'Total sum of Attainment 8 scores for EBacc element',
 'ATT8SCREBAC': 'Average Attainment 8 score per pupil for EBacc element',
 'TOTATT8OPEN': 'Total sum of Attainment 8 scores for open element',
 'ATT8SCROPEN': 'Average Attainment 8 score per pupil for open element',
 'TOTATT8OPENGC': 'Total sum of Attainment 8 scores for open element - GCSE only',
 'ATT8SCROPGC': 'Average Attainment 8 score per pupil for open element - GCSE only',
 'TOTATT8OPENNG': 'Total sum of Attainment 8 scores for open element - non-GCSE only',
 'ATT8SCROPENG': 'Average Attainment 8 score per pupil for open element - non-GCSE only',
 'AVGEBACFILL': 'Average number of EBacc slots filled in Attainment 8 per pupil',
 'AVGOPENFILL': 'Average number of Open slots filled in Attainment 8 per pupil',
 'P8PUP': 'Number of pupils included in Progress 8 measure',
 'TP8ADJ': 'Number of pupils who have had P8 score adjusted in average',
 'P8MEACOV': '% of pupils at the end of key stage 4 included in Progress 8 measure',
 'P8MEA': 'Progress 8 measure after adjustment for extreme scores',
 'P8CILOW': 'Progress 8 lower 95% confidence interval for adjusted average',
 'P8CIUPP': 'Progress 8 upper 95% confidence interval for adjusted average',
 'P8MEA_ORIG': 'Progress 8 measure based on unadjusted pupil scores',
 'P8CILOW_ORIG': 'Progress 8 lower 95% confidence interval for unadjusted average',
 'P8CIUPP_ORIG': 'Progress 8 upper 95% confidence interval for unadjusted average',
 'P8MEAENG': 'Progress 8 measure for English element',
 'P8MEAENG_CILOW': 'Lower 95% confidence interval for Progress 8 English element',
 'P8MEAENG_CIUPP': 'Upper 95% confidence interval for Progress 8 English element',
 'P8MEAMAT': 'Progress 8 measure for mathematics element',
 'P8MEAMAT_CILOW': 'Lower 95% confidence interval for Progress 8 maths element',

'P8MEAMAT_CIUUP': 'Upper 95% confidence interval for Progress 8 maths element',
 'P8MEAEBAC': 'Progress 8 measure for EBacc element',
 'P8MEAEBAC_CILOW': 'Lower 95% confidence interval for Progress 8 EBacc element',
 'P8MEAEBAC_CIUUP': 'Upper 95% confidence interval for Progress 8 EBacc element',
 'P8MEAOPEN': 'Progress 8 measure for open element',
 'P8MEAOPEN_CILOW': 'Lower 95% confidence interval for Progress 8 open element',
 'P8MEAOPEN_CIUUP': 'Upper 95% confidence interval for Progress 8 open element',
 'PTL2BASICS_94': '% of pupils achieving standard 9-4 passes in both English and mathematics',
 'PTL2BASICS_95': '% of pupils achieving strong 9-5 passes in both English and mathematics',
 'TOTEACCAPS': 'Total EBacc APS score per pupil',
 'EBACCAPS': 'Average EBacc APS score per pupil',
 'EBACCAPS_FSM6CLA1A': 'Average EBacc APS score per disadvantaged pupil',
 'EBACCAPS_NFSM6CLA1A': 'Average EBacc APS score per non-disadvantaged pupil',
 'EBACCAPS_LO': 'Average EBacc APS score per pupil with low prior attainment',
 'EBACCAPS_MID': 'Average EBacc APS score per pupil with middle prior attainment',
 'EBACCAPS_HI': 'Average EBacc APS score per pupil with high prior attainment',
 'EBACCAPS_EAL': 'Average EBacc APS score per pupil for whom English is an additional language',
 'EBACCAPS_GIRLS': 'Average EBacc APS score per girl',
 'EBACCAPS_BOYS': 'Average EBacc APS score per boy',
 'EBACCAPS_NMOB': 'Average EBacc APS score per non-mobile pupil',
 'EBACCAPS_21': 'Average EBacc APS score per pupil in 2021',
 'EBACCAPS_FSM6CLA1A_21': 'Average EBacc APS score per disadvantaged pupil in 2021',
 'EBACCAPS_NFSM6CLA1A_21': 'Average EBacc APS score per non-disadvantaged pupil in 2021',
 'EBACCAPS_22': 'Average EBacc APS score per pupil in 2022',
 'EBACCAPS_FSM6CLA1A_22': 'Average EBacc APS score per disadvantaged pupil in 2022',
 'EBACCAPS_NFSM6CLA1A_22': 'Average EBacc APS score per non-disadvantaged pupil in 2022',
 'TEBACC_E_PTQ_EE': 'Number of key stage 4 pupils with entries in all English Baccalaureate subjects',
 'PTEBACC_E_PTQ_EE': '% of key stage 4 pupils with entries in all English Baccalaureate subjects',
 'PTEBACC_94': '% of pupils achieving the English Baccalaureate with 9-4 passes',
 'PTEBACC_95': '% of pupils achieving the English Baccalaureate with 9-5 passes',
 'TEBACENG_E_PTQ_EE': 'Number of pupils entering the English Baccalaureate English subject area',
 'PTEBACENG_E_PTQ_EE': '% of pupils entering the English Baccalaureate English subject area',
 'TEBACMAT_E_PTQ_EE': 'Number of pupils entering the English Baccalaureate Maths subject area',
 'PTEBACMAT_E_PTQ_EE': '% of pupils entering the English Baccalaureate Maths subject area',
 'TEBAC2SCI_E_PTQ_EE': 'Number of pupils entering the English Baccalaureate Science subject area',
 'PTEBAC2SCI_E_PTQ_EE': '% of pupils entering the English Baccalaureate Science subject area',
 'TEBACHUM_E_PTQ_EE': 'Number of pupils entering the English Baccalaureate Humanities subject area',
 'PTEBACHUM_E_PTQ_EE': '% of pupils entering the English Baccalaureate Humanities subject area',
 'TEBACLAN_E_PTQ_EE': 'Number of pupils entering the English Baccalaureate Language subject area',
 'PTEBACLAN_E_PTQ_EE': '% of pupils entering the English Baccalaureate Language subject area',
 'PTEBACENG_94': '% of pupils achieving the EBacc English subject area with a standard 9-4 pass',
 'PTEBACENG_95': '% of pupils achieving the EBacc English subject area with a strong 9-5 pass',
 'PTEBACMAT_94': '% of pupils achieving the EBacc Maths subject area with a standard 9-4 pass',

'PTEBACMAT_95': ' % of pupils achieving the EBacc Maths subject area with a strong 9-5 pass',
 'PTEBAC2SCI_94': ' % of entered pupils achieving the EBacc Science subject area with a 9-4 pass',
 'PTEBAC2SCI_95': ' % of entered pupils achieving the EBacc Science subject area with a 9-5 pass',
 'PTEBACHUM_94': ' % of entered pupils achieving the EBacc Humanities subject area with a 9-4 pass',
 'PTEBACHUM_95': ' % of entered pupils achieving the EBacc Humanities subject area with a 9-5 pass',
 'PTEBACLAN_94': ' % of entered pupils achieving the EBacc Language subject area with a 9-4 pass',
 'PTEBACLAN_95': ' % of entered pupils achieving the EBacc Language subject area with a 9-5 pass',
 'SCIVAPUP_PTQ_EE': 'Number of pupils included in English Baccalaureate Science Value Added indicators of',
 'SCIVACOV_PTQ_EE': 'Coverage of the English Baccalaureate Science Value Added indicators of',
 'HUMVAPUP_PTQ_EE': 'Number of pupils included in English Baccalaureate Humanities Value Added indicators of',
 'HUMVACOV_PTQ_EE': 'Coverage of the English Baccalaureate Humanities Value Added indicators of',
 'LANVAPUP_PTQ_EE': 'Number of pupils included in English Baccalaureate Language Value Added indicators of',
 'LANVACOV_PTQ_EE': 'Coverage of the English Baccalaureate Language Value Added indicators of',
 'SCIVAMEA_PTQ_EE': 'English Baccalaureate Science Value Added measure',
 'SCIVALOW_PTQ_EE': 'English Baccalaureate Science Value Added lower 95% confidence limit',
 'SCIVAUPP_PTQ_EE': 'English Baccalaureate Science Value Added upper 95% confidence limit',
 'HUMVAMEA_PTQ_EE': 'EBacc Humanities VA measure',
 'HUMVALOW_PTQ_EE': 'English Baccalaureate Humanities Value Added lower 95% confidence limit',
 'HUMVAUPP_PTQ_EE': 'English Baccalaureate Humanities Value Added upper 95% confidence limit',
 'LANVAMEA_PTQ_EE': 'English Baccalaureate Languages Value Added measure',
 'LANVALOW_PTQ_EE': 'English Baccalaureate Languages Value Added lower 95% confidence limit',
 'LANVAUPP_PTQ_EE': 'English Baccalaureate Languages Value Added upper 95% confidence limit',
 'TEBACENG_94': 'Number of pupils achieving EBacc English subject area with a standard 9-4 pass',
 'TEBACENG_95': 'Number of pupils achieving EBacc English subject area with a strong 9-5 pass',
 'TEBACMAT_94': 'Number of pupils achieving EBacc Maths subject area with a standard 9-4 pass',
 'TEBACMAT_95': 'Number of pupils achieving EBacc Maths subject area with a strong 9-5 pass',
 'TEBAC2SCI_94': 'Number of pupils achieving EBacc Science subject area with a 9-4 pass',
 'TEBAC2SCI_95': 'Number of pupils achieving EBacc Science subject area with a 9-5 pass',
 'TEBACHUM_94': 'Number of pupils achieving EBacc Humanities subject area with a 9-4 pass',
 'TEBACHUM_95': 'Number of pupils achieving EBacc Humanities subject area with a 9-5 pass',
 'TEBACLAN_94': 'Number of pupils achieving EBacc Language subject area with a 9-4 pass',
 'TEBACLAN_95': 'Number of pupils achieving EBacc Language subject area with a 9-5 pass',
 'TEBACC91': 'Number of pupils achieving the English Baccalaureate at grades 9-1',
 'PTEBACC91': ' % of pupils achieving the English Baccalaureate at grades 9-1',
 'TEBACENG91': 'Number of pupils achieving EBacc English subject area at grade 9-1',
 'PTEBACENG91': ' % of pupils achieving the EBacc English subject area at grade 9-1',
 'TEBACMAT91': 'Number of pupils achieving EBacc Maths subject area at grade 9-1',
 'PTEBACMAT91': ' % of pupils achieving the EBacc Maths subject area at grade 9-1',
 'TEBAC2SCI91': 'Number of pupils achieving EBacc Science subject area with grades 9-1',
 'PTEBAC2SCI91': ' % entered pupils achieving the EBacc Science subject area with grades 9-1',
 'TEBACHUM91': 'Number of pupils achieving EBacc Humanities subject area with grades 9-1',
 'PTEBACHUM91': ' % entered pupils achieving the EBacc Humanities subject area with grades 9-1',
 'TEBACLAN91': 'Number of pupils achieving EBacc Language subject area with grades 9-1',

'PTEBACLAN91': ' % of entered pupils achieving the EBacc Language subject area with grades 9',
 'ATT8SCR_FSM6CLA1A': 'Average Attainment 8 score per disadvantaged pupil',
 'P8PUP_FSM6CLA1A': 'Number of disadvantaged pupils in Progress 8 measure',
 'TP8ADJ_FSM6CLA1A': 'Number of disadvantaged pupils in progress measure with adjusted scores',
 'P8MEA_FSM6CLA1A': 'Adjusted Progress 8 measure - disadvantaged pupils',
 'P8CILOW_FSM6CLA1A': 'Adjusted Progress 8 lower 95% confidence interval - disadvantaged pupils',
 'P8CIUPP_FSM6CLA1A': 'Adjusted Progress 8 upper 95% confidence interval - disadvantaged pupils',
 'P8MEA_FSM6CLA1A_ORIG': 'Unadjusted Progress 8 measure - disadvantaged pupils',
 'P8CILOW_FSM6CLA1A_ORIG': 'Unadjusted Progress 8 lower 95% confidence interval - disadvantaged pupils',
 'P8CIUPP_FSM6CLA1A_ORIG': 'Unadjusted Progress 8 upper 95% confidence interval - disadvantaged pupils',
 'ATT8SCR_NFSM6CLA1A': 'Average Attainment 8 score per non-disadvantaged pupil',
 'P8PUP_NFSM6CLA1A': 'Number of non-disadvantaged pupils in Progress 8 measure',
 'TP8ADJ_NFSM6CLA1A': 'Number of non-disadvantaged pupils in progress measure with adjusted scores',
 'P8MEA_NFSM6CLA1A': 'Adjusted Progress 8 measure - non-disadvantaged pupils',
 'P8CILOW_NFSM6CLA1A': 'Progress 8 lower 95% confidence interval - non-disadvantaged pupils',
 'P8CIUPP_NFSM6CLA1A': 'Progress 8 upper 95% confidence interval - non-disadvantaged pupils',
 'P8MEA_NFSM6CLA1A_ORIG': 'Unadjusted Progress 8 measure - non-disadvantaged pupils',
 'P8CILOW_NFSM6CLA1A_ORIG': 'Unadjusted Progress 8 lower 95% confidence interval - non-disadvantaged pupils',
 'P8CIUPP_NFSM6CLA1A_ORIG': 'Unadjusted Progress 8 upper 95% confidence interval - non-disadvantaged pupils',
 'ATT8SCRENG_FSM6CLA1A': 'Average Attainment 8 score per disadvantaged pupil for English element',
 'P8MEAENG_FSM6CLA1A': 'Progress 8 measure for English element - disadvantaged pupils',
 'P8MEAENG_CILOW_FSM6CLA1A': 'Lower 95% confidence interval for Progress 8 English element for disadvantaged pupils',
 'P8MEAENG_CIUPP_FSM6CLA1A': 'Upper 95% confidence interval for Progress 8 English element for disadvantaged pupils',
 'ATT8SCRMAT_FSM6CLA1A': 'Average Attainment 8 score per disadvantaged pupil for mathematics element',
 'P8MEAMAT_FSM6CLA1A': 'Progress 8 measure for maths element - disadvantaged pupils',
 'P8MEAMAT_CILOW_FSM6CLA1A': 'Lower 95% confidence interval for Progress 8 maths element for disadvantaged pupils',
 'P8MEAMAT_CIUPP_FSM6CLA1A': 'Upper 95% confidence interval for Progress 8 maths element for disadvantaged pupils',
 'ATT8SCREBAC_FSM6CLA1A': 'Average Attainment 8 score per disadvantaged pupil for EBacc element',
 'P8MEAEBAC_FSM6CLA1A': 'Progress 8 measure for EBacc element - disadvantaged pupils',
 'P8MEAEBAC_CILOW_FSM6CLA1A': 'Lower 95% confidence interval for Progress 8 EBacc element for disadvantaged pupils',
 'P8MEAEBAC_CIUPP_FSM6CLA1A': 'Upper 95% confidence interval for Progress 8 EBacc element for disadvantaged pupils',
 'ATT8SCROPEN_FSM6CLA1A': 'Average Attainment 8 score per disadvantaged pupil for open element',
 'P8MEAOPEN_FSM6CLA1A': 'Progress 8 measure for open element - disadvantaged pupils',
 'P8MEAOPEN_CILOW_FSM6CLA1A': 'Lower 95% confidence interval for Progress 8 open element for disadvantaged pupils',
 'P8MEAOPEN_CIUPP_FSM6CLA1A': 'Upper 95% confidence interval for Progress 8 open element for disadvantaged pupils',
 'ATT8SCRENG_NFSM6CLA1A': 'Average Attainment 8 score per non-disadvantaged pupil for English element',
 'P8MEAENG_NFSM6CLA1A': 'Progress 8 measure for English element - non-disadvantaged pupils',
 'P8MEAENG_CILOW_NFSM6CLA1A': 'Lower 95% confidence interval for Progress 8 English element for non-disadvantaged pupils',
 'P8MEAENG_CIUPP_NFSM6CLA1A': 'Upper 95% confidence interval for Progress 8 English element for non-disadvantaged pupils',
 'ATT8SCRMAT_NFSM6CLA1A': 'Average Attainment 8 score per non-disadvantaged pupil for mathematics element',
 'P8MEAMAT_NFSM6CLA1A': 'Progress 8 measure for maths element - non-disadvantaged pupils',
 'P8MEAMAT_CILOW_NFSM6CLA1A': 'Lower 95% confidence interval for Progress 8 maths element for non-disadvantaged pupils',
 'P8MEAMAT_CIUPP_NFSM6CLA1A': 'Upper 95% confidence interval for Progress 8 maths element for non-disadvantaged pupils'

'ATT8SCREBAC_NFSM6CLA1A': 'Average Attainment 8 score per non-disadvantaged pupil for EBacc',
 'P8MEAEBAC_NFSM6CLA1A': 'Progress 8 measure for EBacc element - non-disadvantaged pupils',
 'P8MEAEBAC_CILOW_NFSM6CLA1A': 'Lower 95% confidence interval for Progress 8 EBacc element for',
 'P8MEAEBAC_CIUUP_NFSM6CLA1A': 'Upper 95% confidence interval for Progress 8 EBacc element for',
 'ATT8SCROPEN_NFSM6CLA1A': 'Average Attainment 8 score per non-disadvantaged pupil for open element',
 'P8MEAOPEN_NFSM6CLA1A': 'Progress 8 measure for open element - non-disadvantaged pupils',
 'P8MEAOPEN_CILOW_NFSM6CLA1A': 'Lower 95% confidence interval for Progress 8 open element for',
 'P8MEAOPEN_CIUUP_NFSM6CLA1A': 'Upper 95% confidence interval for Progress 8 open element for',
 'ATT8SCROPENNG_FSM6CLA1A': 'Average Attainment 8 score per disadvantaged pupil for open element',
 'ATT8SCROPENNG_FSM6CLA1A': 'Average Attainment 8 score per disadvantaged pupil for open element',
 'ATT8SCROPENNG_NFSM6CLA1A': 'Average Attainment 8 score per non-disadvantaged pupil for open element',
 'ATT8SCROPENNG_NFSM6CLA1A': 'Average Attainment 8 score per non-disadvantaged pupil for open element',
 'DIFFN_ATT8': 'Difference between Attainment 8 for disadvantaged pupils in school/LA and non-disadvantaged pupils',
 'DIFFN_P8MEA': 'Difference between Progress 8 measure for disadvantaged pupils in school/LA and non-disadvantaged pupils',
 'ATT8SCR_LO': 'Average Attainment 8 score per pupil with low prior attainment',
 'P8PUP_LO': 'Number of pupils with low prior attainment included in Progress 8 measure',
 'TP8ADJ_LO': 'Number of pupils with low prior attainments in progress measure with adjusted measures',
 'P8MEA_LO': 'Adjusted Progress 8 measure - pupils with low prior attainments',
 'P8CILOW_LO': 'Adjusted Progress 8 lower 95% confidence interval - pupils with low prior attainment',
 'P8CIUUP_LO': 'Adjusted Progress 8 upper 95% confidence interval - pupils with low prior attainment',
 'P8MEA_LO_ORIG': 'Unadjusted Progress 8 measure - pupils with low prior attainments',
 'P8CILOW_LO_ORIG': 'Unadjusted Progress 8 lower 95% confidence interval - pupils with low prior attainment',
 'P8CIUUP_LO_ORIG': 'Unadjusted Progress 8 upper 95% confidence interval - pupils with low prior attainment',
 'ATT8SCR_MID': 'Average Attainment 8 score per pupil with middle prior attainment',
 'P8PUP_MID': 'Number of pupils with middle prior attainment included in Progress 8 measure',
 'TP8ADJ_MID': 'Number of pupils with middle prior attainments in progress measure with adjusted measures',
 'P8MEA_MID': 'Adjusted Progress 8 measure - pupils with middle prior attainment',
 'P8CILOW_MID': 'Progress 8 lower 95% confidence interval - pupils with middle prior attainment',
 'P8CIUUP_MID': 'Progress 8 upper 95% confidence interval - pupils with middle prior attainment',
 'P8MEA_MID_ORIG': 'Unadjusted Progress 8 measure - pupils with middle prior attainments',
 'P8CILOW_MID_ORIG': 'Unadjusted Progress 8 lower 95% confidence interval - pupils with middle prior attainment',
 'P8CIUUP_MID_ORIG': 'Unadjusted Progress 8 upper 95% confidence interval - pupils with middle prior attainment',
 'ATT8SCR_HI': 'Average Attainment 8 score per pupil with high prior attainment',
 'P8PUP_HI': 'Number of pupils with high prior attainment included in Progress 8 measure',
 'TP8ADJ_HI': 'Number of pupils with high prior attainments in progress measure with adjusted measures',
 'P8MEA_HI': 'Adjusted Progress 8 measure - pupils with high prior attainment',
 'P8CILOW_HI': 'Progress 8 lower 95% confidence interval - pupils with high prior attainment',
 'P8CIUUP_HI': 'Progress 8 upper 95% confidence interval - pupils with high prior attainment',
 'P8MEA_HI_ORIG': 'Unadjusted Progress 8 measure - pupils with high prior attainments',
 'P8CILOW_HI_ORIG': 'Unadjusted Progress 8 lower 95% confidence interval - pupils with high prior attainment',
 'P8CIUUP_HI_ORIG': 'Unadjusted Progress 8 upper 95% confidence interval - pupils with high prior attainment',
 'ATT8SCR_EAL': 'Average Attainment 8 score per pupil for whom English is an additional language',
 'ATT8SCRENG_EAL': 'Average Attainment 8 score per pupil for whom English is an additional language'

'ATT8SCRMAT_EAL': 'Average Attainment 8 score per pupil for whom English is an additional language',
 'ATT8SCREBAC_EAL': 'Average Attainment 8 score per pupil for whom English is an additional language',
 'ATT8SCROPEN_EAL': 'Average Attainment 8 score per pupil for whom English is an additional language',
 'ATT8SCROPENG_EAL': 'Average Attainment 8 score per pupil for whom English is an additional language',
 'ATT8SCROPENNG_EAL': 'Average Attainment 8 score per pupil for whom English is an additional language',
 'P8PUP_EAL': 'Number of pupils for whom English is an additional language included in Progress 8 measure',
 'TP8ADJ_EAL': 'Number of pupils for whom English is an additional language in progress measure with adjusted scores',
 'P8MEA_EAL': 'Adjusted Progress 8 measure - pupils for whom English is an additional language',
 'P8CILOW_EAL': 'Adjusted Progress 8 lower 95% confidence interval - pupils for whom English is an additional language',
 'P8CIUPP_EAL': 'Adjusted Progress 8 upper 95% confidence interval - pupils for whom English is an additional language',
 'P8MEA_EAL_ORIG': 'Unadjusted Progress 8 measure - pupils for whom English is an additional language',
 'P8CILOW_EAL_ORIG': 'Unadjusted Progress 8 lower 95% confidence interval - pupils for whom English is an additional language',
 'P8CIUPP_EAL_ORIG': 'Unadjusted Progress 8 upper 95% confidence interval - pupils for whom English is an additional language',
 'ATT8SCR_GIRLS': 'Average Attainment 8 score per girl',
 'ATT8SCRENG_GIRLS': 'Average Attainment 8 score per girl for English element',
 'ATT8SCRMAT_GIRLS': 'Average Attainment 8 score per girl for mathematics element',
 'ATT8SCREBAC_GIRLS': 'Average Attainment 8 score per girl for EBacc element',
 'ATT8SCROPEN_GIRLS': 'Average Attainment 8 score per girl for open element',
 'ATT8SCROPENG_GIRLS': 'Average Attainment 8 score per girl - GCSE only',
 'ATT8SCROPENNG_GIRLS': 'Average Attainment 8 score per girl - non-GCSE only',
 'P8PUP_GIRLS': 'Number of girls included in Progress 8 measure',
 'TP8ADJ_GIRLS': 'Number of girls in progress measure with adjusted scores',
 'P8MEA_GIRLS': 'Adjusted Progress 8 measure - girls',
 'P8CILOW_GIRLS': 'Adjusted Progress 8 lower 95% confidence interval - girls',
 'P8CIUPP_GIRLS': 'Adjusted Progress 8 upper 95% confidence interval - girls',
 'P8MEA_GIRLS_ORIG': 'Unadjusted Progress 8 measure - girls',
 'P8CILOW_GIRLS_ORIG': 'Unadjusted Progress 8 lower 95% confidence interval - girls',
 'P8CIUPP_GIRLS_ORIG': 'Unadjusted Progress 8 upper 95% confidence interval - girls',
 'ATT8SCR_BOYS': 'Average Attainment 8 score per boy',
 'ATT8SCRENG_BOYS': 'Average Attainment 8 score per boy for English element',
 'ATT8SCRMAT_BOYS': 'Average Attainment 8 score per boy for mathematics element',
 'ATT8SCREBAC_BOYS': 'Average Attainment 8 score per boy for EBacc element',
 'ATT8SCROPEN_BOYS': 'Average Attainment 8 score per boy for open element',
 'ATT8SCROPENG_BOYS': 'Average Attainment 8 score per boy - GCSE only',
 'ATT8SCROPENNG_BOYS': 'Average Attainment 8 score per boy - non-GCSE only',
 'P8PUP_BOYS': 'Number of boys included in Progress 8 measure',
 'TP8ADJ_BOYS': 'Number of boys in progress measure with adjusted scores',
 'P8MEA_BOYS': 'Adjusted Progress 8 measure - boys',
 'P8CILOW_BOYS': 'Adjusted Progress 8 lower 95% confidence interval - boys',
 'P8CIUPP_BOYS': 'Adjusted Progress 8 upper 95% confidence interval - boys',
 'P8MEA_BOYS_ORIG': 'Unadjusted Progress 8 measure - boys',
 'P8CILOW_BOYS_ORIG': 'Unadjusted Progress 8 lower 95% confidence interval - boys',
 'P8CIUPP_BOYS_ORIG': 'Unadjusted Progress 8 upper 95% confidence interval - boys',

'ATT8SCR_NMOB': 'Average Attainment 8 score per non-mobile pupil',
 'ATT8SCRENG_NMOB': 'Average Attainment 8 score per non-mobile pupil for English element',
 'ATT8SCRMAT_NMOB': 'Average Attainment 8 score per non-mobile pupil for mathematics element',
 'ATT8SCREBAC_NMOB': 'Average Attainment 8 score per non-mobile pupil for EBacc element',
 'ATT8SCROPEN_NMOB': 'Average Attainment 8 score per non-mobile pupil for open element',
 'ATT8SCROPENG_NMOB': 'Average Attainment 8 score per non-mobile pupil - GCSE only',
 'ATT8SCROPENNG_NMOB': 'Average Attainment 8 score per non-mobile pupil - non-GCSE only',
 'P8PUP_NMOB': 'Number of non-mobile pupils included in Progress 8 measure',
 'TP8ADJ_NMOB': 'Number of non-mobile pupils in progress measure with adjusted scores',
 'P8MEA_NMOB': 'Adjusted Progress 8 measure - non-mobile pupils',
 'P8CILOW_NMOB': 'Adjusted Progress 8 lower 95% confidence interval - non-mobile pupils',
 'P8CIUPP_NMOB': 'Adjusted Progress 8 upper 95% confidence interval - non-mobile pupils',
 'P8MEA_NMOB_ORIG': 'Unadjusted Progress 8 measure - non-mobile pupils',
 'P8CILOW_NMOB_ORIG': 'Unadjusted Progress 8 lower 95% confidence interval - non-mobile pupils',
 'P8CIUPP_NMOB_ORIG': 'Unadjusted Progress 8 upper 95% confidence interval - non-mobile pupils',
 'ATT8SCR_21': 'Average Attainment 8 score per pupil - 2021',
 'P8PUP_21': 'Number of pupils in progress measure - 2021',
 'P8MEA_21': 'Progress 8 measure - 2021',
 'P8CILOW_21': 'Progress 8 lower 95% confidence interval - 2021',
 'P8CIUPP_21': 'Progress 8 upper 95% confidence interval - 2021',
 'ATT8SCR_FSM6CLA1A_21': 'Average Attainment 8 score per disadvantaged pupil - 2021',
 'P8PUP_FSM6CLA1A_21': 'Number of disadvantaged pupils in progress measure - 2021',
 'P8MEA_FSM6CLA1A_21': 'Progress 8 measure - disadvantaged pupils - 2021',
 'P8CILOW_FSM6CLA1A_21': 'Progress 8 lower 95% confidence interval - disadvantaged pupils - 2021',
 'P8CIUPP_FSM6CLA1A_21': 'Progress 8 upper 95% confidence interval - disadvantaged pupils - 2021',
 'ATT8SCR_NFSM6CLA1A_21': 'Average Attainment 8 score per non-disadvantaged pupil - 2021',
 'P8PUP_NFSM6CLA1A_21': 'Number of non-disadvantaged pupils in progress measure - 2021',
 'P8MEA_NFSM6CLA1A_21': 'Progress 8 measure - non-disadvantaged pupils - 2021',
 'P8CILOW_NFSM6CLA1A_21': 'Progress 8 lower 95% confidence interval - non-disadvantaged pupils - 2021',
 'P8CIUPP_NFSM6CLA1A_21': 'Progress 8 upper 95% confidence interval - non-disadvantaged pupils - 2021',
 'ATT8SCR_22': 'Average Attainment 8 score per pupil - 2022',
 'P8PUP_22': 'Number of pupils in progress measure - 2022',
 'P8MEA_22': 'Progress 8 measure - 2022',
 'P8CILOW_22': 'Progress 8 lower 95% confidence interval - 2022',
 'P8CIUPP_22': 'Progress 8 upper 95% confidence interval - 2022',
 'ATT8SCR_FSM6CLA1A_22': 'Average Attainment 8 score per disadvantaged pupil - 2022',
 'P8PUP_FSM6CLA1A_22': 'Number of disadvantaged pupils in progress measure - 2022',
 'P8MEA_FSM6CLA1A_22': 'Progress 8 measure - disadvantaged pupils - 2022',
 'P8CILOW_FSM6CLA1A_22': 'Progress 8 lower 95% confidence interval - disadvantaged pupils - 2022',
 'P8CIUPP_FSM6CLA1A_22': 'Progress 8 upper 95% confidence interval - disadvantaged pupils - 2022',
 'ATT8SCR_NFSM6CLA1A_22': 'Average Attainment 8 score per non-disadvantaged pupil - 2022',
 'P8PUP_NFSM6CLA1A_22': 'Number of non-disadvantaged pupils in progress measure - 2022',
 'P8MEA_NFSM6CLA1A_22': 'Progress 8 measure - non-disadvantaged pupils - 2022',

'P8CILOW_NFSM6CLA1A_22': 'Progress 8 lower 95% confidence interval - non-disadvantaged pupils',
 'P8CIUPP_NFSM6CLA1A_22': 'Progress 8 upper 95% confidence interval - non-disadvantaged pupils',
 'TEBACC_ELO_PTQ_EE': 'Number of pupils in low prior attainment band with entries in all EBacc subjects',
 'PTEBACC_ELO_PTQ_EE': 'EBacc entered % by low prior attainment',
 'PTEBACCLO_94': 'EBacc achieved % by low prior attainment - with standard 9-4 passes in English, Maths and Science',
 'PTEBACCLO_95': 'EBacc achieved % by low prior attainment - with 9-5 passes',
 'TEBACC_EAV_PTQ_EE': 'Number of pupils in middle prior attainment band with entries in all EBacc subjects',
 'PTEBACC_EAV_PTQ_EE': 'EBacc entered % by middle prior attainment',
 'PTEBACCAV_94': 'EBacc achieved % by middle prior attainment - with 9-4 passes',
 'PTEBACCAV_95': 'EBacc achieved % by middle prior attainment - with 9-5 passes',
 'TEBACC_EHI_PTQ_EE': 'Number of pupils in high prior attainment band with entries in all EBacc subjects',
 'PTEBACC_EHI_PTQ_EE': 'EBacc entered % by high prior attainment',
 'PTEBACCCHI_94': 'EBacc achieved % by high prior attainment - with 9-4 passes',
 'PTEBACCCHI_95': 'EBacc achieved % by high prior attainment - with 9-5 passes',
 'PTEBACC_EFSM6CLA1A_PTQ_EE': '% of disadvantaged pupils entering all English Baccalaureate subjects',
 'PTEBACC_ENFSM6CLA1A_PTQ_EE': '% of non-disadvantaged pupils entering all English Baccalaureate subjects',
 'PTEBACC_94_FSM6CLA1A': '% of disadvantaged pupils achieving the English Baccalaureate - with standard 9-4 passes',
 'PTEBACC_95_FSM6CLA1A': '% of disadvantaged pupils achieving the English Baccalaureate - with 9-5 passes',
 'PTEBACC_94_NFSM6CLA1A': '% of non-disadvantaged pupils achieving the English Baccalaureate - with standard 9-4 passes',
 'PTEBACC_95_NFSM6CLA1A': '% of non-disadvantaged pupils achieving the English Baccalaureate - with 9-5 passes',
 'SCIVAMEA_LO_PTQ_EE': 'English Baccalaureate Science Value Added measure for pupils with low prior attainment',
 'SCIVAMEA_MID_PTQ_EE': 'English Baccalaureate Science Value Added measure for pupils with middle prior attainment',
 'SCIVAMEA_HI_PTQ_EE': 'English Baccalaureate Science Value Added measure for pupils with high prior attainment',
 'SCIVAMEA_FSM6CLA1A_PTQ_EE': 'English Baccalaureate Science Value Added measure for disadvantaged pupils',
 'SCIVAMEA_NFSM6CLA1A_PTQ_EE': 'English Baccalaureate Science Value Added measure for non-disadvantaged pupils',
 'HUMVAMEA_LO_PTQ_EE': 'English Baccalaureate Humanities Value Added measure for pupils with low prior attainment',
 'HUMVAMEA_MID_PTQ_EE': 'English Baccalaureate Humanities Value Added measure for pupils with middle prior attainment',
 'HUMVAMEA_HI_PTQ_EE': 'English Baccalaureate Humanities Value Added measure for pupils with high prior attainment',
 'HUMVAMEA_FSM6CLA1A_PTQ_EE': 'English Baccalaureate Humanities Value Added measure for disadvantaged pupils',
 'HUMVAMEA_NFSM6CLA1A_PTQ_EE': 'English Baccalaureate Humanities Value Added measure for non-disadvantaged pupils',
 'LANVAMEA_LO_PTQ_EE': 'English Baccalaureate Languages Value Added measure for pupils with low prior attainment',
 'LANVAMEA_MID_PTQ_EE': 'English Baccalaureate Languages Value Added measure for pupils with middle prior attainment',
 'LANVAMEA_HI_PTQ_EE': 'English Baccalaureate Languages Value Added measure for pupils with high prior attainment',
 'LANVAMEA_FSM6CLA1A_PTQ_EE': 'English Baccalaureate Languages Value Added measure for disadvantaged pupils',
 'LANVAMEA_NFSM6CLA1A_PTQ_EE': 'English Baccalaureate Languages Value Added measure for non-disadvantaged pupils',
 'SCIVAUPP_FSM6CLA1A_PTQ_EE': 'Upper 95% confidence limit for English Baccalaureate Science Value Added measure for disadvantaged pupils',
 'SCIVALOW_FSM6CLA1A_PTQ_EE': 'Lower 95% confidence limit for English Baccalaureate Science Value Added measure for disadvantaged pupils',
 'SCIVAUPP_NFSM6CLA1A_PTQ_EE': 'Upper 95% confidence limit for English Baccalaureate Science Value Added measure for non-disadvantaged pupils',
 'SCIVALOW_NFSM6CLA1A_PTQ_EE': 'Lower 95% confidence limit for English Baccalaureate Science Value Added measure for non-disadvantaged pupils',
 'SCIVAUPP_LO_PTQ_EE': 'Upper 95% confidence limit for English Baccalaureate Science Value Added measure for low prior attainment',
 'SCIVALOW_LO_PTQ_EE': 'Lower 95% confidence limit for English Baccalaureate Science Value Added measure for low prior attainment',
 'SCIVAUPP_MID_PTQ_EE': 'Upper 95% confidence limit for English Baccalaureate Science Value Added measure for middle prior attainment',
 'SCIVALOW_MID_PTQ_EE': 'Lower 95% confidence limit for English Baccalaureate Science Value Added measure for middle prior attainment',
 'SCIVAUPP_HI_PTQ_EE': 'Upper 95% confidence limit for English Baccalaureate Science Value Added measure for high prior attainment',
 'SCIVALOW_HI_PTQ_EE': 'Lower 95% confidence limit for English Baccalaureate Science Value Added measure for high prior attainment'

'SCIVAUPP_HI_PTQ_EE': 'Upper 95% confidence limit for English Baccalaureate Science Value A
 'SCIVALOW_HI_PTQ_EE': 'Lower 95% confidence limit for English Baccalaureate Science Value A
 'HUMVAUPP_FSM6CLA1A_PTQ_EE': 'Upper 95% confidence limit for English Baccalaureate Humaniti
 'HUMVALOW_FSM6CLA1A_PTQ_EE': 'Lower 95% confidence limit for English Baccalaureate Humaniti
 'HUMVAUPP_NFSM6CLA1A_PTQ_EE': 'Upper 95% confidence limit for English Baccalaureate Humanit
 'HUMVALOW_NFSM6CLA1A_PTQ_EE': 'Lower 95% confidence limit for English Baccalaureate Humanit
 'HUMVAUPP_LO_PTQ_EE': 'Upper 95% confidence limit for English Baccalaureate Humanities Valu
 'HUMVALOW_LO_PTQ_EE': 'Lower 95% confidence limit for English Baccalaureate Humanities Valu
 'HUMVAUPP_MID_PTQ_EE': 'Upper 95% confidence limit for English Baccalaureate Humanities Val
 'HUMVALOW_MID_PTQ_EE': 'Lower 95% confidence limit for English Baccalaureate Humanities Val
 'HUMVAUPP_HI_PTQ_EE': 'Upper 95% confidence limit for English Baccalaureate Humanities Valu
 'HUMVALOW_HI_PTQ_EE': 'Lower 95% confidence limit for English Baccalaureate Humanities Valu
 'LANVAUPP_FSM6CLA1A_PTQ_EE': 'Upper 95% confidence limit for English Baccalaureate Languages
 'LANVALOW_FSM6CLA1A_PTQ_EE': 'Lower 95% confidence limit for English Baccalaureate Languages
 'LANVAUPP_NFSM6CLA1A_PTQ_EE': 'Upper 95% confidence limit for English Baccalaureate Language
 'LANVALOW_NFSM6CLA1A_PTQ_EE': 'Lower 95% confidence limit for English Baccalaureate Language
 'LANVAUPP_LO_PTQ_EE': 'Upper 95% confidence limit for English Baccalaureate Languages Value
 'LANVALOW_LO_PTQ_EE': 'Lower 95% confidence limit for English Baccalaureate Languages Value
 'LANVAUPP_MID_PTQ_EE': 'Upper 95% confidence limit for English Baccalaureate Languages Valu
 'LANVALOW_MID_PTQ_EE': 'Lower 95% confidence limit for English Baccalaureate Languages Valu
 'LANVAUPP_HI_PTQ_EE': 'Upper 95% confidence limit for English Baccalaureate Languages Value
 'LANVALOW_HI_PTQ_EE': 'Lower 95% confidence limit for English Baccalaureate Languages Value
 'PTEBACC_E_21_PTQ_EE': '% of pupils entering all English Baccalaureate subject areas in 202
 'PTEBACC_94_21': '% of KS4 pupils achieving the Ebacc - with standard 9-4 passes in English
 'PTEBACC_95_21': '% of KS4 pupils achieving the Ebacc - with strong 9-5 passes in English ar
 'PTEBACC_E_22_PTQ_EE': '% of pupils entering all English Baccalaureate subject areas in 202
 'PTEBACC_94_22': '% of KS4 pupils achieving the Ebacc - with standard 9-4 passes in English
 'PTEBACC_95_22': '% of KS4 pupils achieving the Ebacc - with strong 9-5 passes in English ar
 'PBEBACC_E_PTQ_EE': '% of boys with entries in all English Baccalaureate subject areas',
 'PBEBACC_94': '% of KS4 boys achieving the Ebacc - with 9-4 passes',
 'PBEBACC_95': '% of KS4 boys achieving the Ebacc - with 9-5 passes',
 'PGEBACC_E_PTQ_EE': '% of girls with entries in all English Baccalaureate subject areas',
 'PGEBACC_94': '% of KS4 girls achieving the Ebacc - with 9-4 passes',
 'PGEBACC_95': '% of KS4 girls achieving the Ebacc - with 9-5 passes',
 'PTEBACC_ENMOB_PTQ_EE': '% of non-mobile pupils with entries in all English Baccalaureate su
 'PTEBACCENMOB_94': '% of non-mobile pupils achieving the English Baccalaureate with 9-4 pass
 'PTEBACCENMOB_95': '% of non-mobile pupils achieving the English Baccalaureate with 9-5 pass
 'PTEBACC_EEAL_PTQ_EE': '% of pupils for whom English is an additional language with entries
 'PTEBACCCEAL_94': '% of pupils for whom English as an additional language achieving the Engl
 'PTEBACCCEAL_95': '% of pupils for whom English as an additional language achieving the Engl
 'PTEBACC_EFSM6CLA1A_21': '% of disadvantaged pupils entering all English Baccalaureate subj
 'PTEBACC_94_FSM6CLA1A_21': '% of disadvantaged pupils achieving the English Baccalaureate a
 'PTEBACC_95_FSM6CLA1A_21': '% of disadvantaged pupils achieving the English Baccalaureate a

'PTEBACC_ENFSM6CLA1A_21': '% of non-disadvantaged pupils entering all English Baccalaureate
 'PTEBACC_94_NFSM6CLA1A_21': '% of non-disadvantaged pupils achieving the English Baccalaureate
 'PTEBACC_95_NFSM6CLA1A_21': '% of non-disadvantaged pupils achieving the English Baccalaureate
 'PTEBACC_EFSM6CLA1A_22': '% of disadvantaged pupils entering all English Baccalaureate subj
 'PTEBACC_94_FSM6CLA1A_22': '% of disadvantaged pupils achieving the English Baccalaureate in
 'PTEBACC_95_FSM6CLA1A_22': '% of disadvantaged pupils achieving the English Baccalaureate in
 'PTEBACC_ENFSM6CLA1A_22': '% of non-disadvantaged pupils entering all English Baccalaureate
 'PTEBACC_94_NFSM6CLA1A_22': '% of non-disadvantaged pupils achieving the English Baccalaureate
 'PTEBACC_95_NFSM6CLA1A_22': '% of non-disadvantaged pupils achieving the English Baccalaureate
 'PT5EM_94': '% of pupils achieving Level 2 threshold including standard passes 9-4 in both
 'PT5EM_94_21': '% of pupils achieving Level 2 threshold including standard passes 9-4 in bo
 'PT5EM_94_22': '% of pupils achieving Level 2 threshold including standard passes 9-4 in bo
 'PTANYQ_PTQ_EE': '% of pupils achieving any qualifications',
 'PTL2BASICS_94_21': '% of pupils achieving 9-4 passes in GCSE English and maths in 2021',
 'PTL2BASICS_95_21': '% of pupils achieving 9-5 passes in GCSE English and maths in 2021',
 'PTL2BASICS_94_22': '% of pupils achieving 9-4 passes in GCSE English and maths in 2022',
 'PTL2BASICS_95_22': '% of pupils achieving 9-5 passes in GCSE English and maths in 2022',
 'PTFSM6CLA1ABASICS_94': '% of disadvantaged pupils achieving standard 9-4 passes in GCSE Eng
 'PTNOTFSM6CLA1ABASICS_94': '% of non-disadvantaged pupils achieving standard 9-4 passes in C
 'TBASICSL0_94': 'Number of pupils in low prior attainment band who achieved standard 9-4 pas
 'PTBASICSL0_94': '% of pupils in low prior attainment band who achieved standard 9-4 passes
 'TBASICSAV_94': 'Number of pupils in middle prior attainment band who achieved standard 9-4
 'PTBASICSAV_94': '% pupils in middle prior attainment band who achieved standard 9-4 passes
 'TBASICSHI_94': 'Number of pupils in high prior attainment band who achieved standard 9-4 pa
 'PTBASICSHI_94': '% pupils in high prior attainment band who achieved standard 9-4 passes in
 'PBL2BASICS_94': '% of boys achieving standard 9-4 passes in both English and mathematics GC
 'PGL2BASICS_94': '% of girls achieving standard 9-4 passes in both English and mathematics C
 'PTL2BASICSEAL_94': '% of pupils achieving standard 9-4 passes in both English and mathemat
 'PTL2BASICSNMOB_94': '% of non-mobile pupils achieving standard 9-4 passes in both English a
 'PTFSM6CLA1ABASICS_95': '% of disadvantaged pupils achieving strong 9-5 passes in GCSE Engl
 'PTNOTFSM6CLA1ABASICS_95': '% of non-disadvantaged pupils achieving strong 9-5 passes in GC
 'TBASICSL0_95': 'Number of pupils in low prior attainment band who achieved strong 9-5 pass
 'PTBASICSL0_95': '% of pupils in low prior attainment band who achieved strong 9-5 passes in
 'TBASICSAV_95': 'Number of pupils in middle prior attainment band who achieved strong 9-5 pa
 'PTBASICSAV_95': '% pupils in middle prior attainment band who achieved strong 9-5 passes in
 'TBASICSHI_95': 'Number of pupils in high prior attainment band who achieved strong 9-5 pas
 'PTBASICSHI_95': '% pupils in high prior attainment band who achieved strong 9-5 passes in
 'PBL2BASICS_95': '% of boys achieving strong 9-5 passes in both English and mathematics GCSE
 'PGL2BASICS_95': '% of girls achieving strong 9-5 passes in both English and mathematics GC
 'PTL2BASICSEAL_95': '% of pupils achieving strong 9-5 passes in both English and mathematics
 'PTL2BASICSNMOB_95': '% of non-mobile pupils achieving strong 9-5 passes in both English and
 'PTFSM6CLA1ABASICS_94_21': '% of disadvantaged pupils achieving 9-4 in GCSE English and mat
 'PTFSM6CLA1ABASICS_95_21': '% of disadvantaged pupils achieving 9-4 passes in GCSE English a

'PTNOTFSM6CLA1ABASICS_94_21': '% of non-disadvantaged pupils achieving 9-4 passes in GCSE English',
 'PTNOTFSM6CLA1ABASICS_95_21': '% of non-disadvantaged pupils achieving 9-5 passes in GCSE English',
 'PTFSM6CLA1ABASICS_94_22': '% of disadvantaged pupils achieving 9-4 passes in GCSE English',
 'PTFSM6CLA1ABASICS_95_22': '% of disadvantaged pupils achieving 9-5 passes in GCSE English',
 'PTNOTFSM6CLA1ABASICS_94_22': '% of non-disadvantaged pupils achieving 9-4 passes in GCSE English',
 'PTNOTFSM6CLA1ABASICS_95_22': '% of non-disadvantaged pupils achieving 9-5 passes in GCSE English',
 'PTmultiLan_E': '% of pupils entering more than one language',
 'PTtripleSci_E': '% of pupils entering biology, chemistry and physics',
 'TFSM6CLA1A_21': 'Number of disadvantaged pupils at the end of key stage 4 in 2021',
 'PTFSM6CLA1A_21': '% of pupils at the end of key stage 4 who were disadvantaged in 2021',
 'TNOTFSM6CLA1A_21': 'Number of non-disadvantaged pupils at the end of key stage 4 in 2021',
 'PTNOTFSM6CLA1A_21': '% of pupils at the end of key stage 4 who were not disadvantaged in 2021',
 'TFSM6CLA1A_22': 'Number of disadvantaged pupils in 2022',
 'PTFSM6CLA1A_22': '% of pupils who were disadvantaged in 2022',
 'TNOTFSM6CLA1A_22': 'Number of non-disadvantaged pupils in 2022',
 'PTNOTFSM6CLA1A_22': '% of pupils who were not disadvantaged in 2022',
 'TAVENT_E_3NG_PTQ_EE': 'Average number of KS4 entries per pupil',
 'TAVENT_E_3NG_LO_PTQ_EE': 'Average number of KS4 entries per pupil with low prior attainment',
 'TAVENT_E_3NG_MID_PTQ_EE': 'Average number of KS4 entries per pupil with middle prior attainment',
 'TAVENT_E_3NG_HI_PTQ_EE': 'Average number of KS4 entries per pupil with high prior attainment',
 'TAVENT_E_3NG_FSM6CLA1A_PTQ_EE': 'Average number of KS4 entries per disadvantaged pupil',
 'TAVENT_E_3NG_NFSM6CLA1A_PTQ_EE': 'Average number of KS4 entries per non-disadvantaged pupil',
 'TAVENT_EFSM6CLA1A_21_PTQ_EE': 'Average number of KS4 entries per disadvantaged pupil in 2021',
 'TAVENT_ENFSM6CLA1A_21_PTQ_EE': 'Average number of KS4 entries per non-disadvantaged pupil in 2021',
 'TAVENT_EFSM6CLA1A_22_PTQ_EE': 'Average number of KS4 entries per disadvantaged pupil in 2022',
 'TAVENT_ENFSM6CLA1A_22_PTQ_EE': 'Average number of KS4 entries per non-disadvantaged pupil in 2022',
 'TAVENT_G_PTQ_EE': 'Average number of GCSE entries per pupil',
 'TAVENT_GLO_PTQ_EE': 'Average number of GCSE entries per pupil with low prior attainment',
 'TAVENT_GAV_PTQ_EE': 'Average number of GCSE entries per pupil with middle prior attainment',
 'TAVENT_GHI_PTQ_EE': 'Average number of GCSE entries per pupil with high prior attainment',
 'TAVENT_GFSM6CLA1A_PTQ_EE': 'Average number of GCSE entries per disadvantaged pupil',
 'TAVENT_GNFSM6CLA1A_PTQ_EE': 'Average number of GCSE entries per non-disadvantaged pupil',
 'TAVENT_GFSM6CLA1A_21_PTQ_EE': 'Average number of GCSE entries per disadvantaged pupil in 2021',
 'TAVENT_GNFSM6CLA1A_21_PTQ_EE': 'Average number of GCSE entries per non-disadvantaged pupil in 2021',
 'TAVENT_GFSM6CLA1A_22_PTQ_EE': 'Average number of GCSE entries per disadvantaged pupil in 2022',
 'TAVENT_GNFSM6CLA1A_22_PTQ_EE': 'Average number of GCSE entries per non-disadvantaged pupil in 2022',
 'TTOTENT_E_TOTAL_PTQ_EE': 'Total volume of entries without discounting',
 'TTOTENT_E_COVID_IMPACTED_PTQ_EE': 'Total volume of covid-impacted entries without discounting',
 'PTOTENT_E_COVID_IMPACTED_PTQ_EE': '% of covid-impacted entries out of total number of entries',
 'P8_BANDING': 'Progress 8 banding shown on school performance tables website'

```

school_funding_meta = DataWrangler('data/funding_meta.csv') # this is originally in .xlsx for
school_funding_dict = DataWrangler.make_dictionary(school_funding_meta, 'Variable name', 'Var
school_funding_dict

```

CSV file loaded successfully from data/funding_meta.csv

```

{'academy': 'Academy?',
 'allocation_per_pupil': 'Allocation per Pupil',
 'basic_entitlement_ks3': 'Basic Entitlement KS3',
 'basic_entitlement_ks4': 'Basic Entitlement KS4',
 'basic_entitlement_primary': 'Basic Entitlement Primary',
 'basic_entitlement_total_funding': 'Basic Entitlement Total Funding',
 'coronavirus_recovery_premium_funding': 'Coronavirus (COVID-19) recovery premium funding',
 'deprivation_total_funding': 'Deprivation Total Funding',
 'eal_total_funding': 'EAL Total Funding',
 'exceptional_factors_total_funding': 'Exceptional Factors Total Funding',
 'fsm_funding': 'FSM Funding',
 'fsm6_funding': 'FSM6 Funding',
 'idaci_band_a': 'IDACI Band A',
 'idaci_band_b': 'IDACI Band B',
 'idaci_band_c': 'IDACI Band C',
 'idaci_band_d': 'IDACI Band D',
 'idaci_band_e': 'IDACI Band E',
 'idaci_band_f': 'IDACI Band F',
 'idaci_funding': 'IDACI Funding',
 'lac_total_funding': 'LAC Total Funding',
 'london_fringe': 'London Fringe',
 'lump_sum_total_funding': 'Lump Sum Total Funding',
 'mfg_protection_or_capping_scaling': 'MFG protection (+ve) or capping/scaling (-ve)',
 'minimum_per_pupil_funding': 'Minimum per pupil funding',
 'mobility_total_funding': 'Mobility Total Funding',
 'national_non_domestic_rates_funding': 'National Non Domestic Rates Funding',
 'notional_sen': 'Notional SEN',
 'pe_&_sport_premium': 'PE & Sport Premium funding',
 'pe_&_sport_premium_pupils': 'PE & Sport Premium pupils',
 'pfi_total_funding': 'PFI Total Funding',
 'prior_attainment_total_funding': 'Prior Attainment Total Funding',
 'pupil_premium': 'Pupil Premium funding',
 'pupil_premium_pupils': 'Pupil Premium pupils',
 'School_led_tutoring_funding': 'School-led tutoring funding',
 'school_phase': 'Phase',

```

```

'school_type': 'School type',
'school_ukprn': 'UKPRN',
'schools_supplementary_grant': 'Schools Supplementary Grant funding',
'sparsity_total_funding': 'Sparsity Total Funding',
'split_site_total_funding': 'Split Site Total Funding',
'total_funding': 'Total funding',
'total_number_of_pupils': 'Total Number of Pupils (rounded)',
'total_schools_block_allocation_(post_mfg)': 'Total Schools Block Allocation (Post MFG)',
'total_schools_block_allocation_(pre_mfg)': 'Total Schools Block Allocation (Pre MFG)',
'trust': 'Trust',
'universal_infant_free_school_meals_grant': 'Universal Infant Free School Meals Grant funding'

```

Select Columns from Data

Before re-labeling the columns using the definitions in the dictionaries, it will be more efficient to select the columns needed in each data file. I shall therefor re-define each dataframe according to the selected columns needed.

MAT Performance Data:

```

#only the following columns are needed

ks4_mat_performance_df = ks4_mat_performance.df[['TRUST_NAME', 'TRUST_UID', 'TRUST_ID', 'NUMINST_MATPTINC', 'TPUP_MATPTINC', 'ATT8SCR']]
ks4_mat_performance_df.head()

```

	TRUST_NAME	TRUST_UID	TRUST_ID	NUMINST_MATPTINC	TPUP_MATPTINC	ATT8SCR
0	ACTIVATE LEARNING EDUCATION TRUST			15710	TR02786	6
1	ACER TRUST			15720	TR01414	3
2	RED KITE LEARNING TRUST			15727	TR00969	4
3	CONSILIUM ACADEMIES			15728	TR00082	8
4	BATLEY MULTI ACADEMY TRUST			15729	TR00147	3

Keystage 4 School Performance Data:

```

#following columns are key measures for school performance
ks4_school_performance_df = ks4_school_performance.df[['URN',
    'ATT8SCR',
    'P8MEA',
    'PTFSM6CLA1A_22',
    'PTNOTFSM6CLA1A_22',

```



```

'PTFSM6CLA1ABASICS_95', #462
'PTNOTFSM6CLA1ABASICS_95',
'ATT8SCR_NFSM6CLA1A_22',
'P8MEA_NFSM6CLA1A_22',
'ATT8SCR_FSM6CLA1A_22',
'P8MEA_FSM6CLA1A_22',
'P8MEAMAT_FSM6CLA1A',
'P8MEAENG_FSM6CLA1A',
'P8MEAMAT_NFSM6CLA1A',
'P8MEAENG_NFSM6CLA1A'
]]

```

```
ks4_school_performance_df.head()
```

	URN	ATT8SCR	P8MEA	PTFSM6CLA1A_22	PTNOTFSM6CLA1A_22	PTFSM6CLA1ABASICS
0	100003.0	36.8	NP	NP	NP	NP
1	100001.0	29.4	NP	NP	NP	NP
2	100544.0	6.8	NP	NP	NP	NP
3	NaN	NaN	NaN	NaN	NaN	NaN
4	100053.0	50.3	-0.16	42%	58%	28%

School Demographics:

```

school_demographics_df = school_demographics_df[['URN', 'LANAME', 'LA', 'SCHOOLTYPE', 'MINORGROUP']]
school_demographics_df.head()

```

	URN	LANAME	LA	SCHOOLTYPE	MINORGROUP	RELCHAR	ADMPOL	GENDER	OFSTED
0	100000	City of London	201	Voluntary aided school	Maintained school	Church of England	No		
1	100001	City of London	201	Other independent school	Independent school	NaN			Self
2	100002	City of London	201	Other independent school	Independent school	Church of England	No		
3	100003	City of London	201	Other independent school	Independent school	NaN			No
4	100008	Camden	202	Community school	Maintained school	Does not apply	No		

School Funding:

```

school_funding_df = school_funding_df[['school_urn', 'fsm_funding', 'pupil_premium', 'pupil_premium_per_pupil']]
school_funding_df.head()

```

	school_urn	fsm_funding	pupil_premium	pupil_premium_pupils	School_led_tutoring_funding	total_funding
0	101247	118662	291560	296	49248	85428
1	101241	198479	492993	501	84024	13420
2	101202	75028	209135	151	24138	34395
3	101231	55872	153735	111	18117	26339
4	136028	211782	511215	519	91017	98362

```
#change column name
school_funding_df.rename(columns={'school_urn':'URN'}, inplace=True) # change column name to URN
school_funding_df.head()
```

C:\Users\sagib\AppData\Local\Temp\ipykernel_37860\65584972.py:2: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/user_guide/10min.html#setting-with-copy-warning

```
school_funding_df.rename(columns={'school_urn':'URN'}, inplace=True) # change column name to URN
```

	URN	fsm_funding	pupil_premium	pupil_premium_pupils	School_led_tutoring_funding	total_funding
0	101247	118662	291560	296	49248	85428
1	101241	198479	492993	501	84024	13420
2	101202	75028	209135	151	24138	34395
3	101231	55872	153735	111	18117	26339
4	136028	211782	511215	519	91017	98362

Academies Membership

- Only URN, Trust ID, School Name and Trust Name are needed

```
academies_membership_df = academies_membership_df[['URN','Group UID','Group ID','Establishment Type']]
academies_membership_df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 12637 entries, 0 to 12636
Data columns (total 5 columns):
#   Column          Non-Null Count  Dtype
---  -
0   URN              12618 non-null  float64
1   Group UID        12463 non-null  float64
```

```

2   Group ID          12463 non-null object
3   EstablishmentName 12618 non-null object
4   Group Name       12463 non-null object
dtypes: float64(2), object(3)
memory usage: 493.8+ KB

```

Merging DfE Data

I can now begin merging the various DfE data based on school URN

Merge: school demographics and school funding

```

school_funding_df.columns
school_funding_df.rename(columns={'school_urn': 'URN'}, inplace=True) # change name of URN column
school_funding_df.columns

```

C:\Users\saqib\AppData\Local\Temp\ipykernel_37860\2541994176.py:2: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/user_guide

```

school_funding_df.rename(columns={'school_urn': 'URN'}, inplace=True) # change name of URN column

```

```

Index(['URN', 'fsm_funding', 'pupil_premium', 'pupil_premium_pupils',
      'School_led_tutoring_funding', 'total_funding'],
      dtype='object')

```

```

# Merge School Information with Funding Data
merged_df = pd.merge(school_demographics_df, school_funding_df, on='URN', how='inner')
#chosen an inner join, as having incomplete left or right fields will not be of use
print("School Demographics and Funding data merged.")

merged_df.shape

```

School Demographics and Funding data merged.

```

(19973, 15)

```

```
merged_df.columns
```

```
Index(['URN', 'LANAME', 'LA', 'SCHOOLTYPE', 'MINORGROUP', 'RELCHAR', 'ADMPOL',  
      'GENDER', 'OFSTEDRATING', 'POSTCODE', 'fsm_funding', 'pupil_premium',  
      'pupil_premium_pupils', 'School_led_tutoring_funding', 'total_funding'],  
      dtype='object')
```

Merge MAT info

```
# Merge with MAT Performance  
merged_df = pd.merge(merged_df, academies_membership_df, on= ['URN'], how='left')  
# some schools may not have an academy therefore a left join  
print("Merged with MAT Performance data.")
```

Merged with MAT Performance data.

```
merged_df['URN'].nunique() # count how many unique schools exist and therefore if some are d
```

19973

Some of the URNs may be duplicates and will need to be dropped later on when I conduct data cleaning.

merge school performance

```
merged_df = pd.merge(merged_df, ks4_school_performance_df, on='URN', how='inner')  
# inner join essential as having only 'right' or 'left' data wouldnt be of much use  
print("School KS4 performance data merged.")  
merged_df.head()
```

School KS4 performance data merged.

	URN	LANAME	LA	SCHOOLTYPE	MINORGROUP	RELCHAR	ADMPOL	GENDER	OFST
0	100049	Camden	202	Community school	Maintained school	Does not apply	Non-selective	Mix	
1	100050	Camden	202	Community school	Maintained school	Does not apply	Non-selective	Girl	
2	100051	Camden	202	Community school	Maintained school	Does not apply	Non-selective	Mix	
3	100052	Camden	202	Community school	Maintained school	Does not apply	Non-selective	Mix	
4	100053	Camden	202	Community school	Maintained school	Does not apply	Non-selective	Mix	

```
merged_df['URN'].nunique()
```

3281

Observation: The number of unique schools has dropped from 19k to 3k, when the keystage 4 data was merged on an inner join. This is expected as reportedly 3444 state-funded secondary schools in England, with private schools included it is approximately 4175 [7]. Of these a number by newly opened and not have delivered GCSE in 2022/23

[7] Tes. (2024, January 17). How many schools are there in the UK? Retrieved from <https://www.tes.com/magazine/analysis/general/how-many-schools-in-the-uk>

merged MAT performance data

```
merged_df = pd.merge(merged_df, ks4_mat_performance_df, left_on=['Group ID'], right_on=['TRU
print("Merged with MAT Performance data.")
```

Merged with MAT Performance data.

```
merged_df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 3866 entries, 0 to 3865
Data columns (total 41 columns):
#   Column                                Non-Null Count  Dtype
---  -
0   URN                                  3866 non-null   int64
```

1	LANAME	3866 non-null	object
2	LA	3866 non-null	int64
3	SCHOOLTYPE	3866 non-null	object
4	MINORGROUP	3866 non-null	object
5	RELCHAR	2399 non-null	object
6	ADMPOL	3601 non-null	object
7	GENDER	3866 non-null	object
8	OFSTEDRATING	3822 non-null	object
9	POSTCODE	3866 non-null	object
10	fsm_funding	3866 non-null	int64
11	pupil_premium	3866 non-null	object
12	pupil_premium_pupils	3866 non-null	object
13	School_led_tutoring_funding	3866 non-null	object
14	total_funding	3866 non-null	float64
15	Group UID	3214 non-null	float64
16	Group ID	3214 non-null	object
17	EstablishmentName	3214 non-null	object
18	Group Name	3214 non-null	object
19	ATT8SCR	3807 non-null	object
20	P8MEA	3807 non-null	object
21	PTFSM6CLA1A_22	3770 non-null	object
22	PTNOTFSM6CLA1A_22	3770 non-null	object
23	PTFSM6CLA1ABASICS_95	3807 non-null	object
24	PTNOTFSM6CLA1ABASICS_95	3807 non-null	object
25	ATT8SCR_NFSM6CLA1A_22	3770 non-null	object
26	P8MEA_NFSM6CLA1A_22	3770 non-null	object
27	ATT8SCR_FSM6CLA1A_22	3770 non-null	object
28	P8MEA_FSM6CLA1A_22	3770 non-null	object
29	P8MEAMAT_FSM6CLA1A	3807 non-null	object
30	P8MEAENG_FSM6CLA1A	3807 non-null	object
31	P8MEAMAT_NFSM6CLA1A	3807 non-null	object
32	P8MEAENG_NFSM6CLA1A	3807 non-null	object
33	TRUST_NAME	1346 non-null	object
34	TRUST_UID	1346 non-null	float64
35	TRUST_ID	1346 non-null	object
36	NUMINST_MATPTINC	1346 non-null	float64
37	TPUP_MATPTINC	1346 non-null	float64
38	ATT8SCR_WGTAVG	1346 non-null	float64
39	P8MEA_WGTAVG	1346 non-null	float64
40	TIME_PERIOD	1346 non-null	float64

dtypes: float64(8), int64(3), object(30)

memory usage: 1.2+ MB

Data Cleaning

Now that the merging is complete, I can now remove rows which are not needed

Remove NaN values

```
merged_df.isna().sum()
```

URN	0
LANAME	0
LA	0
SCHOOLTYPE	0
MINORGROUP	0
RELCHAR	1467
ADMPOL	265
GENDER	0
OFSTEDRATING	44
POSTCODE	0
fsm_funding	0
pupil_premium	0
pupil_premium_pupils	0
School_led_tutoring_funding	0
total_funding	0
Group UID	652
Group ID	652
EstablishmentName	652
Group Name	652
ATT8SCR	59
P8MEA	59
PTFSM6CLA1A_22	96
PTNOTFSM6CLA1A_22	96
PTFSM6CLA1ABASICS_95	59
PTNOTFSM6CLA1ABASICS_95	59
ATT8SCR_NFSM6CLA1A_22	96
P8MEA_NFSM6CLA1A_22	96
ATT8SCR_FSM6CLA1A_22	96
P8MEA_FSM6CLA1A_22	96
P8MEAMAT_FSM6CLA1A	59
P8MEAENG_FSM6CLA1A	59
P8MEAMAT_NFSM6CLA1A	59

P8MEAENG_NFSM6CLA1A	59
TRUST_NAME	2520
TRUST_UID	2520
TRUST_ID	2520
NUMINST_MATPTINC	2520
TPUP_MATPTINC	2520
ATT8SCR_WGTAVG	2520
P8MEA_WGTAVG	2520
TIME_PERIOD	2520
dtype:	int64

There may not be a need to drop all NaN values in every variable, as some schools may not have an OFSTED rating nor be part of a Trust in 2022/23, and I wouldn't want to discard the rest of their data from analysis

```
merged_df = merged_df.dropna(subset=['P8MEA_FSM6CLA1A_22'])
# Im not dropping all NaN values, as some schools may not have an OFSTED rating nor be part of
merged_df.isna().sum()
```

URN	0
LANAME	0
LA	0
SCHOOLTYPE	0
MINORGROUP	0
RELCHAR	1403
ADMPOL	248
GENDER	0
OFSTEDRATING	33
POSTCODE	0
fsm_funding	0
pupil_premium	0
pupil_premium_pupils	0
School_led_tutoring_funding	0
total_funding	0
Group UID	650
Group ID	650
EstablishmentName	650
Group Name	650
ATT8SCR	0
P8MEA	0
PTFSM6CLA1A_22	0
PTNOTFSM6CLA1A_22	0

PTFSM6CLA1ABASICS_95	0
PTNOTFSM6CLA1ABASICS_95	0
ATT8SCR_NFSM6CLA1A_22	0
P8MEA_NFSM6CLA1A_22	0
ATT8SCR_FSM6CLA1A_22	0
P8MEA_FSM6CLA1A_22	0
P8MEAMAT_FSM6CLA1A	0
P8MEAENG_FSM6CLA1A	0
P8MEAMAT_NFSM6CLA1A	0
P8MEAENG_NFSM6CLA1A	0
TRUST_NAME	2474
TRUST_UID	2474
TRUST_ID	2474
NUMINST_MATPTINC	2474
TPUP_MATPTINC	2474
ATT8SCR_WGTAVG	2474
P8MEA_WGTAVG	2474
TIME_PERIOD	2474
dtype: int64	

Remove Duplicates

We can now check for duplicates and remove them

```
duplicate_urns = merged_df[merged_df.duplicated('URN', keep=False)]
#keep= False will mark all duplicates as True regardless of position
duplicate_urns.info()
```

```
<class 'pandas.core.frame.DataFrame'>
```

```
Index: 1114 entries, 600 to 3819
```

```
Data columns (total 41 columns):
```

#	Column	Non-Null Count	Dtype
0	URN	1114 non-null	int64
1	LANAME	1114 non-null	object
2	LA	1114 non-null	int64
3	SCHOOLTYPE	1114 non-null	object
4	MINORGROUP	1114 non-null	object
5	RELCHAR	654 non-null	object
6	ADMPOL	1031 non-null	object
7	GENDER	1114 non-null	object

8	OFSTEDRATING	1106 non-null	object
9	POSTCODE	1114 non-null	object
10	fsm_funding	1114 non-null	int64
11	pupil_premium	1114 non-null	object
12	pupil_premium_pupils	1114 non-null	object
13	School_led_tutoring_funding	1114 non-null	object
14	total_funding	1114 non-null	float64
15	Group UID	1114 non-null	float64
16	Group ID	1114 non-null	object
17	EstablishmentName	1114 non-null	object
18	Group Name	1114 non-null	object
19	ATT8SCR	1114 non-null	object
20	P8MEA	1114 non-null	object
21	PTFSM6CLA1A_22	1114 non-null	object
22	PTNOTFSM6CLA1A_22	1114 non-null	object
23	PTFSM6CLA1ABASICS_95	1114 non-null	object
24	PTNOTFSM6CLA1ABASICS_95	1114 non-null	object
25	ATT8SCR_NFSM6CLA1A_22	1114 non-null	object
26	P8MEA_NFSM6CLA1A_22	1114 non-null	object
27	ATT8SCR_FSM6CLA1A_22	1114 non-null	object
28	P8MEA_FSM6CLA1A_22	1114 non-null	object
29	P8MEAMAT_FSM6CLA1A	1114 non-null	object
30	P8MEAENG_FSM6CLA1A	1114 non-null	object
31	P8MEAMAT_NFSM6CLA1A	1114 non-null	object
32	P8MEAENG_NFSM6CLA1A	1114 non-null	object
33	TRUST_NAME	386 non-null	object
34	TRUST_UID	386 non-null	float64
35	TRUST_ID	386 non-null	object
36	NUMINST_MATPTINC	386 non-null	float64
37	TPUP_MATPTINC	386 non-null	float64
38	ATT8SCR_WGTAVG	386 non-null	float64
39	P8MEA_WGTAVG	386 non-null	float64
40	TIME_PERIOD	386 non-null	float64

dtypes: float64(8), int64(3), object(30)

memory usage: 365.5+ KB

```
#drop duplicates
merged_df = merged_df.drop_duplicates(subset='URN')
#confirm duplicates are removed
duplicate_urns = merged_df[merged_df.duplicated('URN', keep=False)]
duplicate_urns.info()
```

```
<class 'pandas.core.frame.DataFrame'>
```

```
Index: 0 entries
```

```
Data columns (total 41 columns):
```

#	Column	Non-Null Count	Dtype
0	URN	0 non-null	int64
1	LANAME	0 non-null	object
2	LA	0 non-null	int64
3	SCHOOLTYPE	0 non-null	object
4	MINORGROUP	0 non-null	object
5	RELCHAR	0 non-null	object
6	ADMPOL	0 non-null	object
7	GENDER	0 non-null	object
8	OFSTEDRATING	0 non-null	object
9	POSTCODE	0 non-null	object
10	fsm_funding	0 non-null	int64
11	pupil_premium	0 non-null	object
12	pupil_premium_pupils	0 non-null	object
13	School_led_tutoring_funding	0 non-null	object
14	total_funding	0 non-null	float64
15	Group UID	0 non-null	float64
16	Group ID	0 non-null	object
17	EstablishmentName	0 non-null	object
18	Group Name	0 non-null	object
19	ATT8SCR	0 non-null	object
20	P8MEA	0 non-null	object
21	PTFSM6CLA1A_22	0 non-null	object
22	PTNOTFSM6CLA1A_22	0 non-null	object
23	PTFSM6CLA1ABASICS_95	0 non-null	object
24	PTNOTFSM6CLA1ABASICS_95	0 non-null	object
25	ATT8SCR_NFSM6CLA1A_22	0 non-null	object
26	P8MEA_NFSM6CLA1A_22	0 non-null	object
27	ATT8SCR_FSM6CLA1A_22	0 non-null	object
28	P8MEA_FSM6CLA1A_22	0 non-null	object
29	P8MEAMAT_FSM6CLA1A	0 non-null	object
30	P8MEAENG_FSM6CLA1A	0 non-null	object
31	P8MEAMAT_NFSM6CLA1A	0 non-null	object
32	P8MEAENG_NFSM6CLA1A	0 non-null	object
33	TRUST_NAME	0 non-null	object
34	TRUST_UID	0 non-null	float64
35	TRUST_ID	0 non-null	object
36	NUMINST_MATPTINC	0 non-null	float64
37	TPUP_MATPTINC	0 non-null	float64

```

38 ATT8SCR_WGTAVG          0 non-null    float64
39 P8MEA_WGTAVG            0 non-null    float64
40 TIME_PERIOD             0 non-null    float64
dtypes: float64(8), int64(3), object(30)
memory usage: 0.0+ bytes

```

Correct Data Types

I will not proceed to check the data is in the format needed, particularly for numerical analysis.

```
merged_df.dtypes
```

```

URN                int64
LANAME             object
LA                int64
SCHOOLTYPE         object
MINORGROUP         object
RELCHAR            object
ADMPOL             object
GENDER            object
OFSTEDRATING       object
POSTCODE           object
fsm_funding        int64
pupil_premium      object
pupil_premium_pupils object
School_led_tutoring_funding object
total_funding      float64
Group UID          float64
Group ID           object
EstablishmentName  object
Group Name         object
ATT8SCR            object
P8MEA              object
PTFSM6CLA1A_22     object
PTNOTFSM6CLA1A_22  object
PTFSM6CLA1ABASICS_95 object
PTNOTFSM6CLA1ABASICS_95 object
ATT8SCR_NFSM6CLA1A_22 object
P8MEA_NFSM6CLA1A_22 object
ATT8SCR_FSM6CLA1A_22 object

```

```

P8MEA_FSM6CLA1A_22      object
P8MEAMAT_FSM6CLA1A      object
P8MEAENG_FSM6CLA1A      object
P8MEAMAT_NFSM6CLA1A     object
P8MEAENG_NFSM6CLA1A     object
TRUST_NAME               object
TRUST_UID                float64
TRUST_ID                 object
NUMINST_MATPTINC         float64
TPUP_MATPTINC            float64
ATT8SCR_WGTAVG           float64
P8MEA_WGTAVG             float64
TIME_PERIOD              float64
dtype: object

```

A number of the numerical columns are listed as objects and will need to be changed to a numerical type (integer or float). However, before that, we would need to identify and remove any signs in the data e.g. £ or %

Identified columns which have a % in their data and would need removing

```

percentage_columns = [
    'PTFSM6CLA1A_22',
    'PTNOTFSM6CLA1A_22',
    'PTFSM6CLA1ABASICS_95',
    'PTNOTFSM6CLA1ABASICS_95'
]

merged_df[percentage_columns].head()

```

	PTFSM6CLA1A_22	PTNOTFSM6CLA1A_22	PTFSM6CLA1ABASICS_95	PTNOTFSM6CLA1ABA
0	63%	38%	40%	48%
1	39%	61%	53%	76%
2	72%	28%	35%	48%
3	45%	55%	31%	53%
4	42%	58%	28%	74%

```

data_loader = DataWrangler(dataframe=merged_df)

# Convert percentage columns

```

```
merged_df = data_loader.convert_percentage_columns(percentage_columns)

print("\nAfter removing '%' signs and converting to float:")
merged_df[percentage_columns].head()
```

DataWrangler initialised with the provided DataFrame.
 Column 'PTFSM6CLA1A_22' converted
 Column 'PTNOTFSM6CLA1A_22' converted
 Column 'PTFSM6CLA1ABASICS_95' converted
 Column 'PTNOTFSM6CLA1ABASICS_95' converted

After removing '%' signs and converting to float:

	PTFSM6CLA1A_22	PTNOTFSM6CLA1A_22	PTFSM6CLA1ABASICS_95	PTNOTFSM6CLA1ABA
0	63	38	40	48
1	39	61	53	76
2	72	28	35	48
3	45	55	31	53
4	42	58	28	74

I can now convert all 'numerical columns' to their correct data type

```
columns_to_convert_numeric = [
    'fsm_funding',
    'pupil_premium',
    'pupil_premium_pupils',
    'School_led_tutoring_funding',
    'ATT8SCR',
    'P8MEA',
    'PTFSM6CLA1A_22',
    'PTNOTFSM6CLA1A_22',
    'PTFSM6CLA1ABASICS_95',
    'PTNOTFSM6CLA1ABASICS_95',
    'ATT8SCR_NFSM6CLA1A_22',
    'P8MEA_NFSM6CLA1A_22',
    'ATT8SCR_FSM6CLA1A_22',
    'P8MEA_FSM6CLA1A_22',
    'P8MEAMAT_FSM6CLA1A',
```

```

        'P8MEAENG_FSM6CLA1A',
        'P8MEAMAT_NFSM6CLA1A',
        'P8MEAENG_NFSM6CLA1A',
    ]

    # Convert specified columns to numeric, coercing errors to NaN
    merged_df[columns_to_convert_numeric] = merged_df[columns_to_convert_numeric].apply(pd.to_numeric)

    print("Data types after conversion:")
    merged_df.dtypes

```

Data types after conversion:

URN	int64
LANAME	object
LA	int64
SCHOOLTYPE	object
MINORGROUP	object
RELCHAR	object
ADMPOL	object
GENDER	object
OFSTEDRATING	object
POSTCODE	object
fsm_funding	int64
pupil_premium	int64
pupil_premium_pupils	int64
School_led_tutoring_funding	float64
total_funding	float64
Group UID	float64
Group ID	object
EstablishmentName	object
Group Name	object
ATT8SCR	float64
P8MEA	float64
PTFSM6CLA1A_22	int64
PTNOTFSM6CLA1A_22	int64
PTFSM6CLA1ABASICS_95	float64
PTNOTFSM6CLA1ABASICS_95	float64
ATT8SCR_NFSM6CLA1A_22	float64
P8MEA_NFSM6CLA1A_22	float64
ATT8SCR_FSM6CLA1A_22	float64

P8MEA_FSM6CLA1A_22	float64
P8MEAMAT_FSM6CLA1A	float64
P8MEAENG_FSM6CLA1A	float64
P8MEAMAT_NFSM6CLA1A	float64
P8MEAENG_NFSM6CLA1A	float64
TRUST_NAME	object
TRUST_UID	float64
TRUST_ID	object
NUMINST_MATPTINC	float64
TPUP_MATPTINC	float64
ATT8SCR_WGTAVG	float64
P8MEA_WGTAVG	float64
TIME_PERIOD	float64
dtype:	object

Nomenclature

Using the dictionaries created earlier from the meta data, I can run the column rename function to only rename the columns available in merged_df

```
data_loader = DataWrangler(dataframe=merged_df)

#rename columns based on dictionary
merged_df = data_loader.column_rename(school_performance_dict)
merged_df = data_loader.column_rename(ks4_mat_performance_dict)
merged_df = data_loader.column_rename(school_demographics_dict)
merged_df = data_loader.column_rename(school_funding_dict)
merged_df.info()
```

DataWrangler initialised with the provided DataFrame.

Columns renamed successfully.

Columns renamed successfully.

Columns renamed successfully.

Columns renamed successfully.

<class 'pandas.core.frame.DataFrame'>

Index: 3190 entries, 0 to 3862

Data columns (total 41 columns):

#	Column
0	School Unique Reference Number
1	Local authority name

Non

31
31

2	Local authority number	31
3	School Type eg Voluntary Aided school	31
4	Type of school / college eg maintained school	31
5	Religious character	20
6	School admissions policy (self-declared by schools on Edubase)	29
7	Indicates whether it's a mixed or single sex school	31
8	Ofsted rating	31
9	School postcode	31
10	FSM Funding	31
11	Pupil Premium funding	31
12	Pupil Premium pupils	31
13	School-led tutoring funding	31
14	Total funding	31
15	Group UID	25
16	Group ID	25
17	EstablishmentName	25
18	Group Name	25
19	Average Attainment 8 score per pupil	31
20	Progress 8 measure after adjustment for extreme scores	31
21	% of pupils who were disadvantaged in 2022	31
22	% of pupils who were not disadvantaged in 2022	31
23	% of disadvantaged pupils achieving strong 9-5 passes in GCSE English and maths	31
24	% of non-disadvantaged pupils achieving strong 9-5 passes in GCSE English and maths	31
25	Average Attainment 8 score per non-disadvantaged pupil - 2022	31
26	Progress 8 measure - non-disadvantaged pupils - 2022	31
27	Average Attainment 8 score per disadvantaged pupil - 2022	31
28	Progress 8 measure - disadvantaged pupils - 2022	31
29	Progress 8 measure for maths element - disadvantaged pupils	31
30	Progress 8 measure for English element - disadvantaged pupils	31
31	Progress 8 measure for maths element - non-disadvantaged pupils	31
32	Progress 8 measure for English element - non-disadvantaged pupils	31
33	Trust name	10
34	Trust Unique identifier	10
35	Trust Identifier	10
36	Number of academies in the trust, included in performance measures	10
37	Number of pupils at the end of ks4, included in performance measures	10
38	Average Attainment 8 score per pupil at the end of KS4, weighted average	10
39	Progress 8 measure after adjustment for extreme scores, weighted average	10
40	nan	10

dtypes: float64(21), int64(7), object(13)

memory usage: 1.0+ MB

The columns have a new name based on a description. This can now be changed to a more

column friendly format using a new dictionary:

```
# A dictionary mapping old column names to new column names
column_rename_dict = {
    'School Unique Reference Number': 'URN',
    'Local authority name': 'Local_Authority_Name',
    'Local authority number': 'Local_Authority_Number',
    'School Type eg Voluntary Aided school': 'School_Type',
    'Type of school / college eg maintained school': 'School_College_Type',
    'Religious character': 'Religious_Character',
    'School admissions policy (self-declared by schools on Edubase)': 'Admissions_Policy',
    'Indicates whether it\'s a mixed or single sex school': 'School_Gender',
    'Ofsted rating': 'Ofsted_Rating',
    'FSM Funding': 'FSM_Funding',
    'Pupil Premium funding': 'Pupil_Premium_Funding',
    'Pupil Premium pupils': 'Pupil_Premium_Pupils',
    'School-led tutoring funding': 'School_Led_Tutoring_Funding',
    'Total funding': 'Total_Funding',
    'Group UID': 'Group_UID',
    'Group ID': 'Group_ID',
    'EstablishmentName': 'School_Name',
    'Group Name': 'Trust_Name', #first option for Trust Name
    'Average Attainment 8 score per pupil': 'Attainment8',
    'Progress 8 measure after adjustment for extreme scores': 'Progress8',
    '% of pupils who were disadvantaged in 2022': 'Percent_Disadvantaged_2022',
    '% of pupils who were not disadvantaged in 2022': 'Percent_Not_Disadvantaged_2022',
    '% of disadvantaged pupils achieving strong 9-5 passes in GCSE English and maths': 'Percent_Disadvantaged_Pupils_9_5_Passes_2022',
    '% of non-disadvantaged pupils achieving strong 9-5 passes in GCSE English and maths': 'Percent_Non_Disadvantaged_Pupils_9_5_Passes_2022',
    'Average Attainment 8 score per non-disadvantaged pupil - 2022': 'Attainment8_NonDisadvantaged_Pupil_2022',
    'Progress 8 measure - non-disadvantaged pupils - 2022': 'Progress8_NonDisadvantaged_Pupils_2022',
    'Average Attainment 8 score per disadvantaged pupil - 2022': 'Attainment8_Disadvantaged_Pupil_2022',
    'Progress 8 measure - disadvantaged pupils - 2022': 'Progress8_Disadvantaged_Pupils_2022',
    'Progress 8 measure for maths element - disadvantaged pupils': 'Progress8_Maths_Disadvantaged_Pupils_2022',
    'Progress 8 measure for English element - disadvantaged pupils': 'Progress8_English_Disadvantaged_Pupils_2022',
    'Progress 8 measure for maths element - non-disadvantaged pupils': 'Progress8_Maths_NonDisadvantaged_Pupils_2022',
    'Progress 8 measure for English element - non-disadvantaged pupils': 'Progress8_English_NonDisadvantaged_Pupils_2022',
    'Trust name': 'trust_name', # second option to match Trust and quality assure data
    'Trust Unique identifier': 'Trust_UID',
    'Trust Identifier': 'Trust_ID',
    'Number of academies in the trust, included in performance measures': 'Num_Academies_Performance_Measures',
    'Number of pupils at the end of ks4, included in performance measures': 'Num_Pupils_KS4_Performance_Measures',
    'Average Attainment 8 score per pupil at the end of KS4, weighted average': 'Avg_Attainment8_KS4_Weighted_Average',
    'Progress 8 measure after adjustment for extreme scores, weighted average': 'Progress8_After_Adjustment_Weighted_Average'
}
```

```
'nan': 'Time_Period' # I can remove if not needed and time analysis isnt conducted
}
```

This new dictionary will now be used to rename the columns to a more userfriendly format

```
# Rename the columns in the DataFrame using new dictionary
merged_df.rename(columns=column_rename_dict, inplace=True)

merged_df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
```

```
Index: 3190 entries, 0 to 3862
```

```
Data columns (total 41 columns):
```

#	Column	Non-Null Count	Dtype
0	URN	3190 non-null	int64
1	Local_Authority_Name	3190 non-null	object
2	Local_Authority_Number	3190 non-null	int64
3	School_Type	3190 non-null	object
4	School_College_Type	3190 non-null	object
5	Religious_Character	2028 non-null	object
6	Admissions_Policy	2985 non-null	object
7	School_Gender	3190 non-null	object
8	Ofsted_Rating	3161 non-null	object
9	School_postcode	3190 non-null	object
10	FSM_Funding	3190 non-null	int64
11	Pupil_Premium_Funding	3190 non-null	int64
12	Pupil_Premium_Pupils	3190 non-null	int64
13	School_Led_Tutoring_Funding	3188 non-null	float64
14	Total_Funding	3190 non-null	float64
15	Group_UID	2540 non-null	float64
16	Group_ID	2540 non-null	object
17	School_Name	2540 non-null	object
18	Trust_Name	2540 non-null	object
19	Attainment8	3190 non-null	float64
20	Progress8	3187 non-null	float64
21	Percent_Disadvantaged_2022	3190 non-null	int64
22	Percent_Not_Disadvantaged_2022	3190 non-null	int64
23	Percent_Disadvantaged_Strong_Passes	3145 non-null	float64
24	Percent_Not_Disadvantaged_Strong_Passes	3147 non-null	float64
25	Average Attainment 8 score per non-disadvantaged pupil - 2022	3149 non-null	float64

```

26 Progress8_NonDisadvantaged_2022      3147 non-null float64
27 Attainment8_Disadvantaged_2022      3147 non-null float64
28 Progress8_Disadvantaged_2022        3139 non-null float64
29 Progress8_Maths_Disadvantaged        3134 non-null float64
30 Progress8_English_Disadvantaged      3134 non-null float64
31 Progress8_Maths_NonDisadvantaged     3142 non-null float64
32 Progress8_English_NonDisadvantaged   3142 non-null float64
33 trust_name                          1077 non-null object
34 Trust_UID                           1077 non-null float64
35 Trust_ID                            1077 non-null object
36 Num_Academies_Performance            1077 non-null float64
37 Num_Pupils_KS4_Performance           1077 non-null float64
38 Avg_Attainment8_KS4_Weighted         1077 non-null float64
39 Progress8_Adjusted_Weighted          1077 non-null float64
40 nan                                  1077 non-null float64
dtypes: float64(21), int64(7), object(13)
memory usage: 1.0+ MB

```

```
merged_df.head()
```

	URN	Local_Authority_Name	Local_Authority_Number	School_Type	School_College_Type	Reli
0	100049	Camden	202	Community school	Maintained school	
1	100050	Camden	202	Community school	Maintained school	
2	100051	Camden	202	Community school	Maintained school	
3	100052	Camden	202	Community school	Maintained school	
4	100053	Camden	202	Community school	Maintained school	

Add Deprivation Index

I will now add the deprivation information from the Ministry of Housing, Communities and Local Government. The website return deprivation information based on a postcode. However before that, I will create a variable from the school demographics, which has a postcode for each school's URN

```
school_demographics_df.columns
```

```

Index(['URN', 'LANAME', 'LA', 'SCHOOLTYPE', 'MINORGROUP', 'RELCHAR', 'ADMPOL',
      'GENDER', 'OFSTEDRATING', 'POSTCODE'],
      dtype='object')

```

```
urn_postcode = school_demographics_df[['URN', 'POSTCODE']]
urn_postcode.dropna(inplace=True)
```

C:\Users\saqib\AppData\Local\Temp\ipykernel_37860\3856292752.py:2: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/user_guide
urn_postcode.dropna(inplace=True)

```
urn_postcode.drop_duplicates(subset='URN', inplace=True) # drop duplicates
total_postcodes = len(urn_postcode)
total_postcodes
```

C:\Users\saqib\AppData\Local\Temp\ipykernel_37860\3769423262.py:1: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/user_guide
urn_postcode.drop_duplicates(subset='URN', inplace=True) # drop duplicates

25112

As the MHCLG website only allows for 10,000 postcodes to be uploaded at a time, I will need to split the urn_postcode into smaller groups, to upload to the website

```
total_postcode_1 = urn_postcode.iloc[0:9000]
total_postcode_2 = urn_postcode.iloc[9001:18000]
total_postcode_3= urn_postcode.iloc[18001:25135]

output_path1 = 'data/urn_postcode_list1.csv'
output_path2 = 'data/urn_postcode_list2.csv'
output_path3 = 'data/urn_postcode_list3.csv'
total_postcode_1['POSTCODE'].to_csv(output_path1, index=False)
total_postcode_2['POSTCODE'].to_csv(output_path2, index=False)
total_postcode_3['POSTCODE'].to_csv(output_path3, index=False)
```

Read and convert the deprivation data download from the MHCLG website into data frames

```
deprivation_by_index1 = pd.read_csv(r'data\deprivation-by-postcode (1).csv')
deprivation_by_index2 = pd.read_csv(r'data\deprivation-by-postcode (2).csv')
deprivation_by_index3 = pd.read_csv(r'data\deprivation-by-postcode (3).csv')
```

Check on the shape of the data

```
deprivation_by_index1.shape, deprivation_by_index2.shape, deprivation_by_index3.shape
```

```
((9001, 28), (9000, 28), (7135, 28))
```

Combine the three data frames together

```
combined_deprivation = pd.concat([deprivation_by_index1, deprivation_by_index2, deprivation_by_index3])
combined_deprivation.shape
```

```
(25136, 28)
```

Check columns and data types

```
combined_deprivation.dtypes
```

Postcode	object
Postcode Status	object
LSOA code	object
LSOA Name	object
Index of Multiple Deprivation Rank	float64
Index of Multiple Deprivation Decile	float64
Income Rank	float64
Income Decile	float64
Income Score	float64
Employment Rank	float64
Employment Decile	float64
Employment Score	float64
Education and Skills Rank	float64
Education and Skills Decile	float64
Health and Disability Rank	float64
Health and Disability Decile	float64
Crime Rank	float64
Crime Decile	float64
Barriers to Housing and Services Rank	float64
Barriers to Housing and Services Decile	float64
Living Environment Rank	float64
Living Environment Decile	float64

```
IDACI Rank          float64
IDACI Decile        float64
IDACI Score         float64
IDAOPI Rank         float64
IDAOPI Decile       float64
IDAOPI Score        float64
dtype: object
```

```
urn_postcode.dtypes
```

```
URN          int64
POSTCODE     object
dtype: object
```

```
#rename postcode to match
combined_deprivation.rename(columns={'Postcode':'POSTCODE'}, inplace=True)
```

I can now combine deprivation with school URN fields

```
deprivation_urn = combined_deprivation.merge(urn_postcode, on='POSTCODE', how='inner')
#I selected an inner join as I am only interested in deprivation data that can be linked to a
deprivation_urn.shape
deprivation_urn.head()
```

	POSTCODE	Postcode Status	LSOA code	LSOA Name	Index of Multiple Deprivation Rank	Index of Multiple Deprivation Decile
0	EC3A 5DE	Live	E01032739	City of London 001F	E01032739	20391.0
1	EC2Y 8BB	Live	E01000002	City of London 001B	E01000002	30379.0
2	EC4M 9AD	Live	E01032739	City of London 001F	E01032739	20391.0
3	EC4V 3AL	Live	E01032739	City of London 001F	E01032739	20391.0
4	WC1H 9EG	Live	E01000941	Camden 025C	E01000941	4860.0

Of the various columns available from the MHCLG, I will use Index of Multiple Deprivation Decile as this gives a values between 1-10 for each postcode or small geographic areas known as LSOA (lower-layer super output areas); with decile 1 representing the most 10% of deprived areas and a decile of 10 representing the least deprived areas. The multiple deprivation index is calculated from several domains including income, employment, education, health, crime, barriers to housing and services and living environment. The LSOA Name is also selected to verify against merged_df columns when combined later, as a data integrity measure.

```
deprivation_urn= deprivation_urn[['Index of Multiple Deprivation Decile','LSOA Name','POSTCODE']]
```

```
merged_df = merged_df.merge(deprivation_urn, on= 'URN', how ='inner')
# i will do an inner join here as deprivation is an essential criteria for analysis
```

```
merged_df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
```

```
RangeIndex: 3685 entries, 0 to 3684
```

```
Data columns (total 44 columns):
```

#	Column	Non-Null Count	Dtype
0	URN	3685 non-null	int64
1	Local_Authority_Name	3685 non-null	object
2	Local_Authority_Number	3685 non-null	int64
3	School_Type	3685 non-null	object
4	School_College_Type	3685 non-null	object
5	Religious_Character	2353 non-null	object
6	Admissions_Policy	3440 non-null	object
7	School_Gender	3685 non-null	object
8	Ofsted_Rating	3653 non-null	object
9	School postcode	3685 non-null	object
10	FSM_Funding	3685 non-null	int64
11	Pupil_Premium_Funding	3685 non-null	int64
12	Pupil_Premium_Pupils	3685 non-null	int64
13	School_Led_Tutoring_Funding	3681 non-null	float64
14	Total_Funding	3685 non-null	float64
15	Group_UID	2893 non-null	float64
16	Group_ID	2893 non-null	object
17	School_Name	2893 non-null	object
18	Trust_Name	2893 non-null	object
19	Attainment8	3685 non-null	float64
20	Progress8	3682 non-null	float64
21	Percent_Disadvantaged_2022	3685 non-null	int64
22	Percent_Not_Disadvantaged_2022	3685 non-null	int64
23	Percent_Disadvantaged_Strong_Passes	3636 non-null	float64
24	Percent_Not_Disadvantaged_Strong_Passes	3639 non-null	float64
25	Average Attainment 8 score per non-disadvantaged pupil - 2022	3642 non-null	float64
26	Progress8_NonDisadvantaged_2022	3640 non-null	float64
27	Attainment8_Disadvantaged_2022	3640 non-null	float64

28	Progress8_Disadvantaged_2022	3629	non-null	float64
29	Progress8_Maths_Disadvantaged	3624	non-null	float64
30	Progress8_English_Disadvantaged	3624	non-null	float64
31	Progress8_Maths_NonDisadvantaged	3632	non-null	float64
32	Progress8_English_NonDisadvantaged	3632	non-null	float64
33	trust_name	1253	non-null	object
34	Trust_UID	1253	non-null	float64
35	Trust_ID	1253	non-null	object
36	Num_Academies_Performance	1253	non-null	float64
37	Num_Pupils_KS4_Performance	1253	non-null	float64
38	Avg_Attainment8_KS4_Weighted	1253	non-null	float64
39	Progress8_Adjusted_Weighted	1253	non-null	float64
40	nan	1253	non-null	float64
41	Index of Multiple Deprivation Decile	3682	non-null	float64
42	LSOA Name	3682	non-null	object
43	POSTCODE	3685	non-null	object

dtypes: float64(22), int64(7), object(15)
memory usage: 1.2+ MB

Feature Engineering - Gaps

A number of features are needed as part of the analysis which include gaps between disadvantaged and advantaged pupils, and pupil premium funding per pupil. These will therefore be feature engineered

```
# Gap between progress 8 scores
merged_df['progress8_gap'] = merged_df['Progress8_NonDisadvantaged_2022']-merged_df['Progress8_Disadvantaged_2022']

#Gap between attainment 8
merged_df['attainment8_gap'] = merged_df['Average Attainment 8 score per non-disadvantaged pupil']-merged_df['Average Attainment 8 score per disadvantaged pupil']

#Gap between maths scores
merged_df['maths_gap'] = merged_df['Progress8_Maths_NonDisadvantaged']- merged_df['Progress8_Maths_Disadvantaged']

#Gap between English scores
merged_df['english_gap'] = merged_df['Progress8_English_NonDisadvantaged']- merged_df['Progress8_English_Disadvantaged']

#Gap between 5 GCSE strong pass percentages
merged_df['5_GCSE_gap'] = merged_df['Percent_Not_Disadvantaged_Strong_Passes'] - merged_df['Percent_Disadvantaged_Strong_Passes']
```

```
#Pupil premium per pupil calculation
merged_df['pupilpremium_per_pupil'] = merged_df['Pupil_Premium_Funding'] / merged_df['Pupil_
```

Another check for NaN values and duplicates given earlier data wrangling work

```
merged_df.isna().sum()
```

URN	0
Local_Authority_Name	0
Local_Authority_Number	0
School_Type	0
School_College_Type	0
Religious_Character	1332
Admissions_Policy	245
School_Gender	0
Ofsted_Rating	32
School_postcode	0
FSM_Funding	0
Pupil_Premium_Funding	0
Pupil_Premium_Pupils	0
School_Led_Tutoring_Funding	4
Total_Funding	0
Group_UID	792
Group_ID	792
School_Name	792
Trust_Name	792
Attainment8	0
Progress8	3
Percent_Disadvantaged_2022	0
Percent_Not_Disadvantaged_2022	0
Percent_Disadvantaged_Strong_Passes	49
Percent_Not_Disadvantaged_Strong_Passes	46
Average Attainment 8 score per non-disadvantaged pupil - 2022	43
Progress8_NonDisadvantaged_2022	45
Attainment8_Disadvantaged_2022	45
Progress8_Disadvantaged_2022	56
Progress8_Maths_Disadvantaged	61
Progress8_English_Disadvantaged	61
Progress8_Maths_NonDisadvantaged	53
Progress8_English_NonDisadvantaged	53
trust_name	2432

Trust_UID	2432
Trust_ID	2432
Num_Academies_Performance	2432
Num_Pupils_KS4_Performance	2432
Avg_Attainment8_KS4_Weighted	2432
Progress8_Adjusted_Weighted	2432
NaN	2432
Index of Multiple Deprivation Decile	3
LSOA Name	3
POSTCODE	0
progress8_gap	57
attainment8_gap	45
maths_gap	65
english_gap	65
5_GCSE_gap	49
pupilpremium_per_pupil	0
dtype: int64	

Not all NaN rows need to be dropped. Essential ones are URN and 'Progress8_Maths_Disadvantaged', 'Index of Multiple Deprivation Decile', 'Progress8_Maths_NonDisadvantaged', 'Progress8_Disadvantaged_2022', 'Progress8_Maths_Advantaged' which will impact analysis.

```
merged_df.isna().sum()
merged_df = merged_df.dropna(subset=['Progress8_Maths_Disadvantaged', 'Index of Multiple Deprivation Decile', 'Progress8_Maths_Advantaged'])
merged_df.isna().sum()
```

URN	0
Local_Authority_Name	0
Local_Authority_Number	0
School_Type	0
School_College_Type	0
Religious_Character	1126
Admissions_Policy	235
School_Gender	0
Ofsted_Rating	32
School_postcode	0
FSM_Funding	0
Pupil_Premium_Funding	0
Pupil_Premium_Pupils	0
School_Led_Tutoring_Funding	4
Total_Funding	0
Group_UID	0

Group_ID	0
School_Name	0
Trust_Name	0
Attainment8	0
Progress8	0
Percent_Disadvantaged_2022	0
Percent_Not_Disadvantaged_2022	0
Percent_Disadvantaged_Strong_Passes	0
Percent_Not_Disadvantaged_Strong_Passes	0
Average Attainment 8 score per non-disadvantaged pupil - 2022	0
Progress8_NonDisadvantaged_2022	0
Attainment8_Disadvantaged_2022	0
Progress8_Disadvantaged_2022	0
Progress8_Maths_Disadvantaged	0
Progress8_English_Disadvantaged	0
Progress8_Maths_NonDisadvantaged	0
Progress8_English_NonDisadvantaged	0
trust_name	1579
Trust_UID	1579
Trust_ID	1579
Num_Academies_Performance	1579
Num_Pupils_KS4_Performance	1579
Avg_Attainment8_KS4_Weighted	1579
Progress8_Adjusted_Weighted	1579
NaN	1579
Index of Multiple Deprivation Decile	0
LSOA Name	0
POSTCODE	0
progress8_gap	0
attainment8_gap	0
maths_gap	0
english_gap	0
5_GCSE_gap	0
pupilpremium_per_pupil	0
dtype: int64	

Remaining NaN values:

- Religious_Character 1126
- Admissions_Policy 235
- Ofsted_Rating 32

These cant be filled with a mean, median, mode or 0, and as they are categorical variables,

I am not too concerned for now. To check data integrity, I will need to ensure, each row for each school as a unique URN

Also check and remove duplicates based on URN so each school has only 1 row

```
duplicate_urns = merged_df[merged_df.duplicated('URN', keep=False)]
#keep= False will mark all duplicates as True regardless of position
duplicate_urns.info()
```

```
<class 'pandas.core.frame.DataFrame'>
```

```
Index: 644 entries, 754 to 3664
```

```
Data columns (total 50 columns):
```

#	Column	Non-Null Count	Dtype
0	URN	644 non-null	int64
1	Local_Authority_Name	644 non-null	object
2	Local_Authority_Number	644 non-null	int64
3	School_Type	644 non-null	object
4	School_College_Type	644 non-null	object
5	Religious_Character	378 non-null	object
6	Admissions_Policy	575 non-null	object
7	School_Gender	644 non-null	object
8	Ofsted_Rating	638 non-null	object
9	School_postcode	644 non-null	object
10	FSM_Funding	644 non-null	int64
11	Pupil_Premium_Funding	644 non-null	int64
12	Pupil_Premium_Pupils	644 non-null	int64
13	School_Led_Tutoring_Funding	640 non-null	float64
14	Total_Funding	644 non-null	float64
15	Group_UID	644 non-null	float64
16	Group_ID	644 non-null	object
17	School_Name	644 non-null	object
18	Trust_Name	644 non-null	object
19	Attainment8	644 non-null	float64
20	Progress8	644 non-null	float64
21	Percent_Disadvantaged_2022	644 non-null	int64
22	Percent_Not_Disadvantaged_2022	644 non-null	int64
23	Percent_Disadvantaged_Strong_Passes	644 non-null	float64
24	Percent_Not_Disadvantaged_Strong_Passes	644 non-null	float64
25	Average Attainment 8 score per non-disadvantaged pupil - 2022	644 non-null	float64
26	Progress8_NonDisadvantaged_2022	644 non-null	float64
27	Attainment8_Disadvantaged_2022	644 non-null	float64
28	Progress8_Disadvantaged_2022	644 non-null	float64

29	Progress8_Maths_Disadvantaged	644 non-null	float64
30	Progress8_English_Disadvantaged	644 non-null	float64
31	Progress8_Maths_NonDisadvantaged	644 non-null	float64
32	Progress8_English_NonDisadvantaged	644 non-null	float64
33	trust_name	316 non-null	object
34	Trust_UID	316 non-null	float64
35	Trust_ID	316 non-null	object
36	Num_Academies_Performance	316 non-null	float64
37	Num_Pupils_KS4_Performance	316 non-null	float64
38	Avg_Attainment8_KS4_Weighted	316 non-null	float64
39	Progress8_Adjusted_Weighted	316 non-null	float64
40	nan	316 non-null	float64
41	Index of Multiple Deprivation Decile	644 non-null	float64
42	LSOA Name	644 non-null	object
43	POSTCODE	644 non-null	object
44	progress8_gap	644 non-null	float64
45	attainment8_gap	644 non-null	float64
46	maths_gap	644 non-null	float64
47	english_gap	644 non-null	float64
48	5_GCSE_gap	644 non-null	float64
49	pupilpremium_per_pupil	644 non-null	float64

dtypes: float64(28), int64(7), object(15)

memory usage: 256.6+ KB

```
#drop duplicates
merged_df = merged_df.drop_duplicates(subset='URN')
duplicate_urns = merged_df[merged_df.duplicated('URN', keep=False)]
#keep= False will mark all duplicates as True regardless of position
duplicate_urns.info()
```

```
<class 'pandas.core.frame.DataFrame'>
```

```
Index: 0 entries
```

```
Data columns (total 50 columns):
```

#	Column	Non-Null Count	Dtype
0	URN	0 non-null	int64
1	Local_Authority_Name	0 non-null	object
2	Local_Authority_Number	0 non-null	int64
3	School_Type	0 non-null	object
4	School_College_Type	0 non-null	object
5	Religious_Character	0 non-null	object
6	Admissions_Policy	0 non-null	object

7	School_Gender	0 non-null	object
8	Ofsted_Rating	0 non-null	object
9	School_postcode	0 non-null	object
10	FSM_Funding	0 non-null	int64
11	Pupil_Premium_Funding	0 non-null	int64
12	Pupil_Premium_Pupils	0 non-null	int64
13	School_Led_Tutoring_Funding	0 non-null	float64
14	Total_Funding	0 non-null	float64
15	Group_UID	0 non-null	float64
16	Group_ID	0 non-null	object
17	School_Name	0 non-null	object
18	Trust_Name	0 non-null	object
19	Attainment8	0 non-null	float64
20	Progress8	0 non-null	float64
21	Percent_Disadvantaged_2022	0 non-null	int64
22	Percent_Not_Disadvantaged_2022	0 non-null	int64
23	Percent_Disadvantaged_Strong_Passes	0 non-null	float64
24	Percent_Not_Disadvantaged_Strong_Passes	0 non-null	float64
25	Average Attainment 8 score per non-disadvantaged pupil - 2022	0 non-null	float64
26	Progress8_NonDisadvantaged_2022	0 non-null	float64
27	Attainment8_Disadvantaged_2022	0 non-null	float64
28	Progress8_Disadvantaged_2022	0 non-null	float64
29	Progress8_Maths_Disadvantaged	0 non-null	float64
30	Progress8_English_Disadvantaged	0 non-null	float64
31	Progress8_Maths_NonDisadvantaged	0 non-null	float64
32	Progress8_English_NonDisadvantaged	0 non-null	float64
33	trust_name	0 non-null	object
34	Trust_UID	0 non-null	float64
35	Trust_ID	0 non-null	object
36	Num_Academies_Performance	0 non-null	float64
37	Num_Pupils_KS4_Performance	0 non-null	float64
38	Avg_Attainment8_KS4_Weighted	0 non-null	float64
39	Progress8_Adjusted_Weighted	0 non-null	float64
40	nan	0 non-null	float64
41	Index of Multiple Deprivation Decile	0 non-null	float64
42	LSOA Name	0 non-null	object
43	POSTCODE	0 non-null	object
44	progress8_gap	0 non-null	float64
45	attainment8_gap	0 non-null	float64
46	maths_gap	0 non-null	float64
47	english_gap	0 non-null	float64
48	5_GCSE_gap	0 non-null	float64
49	pupilpremium_per_pupil	0 non-null	float64

```
dtypes: float64(28), int64(7), object(15)
memory usage: 0.0+ bytes
```

Descriptive Statistics

Before delving into investigating the merged_df data, I will conduct some basic descriptive statistics to get a feel of the distribution and spread of the data

```
#distribution of deprivation decile
merged_df['Index of Multiple Deprivation Decile'].value_counts()
```

```
Index of Multiple Deprivation Decile
```

```
9.0      275
4.0      258
3.0      255
2.0      253
7.0      249
10.0     248
8.0      246
5.0      245
6.0      236
1.0      204
```

```
Name: count, dtype: int64
```

```
## Descriptive Statistics
descriptive_stats = merged_df[['Progress8', 'Attainment8',
                               'Total_Funding', 'Pupil_Premium_Funding', 'School_Led_Tutoring_Funding']]
print(descriptive_stats)
```

	Progress8	Attainment8	Total_Funding	Pupil_Premium_Funding \
count	2469.000000	2469.000000	2.469000e+03	2.469000e+03
mean	-0.033925	46.391009	6.360708e+06	2.538846e+05
std	0.516867	8.851594	2.156262e+06	1.533941e+05
min	-2.160000	12.000000	4.100040e+05	1.379000e+04
25%	-0.360000	40.700000	4.946905e+06	1.408550e+05
50%	-0.030000	45.600000	6.317928e+06	2.260580e+05
75%	0.310000	51.100000	7.629306e+06	3.329300e+05
max	2.370000	86.400000	1.770485e+07	1.098175e+06

	School_Led_Tutoring_Funding	Index of Multiple Deprivation Decile
count	2467.000000	2469.000000

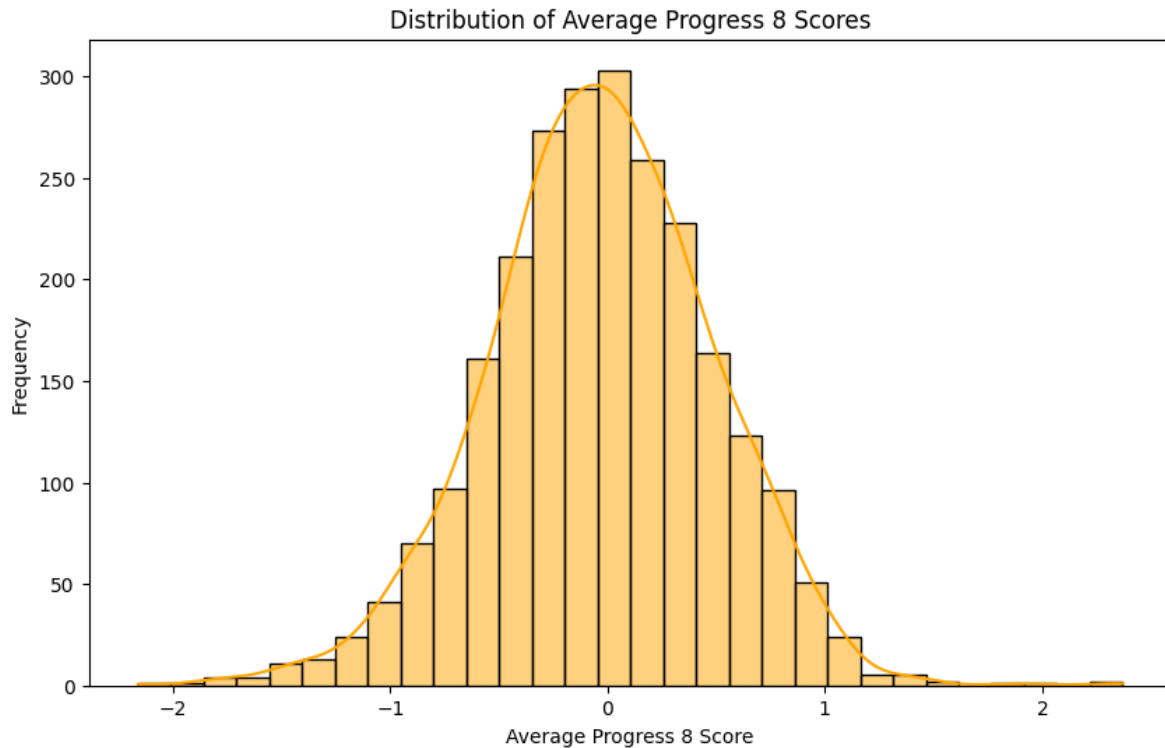
mean	43365.633563	5.594978
std	25025.220705	2.846063
min	2430.000000	1.000000
25%	24916.500000	3.000000
50%	39042.000000	6.000000
75%	56587.500000	8.000000
max	162450.000000	10.000000

View the distribution of progress 8 scores nationally

```
# Histogram for Progress 8 scores
plt.figure(figsize=(10, 6))
sns.histplot(merged_df['Progress8'], bins=30, kde=True, color='orange')
plt.title('Distribution of Average Progress 8 Scores')
plt.xlabel('Average Progress 8 Score')
plt.ylabel('Frequency')

images_dir = 'images'
image_path = os.path.join(images_dir, 'Obj1_progress8_distribution_nationally.png' )
plt.savefig(image_path)

plt.show()
```



Objective 1 Code: Analysing Gaps Between Disadvantaged and Advantaged Pupils

I'll start by finding the mean and gaps of key performance indicators

```
# Calculate mean scores
mean_scores = {
    'Attainment 8 Disadvantaged': merged_df['Attainment8_Disadvantaged_2022'].mean(),
    'Attainment 8 Non-Disadvantaged': merged_df['Average Attainment 8 score per non-disadvantaged_2022'].mean(),
    'Progress 8 Disadvantaged': merged_df['Progress8_Disadvantaged_2022'].mean(),
    'Progress 8 Non-Disadvantaged': merged_df['Progress8_NonDisadvantaged_2022'].mean(),
    'Maths Disadvantaged': merged_df['Progress8_Maths_Disadvantaged'].mean(),
    'Maths Non-Disadvantaged': merged_df['Progress8_Maths_NonDisadvantaged'].mean(),
    'English Disadvantaged': merged_df['Progress8_English_Disadvantaged'].mean(),
    'English Non-Disadvantaged': merged_df['Progress8_English_NonDisadvantaged'].mean(),
    'Percentage Disadvantaged EngMaths_95': merged_df['Percent_Disadvantaged_Strong_Passes'].mean(),
    'Percentage Nondisadv Student EngMaths_95': merged_df['Percent_Not_Disadvantaged_Strong_Passes'].mean()
}

gaps = {
```

```

    'Attainment 8 Gap': merged_df['attainment8_gap'].mean(),
    'Progress 8 Gap': merged_df['progress8_gap'].mean(),
    'Maths Gap': merged_df['maths_gap'].mean(),
    'English Gap': merged_df['english_gap'].mean(),
    'percentage_95': merged_df['5_GCSE_gap'].mean()
}

print("\nMean Scores:")
for key, value in mean_scores.items():
    print(f"{key}: {value:.2f}") #round to 2 decimal places

print("\nGaps Between Groups:")
for key, value in gaps.items():
    print(f"{key}: {value:.2f}") #round to 2 decimal places

```

Mean Scores:

```

Attainment 8 Disadvantaged: 40.22
Attainment 8 Non-Disadvantaged: 51.83
Progress 8 Disadvantaged: -0.47
Progress 8 Non-Disadvantaged: 0.13
Maths Disadvantaged: -0.44
Maths Non-Disadvantaged: 0.11
English Disadvantaged: -0.46
English Non-Disadvantaged: 0.12
Percentage Disadvantaged EngMaths_95: 28.09
Percentage Nondisadv Student EngMaths_95: 50.01

```

Gaps Between Groups:

```

Attainment 8 Gap: 11.61
Progress 8 Gap: 0.60
Maths Gap: 0.55
English Gap: 0.58
percentage_95: 21.92

```

So as to avoid repetition, I will interpret the results when discussing objectives later in this notebook. For now, it is worth noting, all the gaps are positive, suggesting disadvantaged pupils are on average underperforming in every area compared to non-disadvantaged pupils

```

# DataFrames for Progress 8 and Attainment 8

#style
sns.set(style="whitegrid")

# Progress 8 Performance Data
progress8_data = merged_df[['Progress8_Disadvantaged_2022', 'Progress8_NonDisadvantaged_2022']]
progress8_melted = progress8_data.melt(var_name='Group', value_name='Progress 8 Score')
progress8_melted['Group'] = progress8_melted['Group'].map({
    'Progress8_Disadvantaged_2022': 'Disadvantaged',
    'Progress8_NonDisadvantaged_2022': 'Non-Disadvantaged'
})

# Attainment 8 Performance Data
attainment8_data = merged_df[['Attainment8_Disadvantaged_2022', 'Average Attainment 8 score per non-disadvantaged pupil - 2022']]
attainment8_melted = attainment8_data.melt(var_name='Group', value_name='Attainment 8 Score')
attainment8_melted['Group'] = attainment8_melted['Group'].map({
    'Attainment8_Disadvantaged_2022': 'Disadvantaged',
    'Average Attainment 8 score per non-disadvantaged pupil - 2022': 'Non-Disadvantaged'
})

fig, axes = plt.subplots(1, 2, figsize=(16, 8))

# Box Plot for Progress 8 Scores
sns.boxplot(
    x='Group',
    y='Progress 8 Score',
    data=progress8_melted,
    palette="Set1",
    ax=axes[0]
)
axes[0].set_title('Progress 8 Scores by Group', fontsize=16)
axes[0].set_xlabel('Group', fontsize=14)
axes[0].set_ylabel('Progress 8 Score', fontsize=14)

# Box Plot for Attainment 8 Scores
sns.boxplot(
    x='Group',
    y='Attainment 8 Score',
    data=attainment8_melted,
    palette="Set2",

```

```

    ax=axes[1]
)
axes[1].set_title('Attainment 8 Scores by Group', fontsize=16)
axes[1].set_xlabel('Group', fontsize=14)
axes[1].set_ylabel('Attainment 8 Score', fontsize=14)
plt.tight_layout() #adjust plot for better fit

#save the file to the images folder
images_dir = 'images'
image_path = os.path.join(images_dir,'obj1_progress8_attainment8_boxplot.png' )
plt.savefig(image_path)

plt.show()

```

C:\Users\saqib\AppData\Local\Temp\ipykernel_37860\2793708818.py:26: FutureWarning:

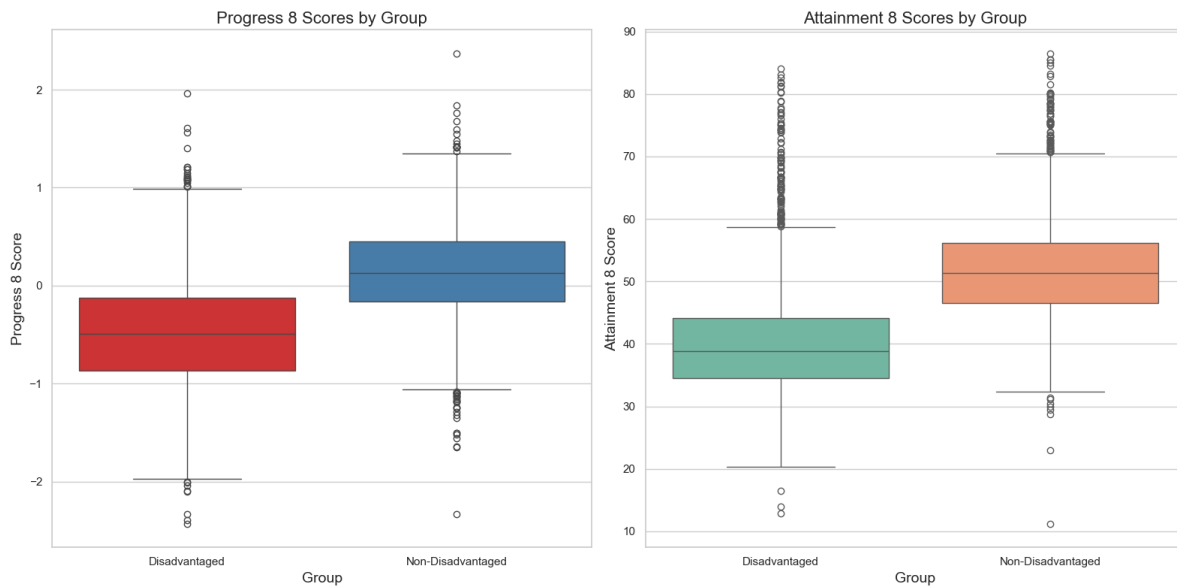
Passing `palette` without assigning `hue` is deprecated and will be removed in v0.14.0. Assign

```
sns.boxplot(
```

C:\Users\saqib\AppData\Local\Temp\ipykernel_37860\2793708818.py:38: FutureWarning:

Passing `palette` without assigning `hue` is deprecated and will be removed in v0.14.0. Assign

```
sns.boxplot(
```



```

# Calculate summary statistics for Attainment 8 Scores
attainment8_grouped = attainment8_melted.groupby('Group')['Attainment 8 Score']

# Calculate statistics for Attainment 8
attainment8_stats = attainment8_grouped.describe()
attainment8_q1 = attainment8_grouped.quantile(0.25)
attainment8_q3 = attainment8_grouped.quantile(0.75)
attainment8_iqr = attainment8_q3 - attainment8_q1
attainment8_range = attainment8_grouped.max() - attainment8_grouped.min()
attainment8_median = attainment8_grouped.median()
attainment8_min = attainment8_grouped.min()
attainment8_max = attainment8_grouped.max()

#
attainment8_summary = pd.DataFrame({
    'Median': attainment8_median,
    'Q1 (25%)': attainment8_q1,
    'Q3 (75%)': attainment8_q3,
    'IQR': attainment8_iqr,
    'Min': attainment8_min,
    'Max': attainment8_max,
    'Range': attainment8_range
})

print("\nAttainment 8 Scores Summary:")
print(attainment8_summary)

# Calculate summary statistics for Progress 8 Scores

progress8_grouped = progress8_melted.groupby('Group')['Progress 8 Score']

# Calculate statistics for Progress 8
progress8_stats = progress8_grouped.describe()
progress8_q1 = progress8_grouped.quantile(0.25)
progress8_q3 = progress8_grouped.quantile(0.75)
progress8_iqr = progress8_q3 - progress8_q1
progress8_range = progress8_grouped.max() - progress8_grouped.min()
progress8_median = progress8_grouped.median()
progress8_min = progress8_grouped.min()
progress8_max = progress8_grouped.max()

```

```

progress8_summary = pd.DataFrame({
    'Median': progress8_median,
    'Q1 (25%)': progress8_q1,
    'Q3 (75%)': progress8_q3,
    'IQR': progress8_iqr,
    'Min': progress8_min,
    'Max': progress8_max,
    'Range': progress8_range
})

print("\nProgress 8 Scores Summary:")
print(progress8_summary)

```

Attainment 8 Scores Summary:

	Median	Q1 (25%)	Q3 (75%)	IQR	Min	Max	Range
Group							
Disadvantaged	38.8	34.5	44.2	9.7	12.9	84.1	71.2
Non-Disadvantaged	51.3	46.6	56.2	9.6	11.2	86.5	75.3

Progress 8 Scores Summary:

	Median	Q1 (25%)	Q3 (75%)	IQR	Min	Max	Range
Group							
Disadvantaged	-0.49	-0.87	-0.12	0.75	-2.43	1.96	4.39
Non-Disadvantaged	0.13	-0.16	0.45	0.61	-2.33	2.37	4.70

```

# Dataframes for Maths and English

# Style
sns.set(style="whitegrid")

# Maths Performance Data
maths_data = merged_df[['Progress8_Maths_Disadvantaged', 'Progress8_Maths_NonDisadvantaged']]
maths_melted = maths_data.melt(var_name='Group', value_name='Maths Score')
maths_melted['Group'] = maths_melted['Group'].map({
    'Progress8_Maths_Disadvantaged': 'Disadvantaged',
    'Progress8_Maths_NonDisadvantaged': 'Non-Disadvantaged'
})

```

```

# English Performance Data
english_data = merged_df[['Progress8_English_Disadvantaged', 'Progress8_English_NonDisadvantaged']]
english_melted = english_data.melt(var_name='Group', value_name='English Score')
english_melted['Group'] = english_melted['Group'].map({
    'Progress8_English_Disadvantaged': 'Disadvantaged',
    'Progress8_English_NonDisadvantaged': 'Non-Disadvantaged'
})

fig, axes = plt.subplots(1, 2, figsize=(16, 8))

# Box Plot for Maths Scores
sns.boxplot(
    x='Group',
    y='Maths Score',
    data=maths_melted,
    palette="Set2",
    ax=axes[0]
)
axes[0].set_title('Maths Scores by Group', fontsize=16)
axes[0].set_xlabel('Group', fontsize=14)
axes[0].set_ylabel('Maths Score', fontsize=14)

# Box Plot for English Scores
sns.boxplot(
    x='Group',
    y='English Score',
    data=english_melted,
    palette="Set3",
    ax=axes[1]
)
axes[1].set_title('English Scores by Group', fontsize=16)
axes[1].set_xlabel('Group', fontsize=14)
axes[1].set_ylabel('English Score', fontsize=14)
plt.tight_layout() # Adjust layout for better fit

# Save the combined box plots as a PNG file in images folder

images_dir = 'images'
image_path = os.path.join(images_dir, 'obj1_maths_english_scores_boxplot.png')
plt.savefig(image_path)

```



```
plt.show()
```

C:\Users\saqib\AppData\Local\Temp\ipykernel_37860\367155346.py:26: FutureWarning:

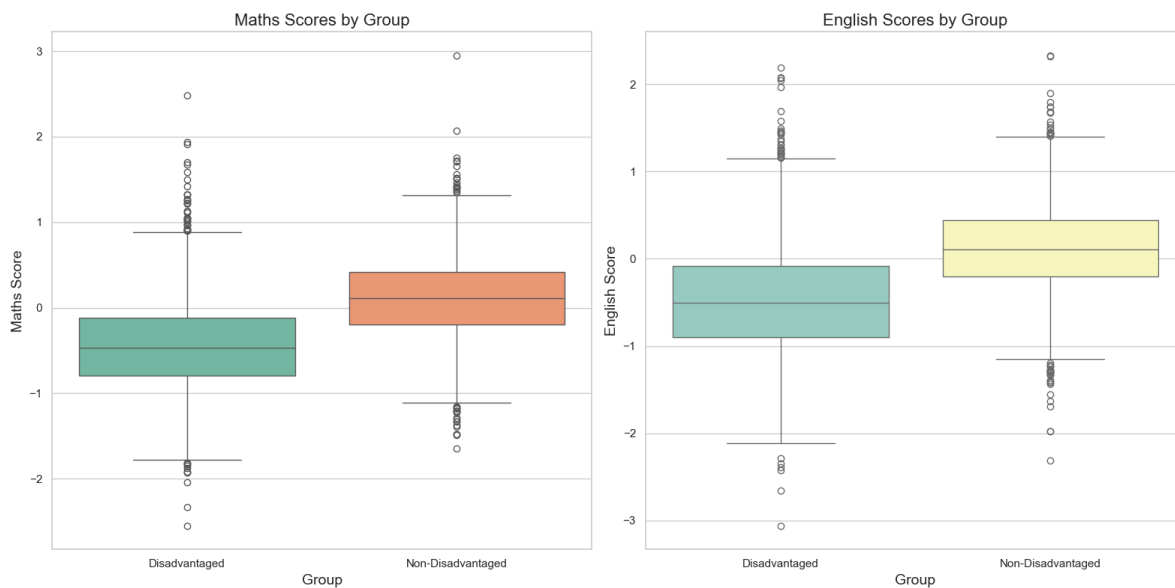
Passing `palette` without assigning `hue` is deprecated and will be removed in v0.14.0. Assign

```
sns.boxplot(
```

C:\Users\saqib\AppData\Local\Temp\ipykernel_37860\367155346.py:39: FutureWarning:

Passing `palette` without assigning `hue` is deprecated and will be removed in v0.14.0. Assign

```
sns.boxplot(
```



```
# Calculate summary statistics for Maths Scores
```

```
maths_grouped = maths_melted.groupby('Group')['Maths Score']
```

```
# Calculate statistics for Maths
```

```
maths_stats = maths_grouped.describe()
```

```
maths_q1 = maths_grouped.quantile(0.25)
```

```
maths_q3 = maths_grouped.quantile(0.75)
```

```
maths_iqr = maths_q3 - maths_q1
```

```

maths_range = maths_grouped.max() - maths_grouped.min()
maths_median = maths_grouped.median()
maths_min = maths_grouped.min()
maths_max = maths_grouped.max()

maths_summary = pd.DataFrame({
    'Median': maths_median,
    'Q1 (25%)': maths_q1,
    'Q3 (75%)': maths_q3,
    'IQR': maths_iqr,
    'Min': maths_min,
    'Max': maths_max,
    'Range': maths_range
})

print("\nMaths Scores Summary:")
print(maths_summary)

# Calculate summary statistics for English Scores

english_grouped = english_melted.groupby('Group')['English Score']

english_stats = english_grouped.describe()
english_q1 = english_grouped.quantile(0.25)
english_q3 = english_grouped.quantile(0.75)
english_iqr = english_q3 - english_q1
english_range = english_grouped.max() - english_grouped.min()
english_median = english_grouped.median()
english_min = english_grouped.min()
english_max = english_grouped.max()

english_summary = pd.DataFrame({
    'Median': english_median,
    'Q1 (25%)': english_q1,
    'Q3 (75%)': english_q3,
    'IQR': english_iqr,
    'Min': english_min,

```

```

    'Max': english_max,
    'Range': english_range
})

```

```

print("\nEnglish Scores Summary:")
print(english_summary)

```

Maths Scores Summary:

	Median	Q1 (25%)	Q3 (75%)	IQR	Min	Max	Range
Group							
Disadvantaged	-0.47	-0.79	-0.12	0.67	-2.55	2.48	5.03
Non-Disadvantaged	0.11	-0.20	0.42	0.62	-1.65	2.95	4.60

English Scores Summary:

	Median	Q1 (25%)	Q3 (75%)	IQR	Min	Max	Range
Group							
Disadvantaged	-0.50	-0.9	-0.08	0.82	-3.06	2.19	5.25
Non-Disadvantaged	0.11	-0.2	0.44	0.64	-2.31	2.33	4.64

Objective 2 Code: Identify and Analyse Outlier Schools in Positive Progress 8 of Disadvantaged Pupils

For simplicity, I have chosen to use the Interquartile Range approach to identify outliers, rather than Z-score. This also allows for easier visualisation using a boxplot. I will begin by establishing quartiles for a box plot to see the distribution of progress-8 disadvantaged students and then determine outliers using standard approach of interquartile range. As I am interested in high performing schools, I will only take the positive outlier schools

```

merged_df_2= merged_df.copy() # copy of merged_df is used for data integrity
merged_df_2.head()

```

URN	Local_Authority_Name	Local_Authority_Number	School_Type	School_College_Type	Reli
207	105135	Greenwich	203	Academy sponsor led	Academy
718	129342	Solihull	334	Academy sponsor led	Academy
722	130247	Reading	870	Academy sponsor led	Academy
723	130908	Middlesbrough	806	Academy sponsor led	Academy
724	130909	Bradford	380	Academy sponsor led	Academy

```
# Outlier detection of schools in progress 8 performance of disadvantaged pupils

Q1 = merged_df_2['Progress8_Disadvantaged_2022'].quantile(0.25)
Q3 = merged_df_2['Progress8_Disadvantaged_2022'].quantile(0.75)
IQR = Q3 - Q1
lower_bound = Q1 - 1.5 * IQR
upper_bound = Q3 + 1.5 * IQR
# only upper bound is taken as we are interested in high-performing schools
outliers_p8_disadv = merged_df_2[(merged_df_2['Progress8_Disadvantaged_2022'] > upper_bound)]
outliers_p8_disadv[['School_Name', 'Trust_Name', 'Progress8_Disadvantaged_2022']].sort_values(
```

	School_Name	Trust_Name	Progress8_Disadvantaged_2022
2699	Michaela Community School		MICHAELA COMMUNITY SCHOOLS TRUST
3545	St Peter's Catholic School		XAVIER CATHOLIC EDUCATION TRUST
2826	Tauheedul Islam Girls' High School		STAR ACADEMIES
3528	Eden Girls' Leadership Academy, Birmingham		STAR ACADEMIES
2058	St Mark's Catholic School		THE DIOCESE OF WESTMINSTER ACADEMY T
2368	Sacred Heart Catholic School		SACRED HEART CATHOLIC SCHOOL
2830	The Hurlingham Academy		UNITED LEARNING TRUST
2981	Ealing Fields High School		TWYFORD CHURCH OF ENGLAND ACADEMIE
2884	Eden Boys' School, Preston		STAR ACADEMIES
3530	St Francis Xavier School - a Joint Catholic an...		NICHOLAS POSTGATE CATHOLIC ACADEMY T
2950	Bolton Muslim Girls School		PROSPER MULTI ACADEMY TRUST
1475	Birmingham Ormiston Academy		BIRMINGHAM ORMISTON ACADEMY
3158	Dartford Grammar School for Girls		THE ARETÉ TRUST
1120	Lancaster Girls' Grammar School		LANCASTER GIRLS' GRAMMAR SCHOOL
833	Ashcroft Technology Academy		PROSPECT EDUCATION (TECHNOLOGY) TRUS
2162	Ark Bolingbroke Academy		ARK SCHOOLS
1279	Wilson's School		WILSON'S SCHOOL
1919	Featherstone High School		GRAND UNION MULTI ACADEMY TRUST
849	Wren Academy Finchley		WREN ACADEMIES TRUST
1645	Bentley Wood High School		THE BENTLEY WOOD TRUST
2264	Nishkam High School		NISHKAM SCHOOL TRUST

Similarly, I will repeat the process for non-disadvantaged pupils' progress-8 score

```
Q1 = merged_df_2['Progress8_NonDisadvantaged_2022'].quantile(0.25)
Q3 = merged_df_2['Progress8_NonDisadvantaged_2022'].quantile(0.75)
IQR = Q3 - Q1
lower_bound = Q1 - 1.5 * IQR
```

```
upper_bound = Q3 + 1.5 * IQR
outliers_p8_adv = merged_df_2[(merged_df_2['Progress8_NonDisadvantaged_2022'] > upper_bound)]
outliers_p8_adv[['School_Name', 'Trust_Name', 'Progress8_NonDisadvantaged_2022']].sort_values(
```

	School_Name	Trust_Name	Progress8_NonDisadvantaged_2022
2699	Michaela Community School		MICHAELA COMMUNITY SCHOOLS TRUST
3528	Eden Girls' Leadership Academy, Birmingham		STAR ACADEMIES
2826	Tauheedul Islam Girls' High School		STAR ACADEMIES
1748	Hillcrest School and Sixth Form Centre		HILLCREST SCHOOL AND SIXTH FORM CENTRE
2779	Levenshulme High School		EDUCATION AND LEADERSHIP TRUST
815	Ark King Solomon Academy		ARK SCHOOLS
1645	Bentley Wood High School		THE BENTLEY WOOD TRUST
2883	Eden Girls' School, Slough		STAR ACADEMIES
787	Northampton Academy		UNITED LEARNING TRUST
2128	Avonbourne Girls Academy		AVONBOURNE INTERNATIONAL BUSINESS AND
2587	Glenmoor Academy		UNITED LEARNING TRUST
2830	The Hurlingham Academy		UNITED LEARNING TRUST
2722	Eden Girls' School Coventry		STAR ACADEMIES
2981	Ealing Fields High School		TWYFORD CHURCH OF ENGLAND ACADEMIES
3545	St Peter's Catholic School		XAVIER CATHOLIC EDUCATION TRUST

Categorical Variables

To investigate the impact of demographics and socioeconomic influence on outlier schools for Progress 8 - disadvantaged pupils, I will use Religious character, Ofsted Rating, Free School Meal Funding, School_Led_Tutoring_Funding, pupilpremium_per_pupil, Percent_Not_Disadvantaged_2022, Percent_Disadvantaged_2022, Index of Multiple Deprivation Decile

```
demographics_columns = ['Religious_Character', 'Ofsted_Rating', 'FSM_Funding', 'School_Led_Tutoring_Funding', 'Pupil_Premium_Per_Pupil', 'Percent_Not_Disadvantaged_2022', 'Percent_Disadvantaged_2022', 'Index_of_Multiple_Deprivation_Decile']
descriptive_stats_disadv = outliers_p8_disadv[demographics_columns].describe()
descriptive_stats_disadv.info()
```

```
<class 'pandas.core.frame.DataFrame'>
```

```
Index: 8 entries, count to max
```

```
Data columns (total 6 columns):
```

#	Column	Non-Null Count	Dtype
---	-----	-----	-----
0	FSM_Funding	8 non-null	float64

```

1   School_Led_Tutoring_Funding      8 non-null    float64
2   pupilpremium_per_pupil           8 non-null    float64
3   Percent_Not_Disadvantaged_2022    8 non-null    float64
4   Percent_Disadvantaged_2022        8 non-null    float64
5   Index of Multiple Deprivation Decile 8 non-null    float64
dtypes: float64(6)
memory usage: 448.0+ bytes

```

```

descriptive_stats_adv = outliers_p8_adv[demographics_columns].describe()
descriptive_stats_adv

```

	FSM_Funding	School_Led_Tutoring_Funding	pupilpremium_per_pupil	Percent_Not_Disadvantage
count	15.000000	15.000000	15.000000	15.000000
mean	86227.333333	43651.800000	999.404370	66.333333
std	50555.511646	21600.087911	58.855940	14.110111
min	0.000000	12150.000000	977.869565	37.000000
25%	50749.000000	29889.000000	985.000000	57.500000
50%	82107.000000	40662.000000	985.000000	69.000000
75%	122693.500000	52717.500000	985.000000	73.000000
max	182830.000000	86994.000000	1212.046263	95.000000

Analysis may not be conclusive of the above descriptive statistics as some schools maybe in both groups of outliers: progress 8 outliers for disadvantaged pupils and progress 8 outliers for non-disadvantaged pupils. To better understand the differences, we should differentiate between schools which are

- a) only progress 8 outliers for diadvantaged pupils
- b) only for advantaged
- c) those which are outliers for both.

To differentiate the schools, I will select and split based on their URN numbers

```

#URN list for non-disadvantaged outliers
nondisadv_outliers= set(outliers_p8_adv['URN'])
nondisadv_outliers

```

```

{134814,
 135242,
 137178,

```

```
137346,  
138193,  
140008,  
140862,  
140958,  
141196,  
141565,  
141617,  
141970,  
142654,  
147201,  
147430}
```

```
#URN list for disadvantaged outliers
```

```
disadv_outliers = set(outliers_p8_disadv['URN'])  
disadv_outliers
```

```
{135316,  
135507,  
136381,  
136621,  
136944,  
137178,  
137729,  
137995,  
138267,  
138586,  
138960,  
140862,  
141565,  
141617,  
141971,  
142340,  
142654,  
144100,  
147201,  
147213,  
147430}
```

```

# Define outlier sets
only_disadvp8_outliers = disadv_outliers - nondisadv_outliers
only_nondisadvp8_outliers = nondisadv_outliers - disadv_outliers
both_p8_outliers = disadv_outliers & nondisadv_outliers

merged_df_2['Outlier_Category'] = 'None'
merged_df_2.loc[merged_df['URN'].isin(both_p8_outliers), 'Outlier_Category'] = 'Both'
merged_df_2.loc[merged_df['URN'].isin(only_disadvp8_outliers), 'Outlier_Category'] = 'Only_D'
merged_df_2.loc[merged_df['URN'].isin(only_nondisadvp8_outliers), 'Outlier_Category'] = 'Only'

category_counts = merged_df_2['Outlier_Category'].value_counts()
print("Distribution of Outlier Categories:")
print(category_counts)

```

Distribution of Outlier Categories:

Outlier_Category	
None	2440
Only_Disadv	14
Only_NonDisadv	8
Both	7

Name: count, dtype: int64

I will now use the categories to reate an outlier dataframe which can be used for analysing just the progress 8 school outliers against each other

```

outlier_df = merged_df_2[merged_df_2['URN'].isin(both_p8_outliers|only_nondisadvp8_outliers|only_disadvp8_outliers)]

category_counts = outlier_df['Outlier_Category'].value_counts()
print("Distribution of Outlier Categories:")
print(category_counts)

```

Distribution of Outlier Categories:

Outlier_Category	
Only_Disadv	14


```
Only_NonDisadv      8
Both                 7
Name: count, dtype: int64
```

Check for null values

```
merged_df_2.isnull().sum()
```

```
URN                                                    0
Local_Authority_Name                                  0
Local_Authority_Number                                0
School_Type                                            0
School_College_Type                                   0
Religious_Character                                    984
Admissions_Policy                                     197
School_Gender                                          0
Ofsted_Rating                                         29
School_postcode                                       0
FSM_Funding                                            0
Pupil_Premium_Funding                                0
Pupil_Premium_Pupils                                 0
School_Led_Tutoring_Funding                           2
Total_Funding                                          0
Group_UID                                              0
Group_ID                                               0
School_Name                                            0
Trust_Name                                             0
Attainment8                                           0
Progress8                                              0
Percent_Disadvantaged_2022                            0
Percent_Not_Disadvantaged_2022                        0
Percent_Disadvantaged_Strong_Passes                   0
Percent_Not_Disadvantaged_Strong_Passes               0
Average Attainment 8 score per non-disadvantaged pupil - 2022  0
Progress8_NonDisadvantaged_2022                      0
Attainment8_Disadvantaged_2022                       0
Progress8_Disadvantaged_2022                          0
Progress8_Maths_Disadvantaged                        0
Progress8_English_Disadvantaged                      0
Progress8_Maths_NonDisadvantaged                     0
Progress8_English_NonDisadvantaged                   0
trust_name                                             1405
```

```

Trust_UID 1405
Trust_ID 1405
Num_Academies_Performance 1405
Num_Pupils_KS4_Performance 1405
Avg_Attainment8_KS4_Weighted 1405
Progress8_Adjusted_Weighted 1405
NaN 1405
Index of Multiple Deprivation Decile 0
LSOA Name 0
POSTCODE 0
progress8_gap 0
attainment8_gap 0
maths_gap 0
english_gap 0
5_GCSE_gap 0
pupilpremium_per_pupil 0
Outlier_Category 0
dtype: int64

```

View mean Deprivation Index spread across across all categories of schools

```

outlier_performance_index = merged_df_2.groupby('Outlier_Category')[['Index of Multiple Deprivation Decile']]
outlier_performance_index

```

	Outlier_Category	Index of Multiple Deprivation Decile
0	Both	6.285714
1	None	5.603279
2	Only_Disadv	5.000000
3	Only_NonDisadv	3.500000

To analyse various fields of the outlier schools, based on their categories, I will plot some graphs and generate summary statistics

```

#represent deprivation index against outlier category

sns.boxplot(
    x='Outlier_Category',
    y='Index of Multiple Deprivation Decile',
    data=merged_df_2,
    palette='Set3'
)

```

```

)
plt.title('Deprivation Decile by Outlier Category')
plt.xlabel('Outlier Category')
plt.ylabel('Index of Multiple Deprivation Decile')

images_dir = 'images'
image_path = os.path.join(images_dir, 'Obj2_Deprivation_by_Outlier_Category.png')
plt.savefig(image_path)

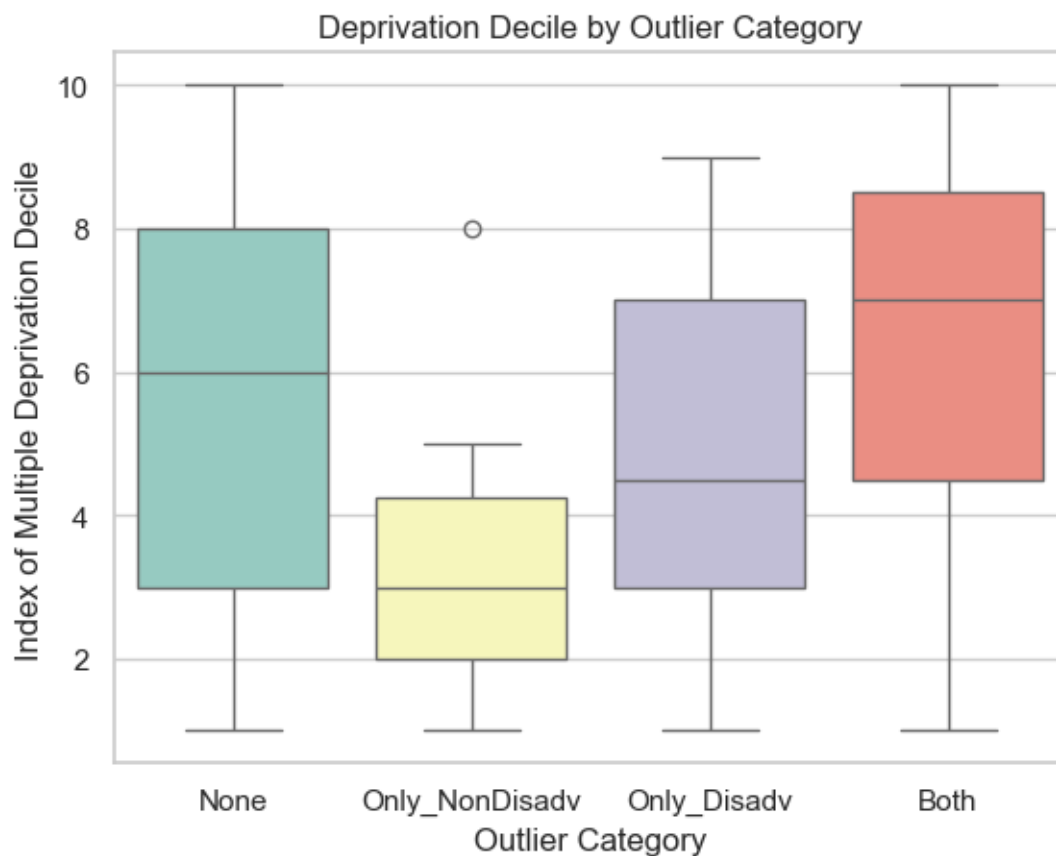
plt.show()

```

C:\Users\saqib\AppData\Local\Temp\ipykernel_37860\1565168097.py:3: FutureWarning:

Passing `palette` without assigning `hue` is deprecated and will be removed in v0.14.0. Assign

```
sns.boxplot(
```



```

# calculate descriptive statistics for outlier categories and index of multiple deprivation

# groupby method used to group merged_df_2 by outlier category, then select IMDC column for
grouped = merged_df_2.groupby('Outlier_Category')['Index of Multiple Deprivation Decile']

median = grouped.median()
q1 = grouped.quantile(0.25)
q3 = grouped.quantile(0.75)
iqr = q3 - q1
minimum = grouped.min()
maximum = grouped.max()
range_ = maximum - minimum

summary = pd.DataFrame({
    'Median': median,
    'Q1 (25%)': q1,
    'Q3 (75%)': q3,
    'IQR': iqr,
    'Min': minimum,
    'Max': maximum,
    'Range': range_
})

#
print("Summary Statistics for 'Index of Multiple Deprivation Decile' by 'Outlier_Category':")
print(summary)

```

Summary Statistics for 'Index of Multiple Deprivation Decile' by 'Outlier_Category':

	Median	Q1 (25%)	Q3 (75%)	IQR	Min	Max	Range
Outlier_Category							
Both	7.0	4.5	8.50	4.00	1.0	10.0	9.0
None	6.0	3.0	8.00	5.00	1.0	10.0	9.0
Only_Disadv	4.5	3.0	7.00	4.00	1.0	9.0	8.0
Only_NonDisadv	3.0	2.0	4.25	2.25	1.0	8.0	7.0

```

outlier_performance = merged_df_2.groupby('Outlier_Category')[['Progress8', 'Progress8_Disadv']]
outlier_performance

```

	Outlier_Category	Progress8	Progress8_Disadvantaged_2022	Progress8_NonDisadvantaged_2022
0	Both	1.614286	1.417143	1.664286
1	None	-0.047791	-0.486266	0.118369
2	Only_Disadv	0.946429	1.100000	0.950714
3	Only_NonDisadv	1.037500	0.626250	1.492500

Plot box plot of progress 8 of disadvantaged pupils by outlier category

```
sns.boxplot(
    x='Outlier_Category',
    y='Progress8_Disadvantaged_2022',
    data= merged_df_2,
    palette='Set3'
)
plt.title('Progress8 of Disadvantaged Pupils by Outlier Category')
plt.xlabel('Outlier Category')
plt.ylabel('Progress8 Disadvantaged 2022')

plt.show()

sns.boxplot(
    x='Outlier_Category',
    y='Progress8_Disadvantaged_2022',
    data= outlier_df,
    palette='Set3'
)
plt.title('Progress8 of Disadvantaged Pupils by Outlier Category')
plt.xlabel('Outlier Category')
plt.ylabel('Progress8 Disadvantaged 2022')

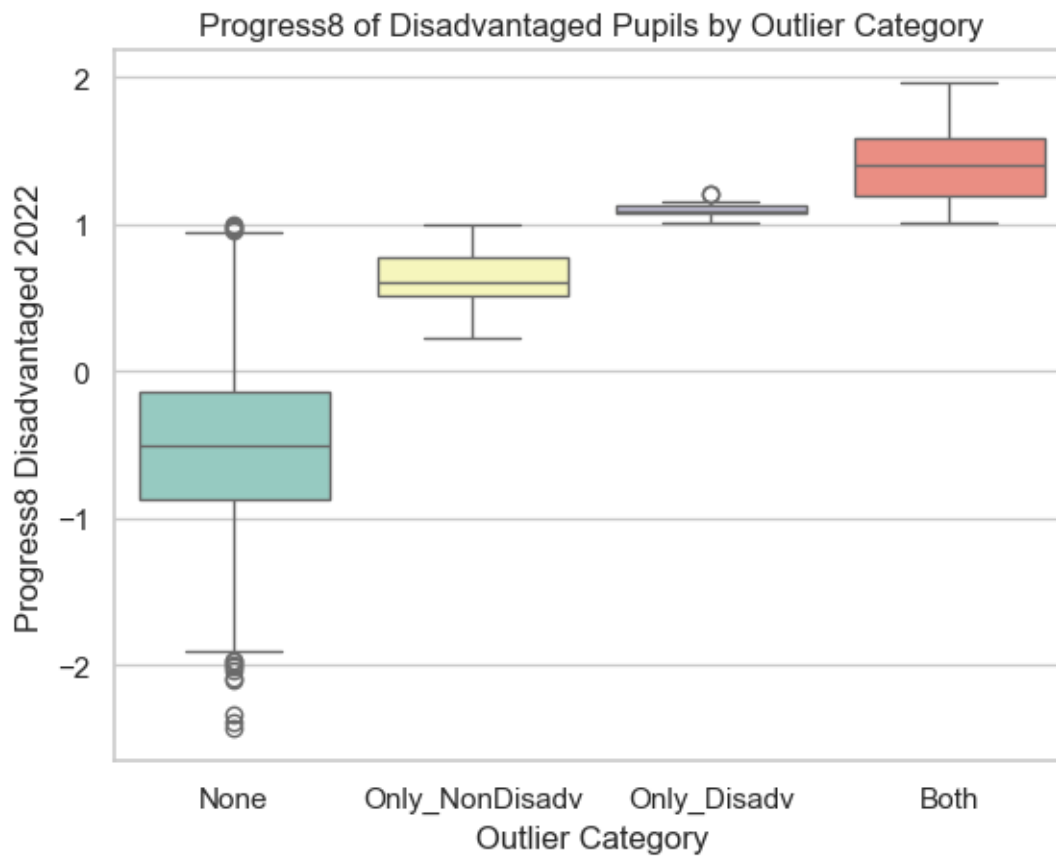
images_dir = 'images'
image_path = os.path.join(images_dir, 'Obj2_Progress8 of Disadvantaged Pupils by Outlier Category')
plt.savefig(image_path)

plt.show()
```

C:\Users\saqib\AppData\Local\Temp\ipykernel_37860\3617201507.py:1: FutureWarning:

Passing `palette` without assigning `hue` is deprecated and will be removed in v0.14.0. Assigning `hue` to the variable is recommended.

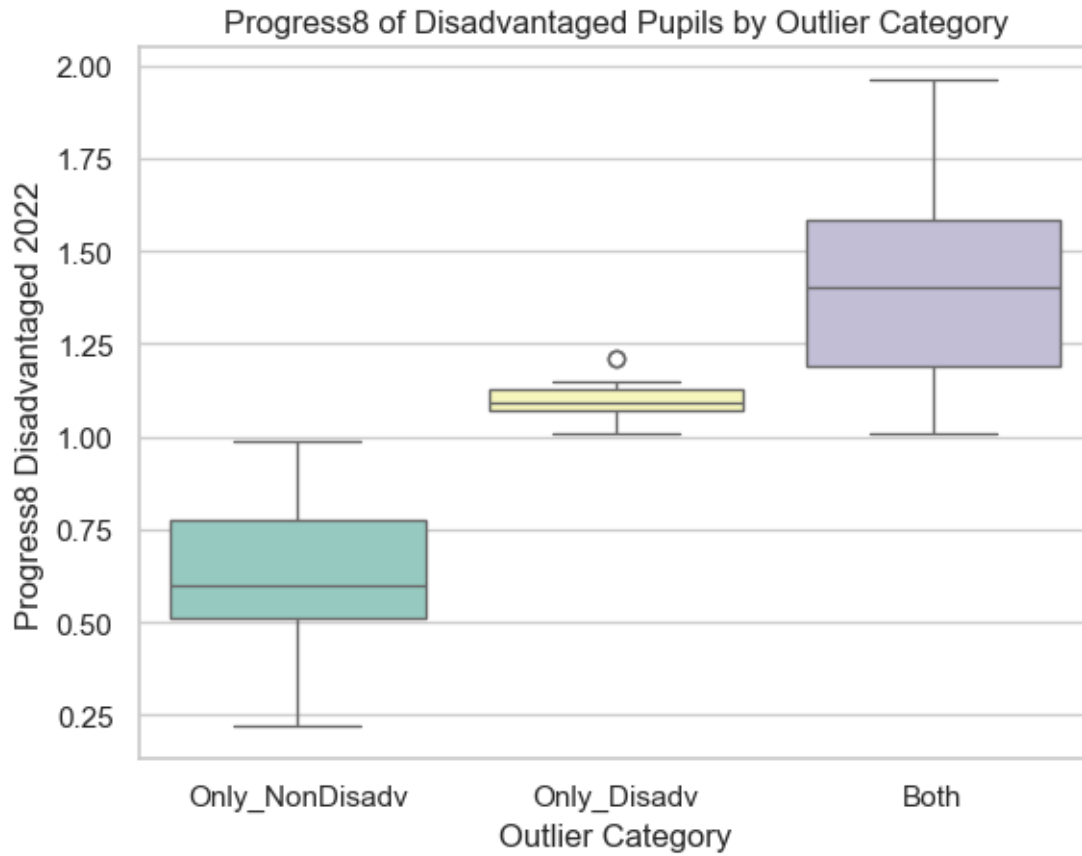
```
sns.boxplot(
```



C:\Users\saqib\AppData\Local\Temp\ipykernel_37860\3617201507.py:14: FutureWarning:

Passing `palette` without assigning `hue` is deprecated and will be removed in v0.14.0. Assign

```
sns.boxplot(
```



```
#calculate summary statistics for the box plot above
grouped = merged_df_2.groupby('Outlier_Category')['Progress8_Disadvantaged_2022']

median = grouped.median()
q1 = grouped.quantile(0.25)
q3 = grouped.quantile(0.75)
iqr = q3 - q1
minimum = grouped.min()
maximum = grouped.max()
range_ = maximum - minimum

summary = pd.DataFrame({
    'Median': median,
    'Q1 (25%)': q1,
    'Q3 (75%)': q3,
```

```

    'IQR': iqr,
    'Min': minimum,
    'Max': maximum,
    'Range': range_
})

print("Summary Statistics for 'Progress8_Disadvantaged_2022' by 'Outlier_Category':")
print(summary)

```

```

Summary Statistics for 'Progress8_Disadvantaged_2022' by 'Outlier_Category':

```

	Median	Q1 (25%)	Q3 (75%)	IQR	Min	Max	Range
Outlier_Category							
Both	1.40	1.19	1.585	0.395	1.01	1.96	0.95
None	-0.50	-0.87	-0.140	0.730	-2.43	0.99	3.42
Only_Disadv	1.09	1.07	1.125	0.055	1.01	1.21	0.20
Only_NonDisadv	0.60	0.51	0.775	0.265	0.22	0.99	0.77

Plot a box plot to show progress 8 scores for all outlier category types

```

sns.boxplot(
    x='Outlier_Category',
    y='Progress8',
    data= merged_df_2,
    palette='Set3'
)

plt.title('Progress8 by Outlier Category')
plt.xlabel('Outlier Category')
plt.ylabel('Progress8 2022')

images_dir = 'images'
image_path = os.path.join(images_dir, 'Obj2_Progress8 by Outlier Category.png')
plt.savefig(image_path)

plt.show()

```

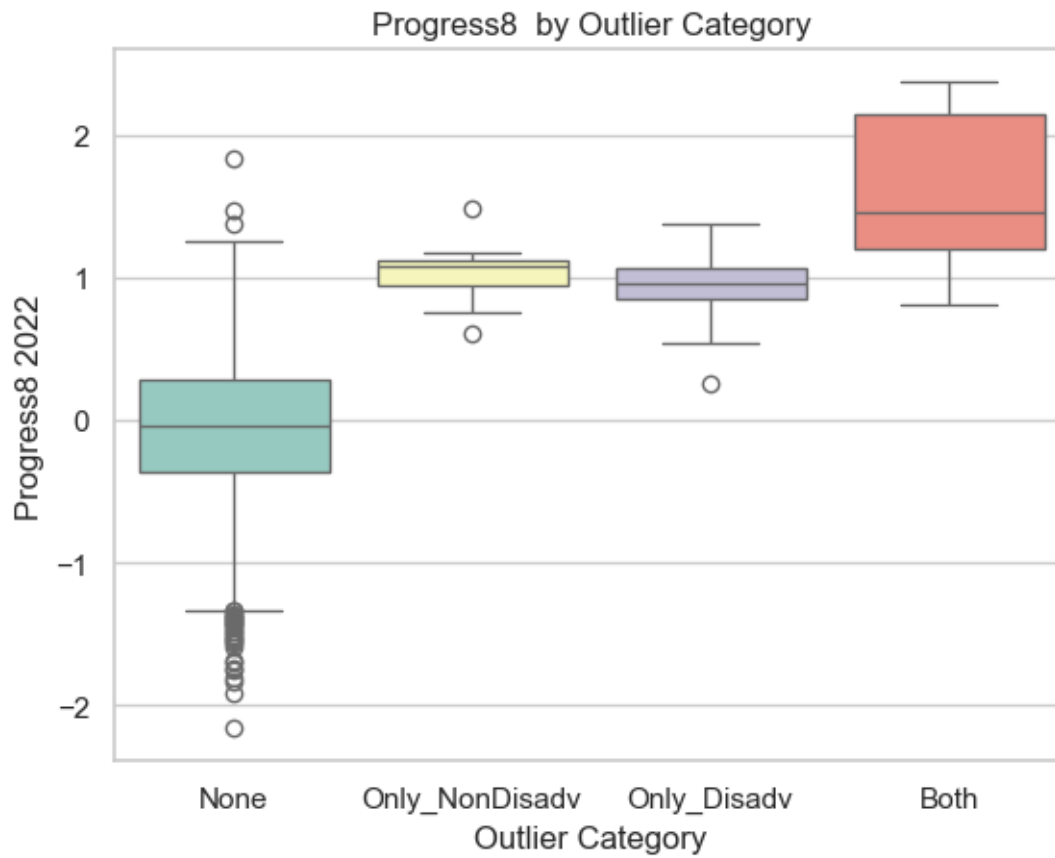
C:\Users\saqib\AppData\Local\Temp\ipykernel_37860\3243744368.py:1: FutureWarning:

Passing `palette` without assigning `hue` is deprecated and will be removed in v0.14.0. Assign

```

sns.boxplot(

```

```
# Calculate summary statistics
grouped = merged_df_2.groupby('Outlier_Category')['Progress8']

median = grouped.median()
q1 = grouped.quantile(0.25)
q3 = grouped.quantile(0.75)
iqr = q3 - q1
minimum = grouped.min()
maximum = grouped.max()
range_ = maximum - minimum

summary = pd.DataFrame({
    'Median': median,
    'Q1 (25%)': q1,
    'Q3 (75%)': q3,
```

```

    'IQR': iqr,
    'Min': minimum,
    'Max': maximum,
    'Range': range_
})

print("Summary Statistics for 'Progress8' by 'Outlier_Category':")
print(summary)

```

```

Summary Statistics for 'Progress8' by 'Outlier_Category':

```

	Median	Q1 (25%)	Q3 (75%)	IQR	Min	Max	Range
Outlier_Category							
Both	1.450	1.1950	2.140	0.9450	0.81	2.37	1.56
None	-0.040	-0.3600	0.290	0.6500	-2.16	1.83	3.99
Only_Disadv	0.955	0.8500	1.070	0.2200	0.25	1.38	1.13
Only_NonDisadv	1.085	0.9375	1.125	0.1875	0.61	1.49	0.88

Using the `get_school_details` function defined earlier, I can extract school details based on a URN list

```

columns = ['School_Name',
            'Trust_Name', 'Percent_Disadvantaged_2022', 'Progress8',
            'Progress8_NonDisadvantaged_2022', 'Progress8_Disadvantaged_2022',
            'Percent_Not_Disadvantaged_2022',
            'Religious_Character',
            'Admissions_Policy',
            'School_Gender',
            'Ofsted_Rating',]

data_loader = DataWrangler(dataframe=outlier_df)
# Schools only in outliers_disadvantaged
schools_only_disdv_outliers = data_loader.get_school_details(only_disadvp8_outliers, columns)
print("schools_only_disdv_outliers:")
print(schools_only_disdv_outliers.to_string(index=False), "\n")

```

DataWrangler initialised with the provided DataFrame.
schools_only_disdv_outliers:

```

School_Name
Ashcroft Technology Academy PROSPECT

```

	Wren Academy Finchley	
	Lancaster Girls' Grammar School	
	Wilson's School	
	Birmingham Ormiston Academy	
	Featherstone High School	
	St Mark's Catholic School	THE
	Ark Bolingbroke Academy	
	Nishkam High School	
	Sacred Heart Catholic School	
	Eden Boys' School, Preston	
	Bolton Muslim Girls School	
	Dartford Grammar School for Girls	
St Francis Xavier School - a Joint Catholic and Church of England Voluntary Academy		NIC

```
# Schools only in outliers_not disadvantaged
data_loader = DataWrangler(dataframe=outlier_df)
schools_only_nondisadv_outliers = data_loader.get_school_details(only_nondisadvp8_outliers, columns)
print("schools_only_nondisadv_outliers:")
print(schools_only_nondisadv_outliers.to_string(index=False), "\n")
```

DataWrangler initialised with the provided DataFrame.

schools_only_nondisadv_outliers:

	School_Name	
	Northampton Academy	UNITED LEARNING
	Ark King Solomon Academy	ARK
Hillcrest School and Sixth Form Centre		HILLCREST SCHOOL AND SIXTH FORM
Avonbourne Girls Academy	AVONBOURNE INTERNATIONAL BUSINESS AND ENTERPRISE ACADEMY	
Glenmoor Academy		UNITED LEARNING
Eden Girls' School Coventry		STAR
Levenshulme High School	EDUCATION AND LEADERSHIP	
Eden Girls' School, Slough		STAR

```
# Schools in both outliers_disadvantaged only and outliers_not disadvantaged

data_loader = DataWrangler(dataframe=outlier_df)

schools_both = data_loader.get_school_details(both_p8_outliers, columns)
print("Schools in both disadv and nondisadv outliers:")
print(schools_both.to_string(index=False))
```

DataWrangler initialised with the provided DataFrame.

Schools in both disadv and nondisadv outliers:

	School_Name	Trust_Name	Perce
	Bentley Wood High School	THE BENTLEY WOOD TRUST	
	Michaela Community School	MICHAELA COMMUNITY SCHOOLS TRUST	
	Tauheedul Islam Girls' High School	STAR ACADEMIES	
	The Hurlingham Academy	UNITED LEARNING TRUST	
	Ealing Fields High School	TWYFORD CHURCH OF ENGLAND ACADEMIES TRUST	
Eden Girls'	Leadership Academy, Birmingham	STAR ACADEMIES	
	St Peter's Catholic School	XAVIER CATHOLIC EDUCATION TRUST	

I will also evaluate the categorical columns in the outlier schools

```
categorical_columns= outlier_df[['School_Type','School_College_Type',
'Religious_Character',
'Admissions_Policy',
'School_Gender',
'Ofsted_Rating','Trust_Name','Outlier_Category']]
```

```
numerical_variables = outlier_df[['FSM_Funding',
'Pupil_Premium_Funding',
'Pupil_Premium_Pupils',
'School_Led_Tutoring_Funding',
'Total_Funding','Attainment8',
'Progress8',
'Percent_Disadvantaged_2022',
'Percent_Not_Disadvantaged_2022',
'Percent_Disadvantaged_Strong_Passes',
'Percent_Not_Disadvantaged_Strong_Passes',
'Average Attainment 8 score per non-disadvantaged pupil - 2022',
'Progress8_NonDisadvantaged_2022',
'Attainment8_Disadvantaged_2022',
'Progress8_Disadvantaged_2022',
'Progress8_Maths_Disadvantaged',
'Progress8_English_Disadvantaged',
'Progress8_Maths_NonDisadvantaged',
'Progress8_English_NonDisadvantaged', 'Index of Multiple L
```

```
numerical_variables.describe()
```

	FSM_Funding	Pupil_Premium_Funding	Pupil_Premium_Pupils	School_Led_Tutoring_Funding	T
count	29.000000	29.000000	29.000000	29.000000	
mean	105475.413793	221438.344828	219.586207	36555.517241	
std	138064.982769	143085.321561	131.774027	21375.221128	
min	0.000000	39400.000000	40.000000	6480.000000	
25%	37515.000000	129035.000000	131.000000	22194.000000	
50%	74260.000000	183210.000000	186.000000	31590.000000	
75%	121260.000000	279740.000000	284.000000	46818.000000	
max	730864.000000	681170.000000	562.000000	86994.000000	

```
#Standardise the numerical values to ensure accurate correlation

scaler = StandardScaler() # use standard scaler which will make each feature have 0 mean and

scaled_data = scaler.fit_transform(numerical_variables)

scaled_numerical_df = pd.DataFrame(scaled_data, columns=numerical_variables.columns)

merged_df_2[scaled_numerical_df.columns] = scaled_numerical_df

scaled_numerical_df.head()
```

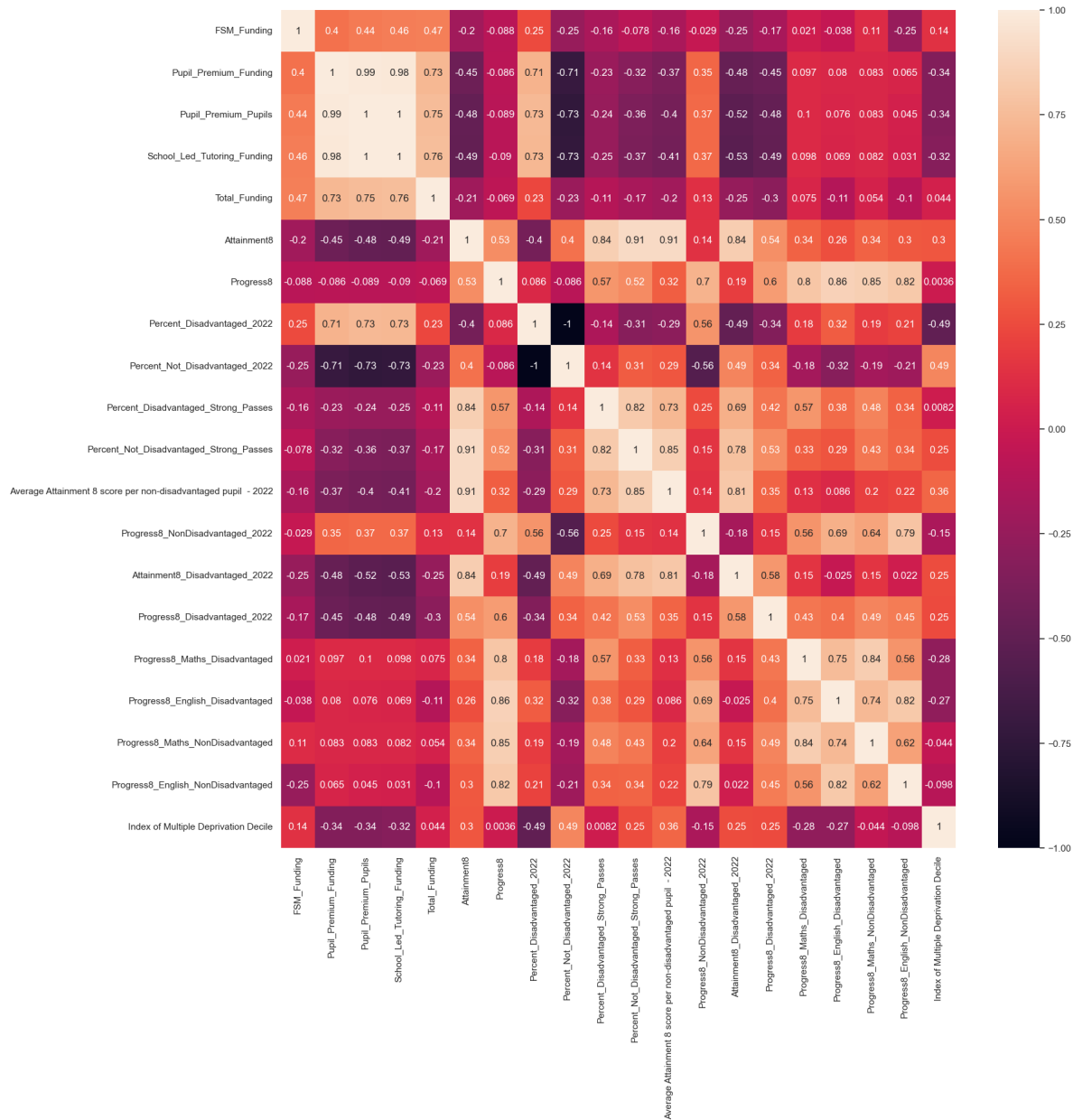
	FSM_Funding	Pupil_Premium_Funding	Pupil_Premium_Pupils	School_Led_Tutoring_Funding	T
0	0.341747	1.458545	1.648210	1.768970	
1	0.199734	3.269861	2.644487	2.401438	
2	4.609846	0.901583	1.038088	1.136931	
3	-0.213244	0.050084	-0.089481	-0.105293	
4	-0.652757	-1.273738	-1.363788	-1.401080	

As part of the analysis, I will create a heatmap of numerical variables

```
# Correlation matrix
corr_matrix = numerical_variables.corr()
plt.figure(figsize=(18,18))
```

```
images_dir = 'images'
image_path = os.path.join(images_dir, 'Obj2_Heatmap Outlier Schools.png')
plt.savefig(image_path)

# Heatmap
sns.heatmap(corr_matrix, annot=True)
plt.show()
```



Objective 3 Code: Identify and evaluate the top performing multi-academy trusts in supporting disadvantaged pupils

```
merged_df_3=merged_df.copy() # make another copy of merged_df
```

```
# group schools by MAT for analysis
MAT_performance = merged_df_3.groupby('Trust_Name')[['Progress8', 'Progress8_Disadvantaged_2022',
                                                    'attainment8_gap',
                                                    'maths_gap',
                                                    'english_gap',
                                                    '5_GCSE_gap',
                                                    'pupilpremium_per_pupil', 'Progress8_NonDisadvantaged_2022']]
MAT_performance.head()
```

	Trust_Name	Progress8	Progress8_Disadvantaged_2022	progress8_gap	attainment8_gap	maths_g
0	5 DIMENSIONS TRUST		-0.13	-0.89		0.930
1	ABBEY ACADEMIES TRUST		-0.43	-1.11		0.930
2	ABBEY COLLEGE, RAMSEY		-0.10	-0.55		0.690
3	ABBEY MULTI ACADEMY TRUST		0.07	-0.05		0.305
4	ABBS CROSS ACADEMY AND ARTS COLLEGE		0.04	-1.17		0.990

```
MAT_performance_sorted = MAT_performance.sort_values(by='Progress8_Disadvantaged_2022', ascending=False)
#we can now sort by progress 8 score of disadvantaged pupils
MAT_performance_sorted.head()
```

	Trust_Name	Progress8	Progress8_Disadvantaged_2022	progress8_gap	attainment8_gap	maths_g
654	MICHAELA COMMUNITY SCHOOLS TRUST	2.37		1.96		0.41
833	SACRED HEART CATHOLIC SCHOOL	1.38		1.21		0.14
780	PROSPER MULTI ACADEMY TRUST	1.01		1.11		-0.13
111	BIRMINGHAM ORMISTON ACADEMY	0.25		1.10		-0.52
587	LANCASTER GIRLS' GRAMMAR SCHOOL	0.54		1.09		-0.41

```
# Group by Trust and calculate mean scores along with the count of schools
MAT_performance = merged_df_3.groupby('Trust_Name').agg(
    avg_progress8_score=('Progress8', 'mean'),
    prog8_score_disadv=('Progress8_Disadvantaged_2022', 'mean'),
    prog8_score_nondisadv=('Progress8_NonDisadvantaged_2022', 'mean'),
    progress8_gap=('progress8_gap', 'mean'),
    attainment8_gap=('attainment8_gap', 'mean'),
    maths_gap=('maths_gap', 'mean'),
    english_gap=('english_gap', 'mean'),
    FiveGCSE_gap=('5_GCSE_gap', 'mean'),
    deprivation_index= ('Index of Multiple Deprivation Decile', 'mean'),
)
```



```

    school_count=('URN', 'count') # Counting the number of schools per Group Name
).reset_index()

# Sort the MAT_performance DataFrame by 'avg_progress8_score' in descending order
MAT_performance_sorted = MAT_performance.sort_values(by='prog8_score_disadv', ascending=False)

MAT_performance_sorted.head()

```

	Trust_Name	avg_progress8_score	prog8_score_disadv	prog8_score_nondisadv	progress8_gap	att
654	MICHAELA COMMUNITY SCHOOLS TRUST		2.37	1.96	2.37	
833	SACRED HEART CATHOLIC SCHOOL		1.38	1.21	1.35	
780	PROSPER MULTI ACADEMY TRUST		1.01	1.11	0.98	
111	BIRMINGHAM ORMISTON ACADEMY		0.25	1.10	0.58	
587	LANCASTER GIRLS' GRAMMAR SCHOOL		0.54	1.09	0.68	

A number of MATs have 1 or 2 schools, so I will filter for those with at least 4 schools as I want to explore organisational impact of Trusts working with mutiple schools

```

# Filter MATs with school_count >= 4
MAT_performance_filtered = MAT_performance[MAT_performance['school_count'] >= 4]

# top 10 MATs with the highest average Progress 8 scores disadvantaged and at least 4 schools
MAT_performance_sorted = MAT_performance_filtered.sort_values(by='prog8_score_disadv', ascending=False)

Top_10MAT = MAT_performance_sorted.head(10)
print(Top_10MAT)

```

	Trust_Name	avg_progress8_score	\
975	STAR ACADEMIES	0.640526	
219	CHILTERN LEARNING TRUST	0.424000	
57	ARK SCHOOLS	0.208421	
1096	THE DIOCESE OF WESTMINSTER ACADEMY TRUST	0.645000	
628	LOXFORD SCHOOL TRUST LIMITED	0.337500	
335	EDUCATION AND LEADERSHIP TRUST	0.260000	
1123	THE GORSE ACADEMIES TRUST	0.435714	
451	HARRIS FEDERATION	0.265652	
827	RUSSELL EDUCATION TRUST	0.466000	
1323	UNITED LEARNING TRUST	0.146757	

	prog8_score_disadv	prog8_score_nondisadv	progress8_gap \
975	0.231579	0.495789	0.264211
219	0.226000	0.634000	0.408000
57	0.085789	0.435789	0.350000
1096	0.076667	0.691667	0.615000
628	0.075000	0.422500	0.347500
335	0.030000	0.762500	0.732500
1123	0.024286	0.645714	0.621429
451	0.020870	0.644348	0.623478
827	-0.068000	0.534000	0.602000
1323	-0.098378	0.487297	0.585676

	attainment8_gap	maths_gap	english_gap	FiveGCSE_gap \
975	6.247368	0.294737	0.244211	10.210526
219	7.680000	0.456000	0.348000	17.000000
57	6.131579	0.437368	0.290526	16.263158
1096	10.700000	0.303333	0.161667	20.833333
628	5.075000	0.312500	0.147500	15.250000
335	10.450000	0.575000	0.527500	18.500000
1123	12.842857	0.675714	0.810000	25.571429
451	10.065217	0.574783	0.400435	19.826087
827	15.380000	0.730000	0.836000	34.000000
1323	9.802703	0.518108	0.507027	18.297297

	deprivation_index	school_count
975	2.421053	19
219	4.600000	5
57	3.157895	19
1096	5.833333	6
628	5.000000	4
335	3.000000	4
1123	5.714286	7
451	4.956522	23
827	5.200000	5
1323	4.297297	37

```
#represent data on a graphs
```

```
# Set the style
sns.set(style="whitegrid")
```

```

# Pconfigure the barplot
plt.figure(figsize=(12, 8))
sns.barplot(
    x='prog8_score_disadv',
    y='Trust_Name',
    data=MAT_performance_sorted.head(10),
    palette='Blues_d'
)
plt.title('Average Progress 8 Scores for disadvantaged pupils of Top 10 MATs')
plt.xlabel('Progress 8 Score - Disadvantaged ')
plt.ylabel('Multi-Academy Trust')
plt.tight_layout()

#save image in data folder
images_dir = 'images'
image_path = os.path.join(images_dir, 'Obj3_Progress8 disadvantaged top 10 MATs.png')
plt.savefig(image_path)

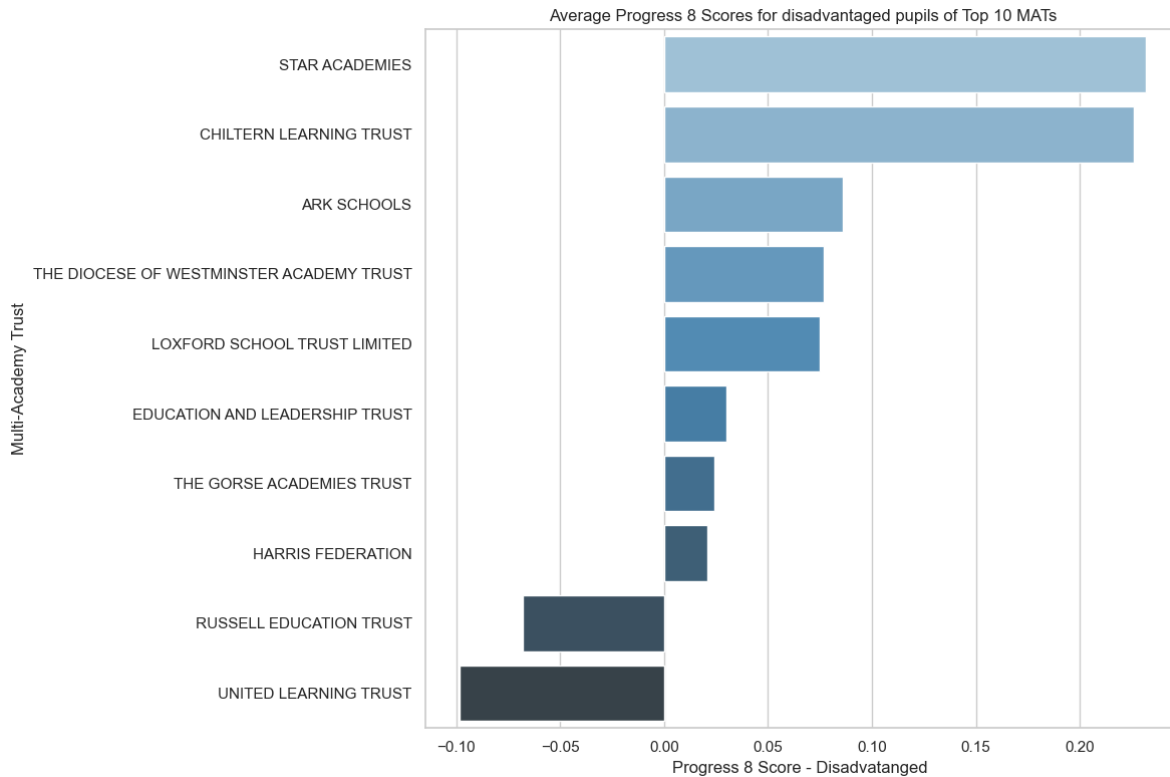
plt.show()

```

C:\Users\saqib\AppData\Local\Temp\ipykernel_37860\2443691115.py:8: FutureWarning:

Passing `palette` without assigning `hue` is deprecated and will be removed in v0.14.0. Assign

```
sns.barplot(
```



```
# Melt the DataFrame for easier plotting
top_10_MATs= MAT_performance_sorted.head(10)
prog8_melted = top_10_MATs.melt(
    id_vars='Trust_Name',
    value_vars=['prog8_score_disadv', 'prog8_score_nondisadv'],
    var_name='Group',
    value_name='Progress8_Score'
)

# Replace group names for clarity
prog8_melted['Group'] = prog8_melted['Group'].map({
    'prog8_score_disadv': 'Disadvantaged',
    'prog8_score_nondisadv': 'Non-Disadvantaged'
})

# Plot
plt.figure(figsize=(14, 8))
sns.barplot(
    x='Trust_Name',
    y='Progress8_Score',
```

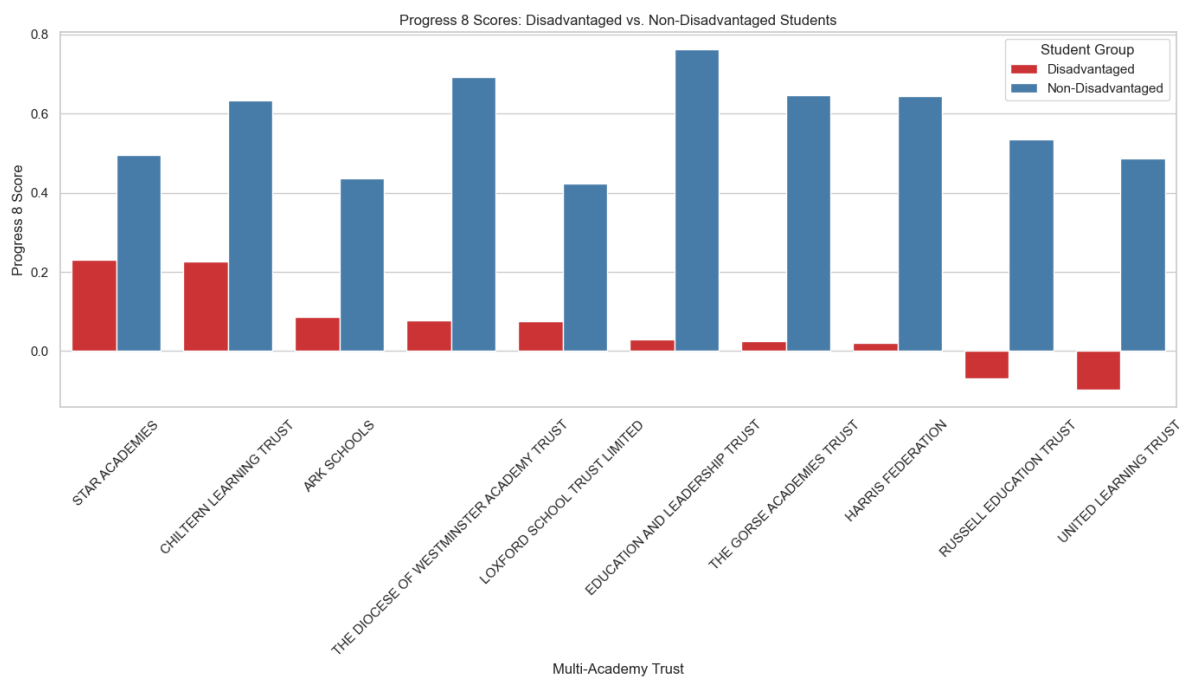
```

    hue='Group',
    data=prog8_melted,
    palette='Set1'
)
plt.title('Progress 8 Scores: Disadvantaged vs. Non-Disadvantaged Students')
plt.xlabel('Multi-Academy Trust')
plt.ylabel('Progress 8 Score')
plt.legend(title='Student Group')
plt.xticks(rotation=45)
plt.tight_layout()

#save the image in data folder
images_dir = 'images'
image_path = os.path.join(images_dir,'Obj3_Progress8 disadv vs advantaged in top 10 MATs.png')
plt.savefig(image_path)

plt.show()

```



Plot a scatterplot of MATs and average progress 8

```

plt.figure(figsize=(10, 6))
sns.scatterplot(

```

```

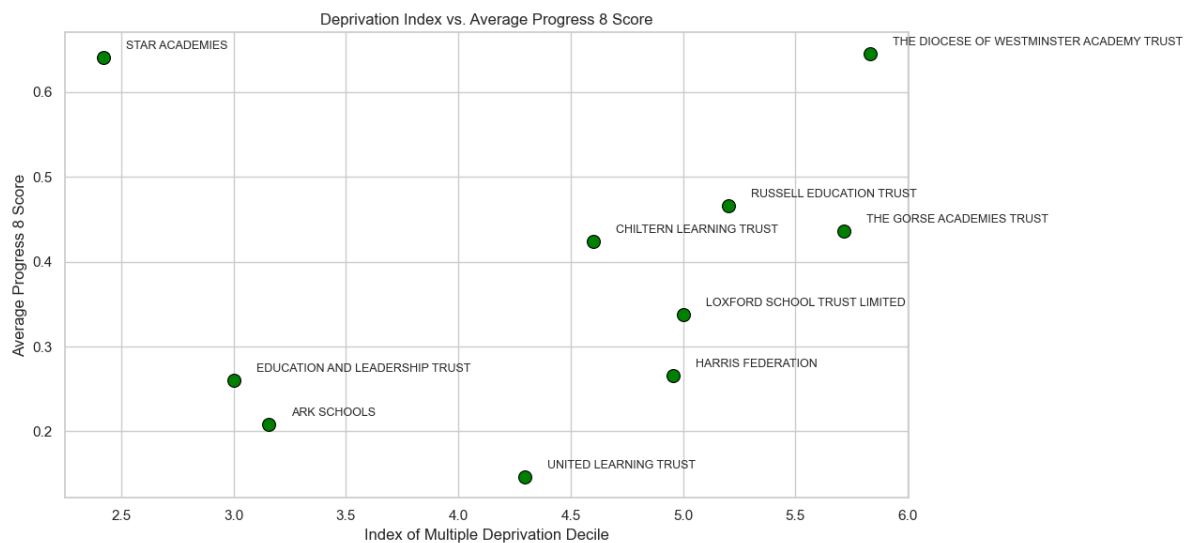
x='deprivation_index',
y='avg_progress8_score',
data=top_10_MATs,
s=100,
color='green',
edgecolor='black'
)
plt.title('Deprivation Index vs. Average Progress 8 Score')
plt.xlabel('Index of Multiple Deprivation Decile')
plt.ylabel('Average Progress 8 Score')
plt.tight_layout()

# Annotate MAT names
for idx, row in top_10_MATs.iterrows():
    plt.text(row['deprivation_index']+0.1, row['avg_progress8_score']+0.01,
             row['Trust_Name'], fontsize=9)

images_dir = 'images'
image_path = os.path.join(images_dir, 'Obj3_Deprivation Index vs P8 top 10 MATS.png')
plt.savefig(image_path)

plt.show()

```



Plot a scatter plot of MATs and progress 8 disadvantages

```

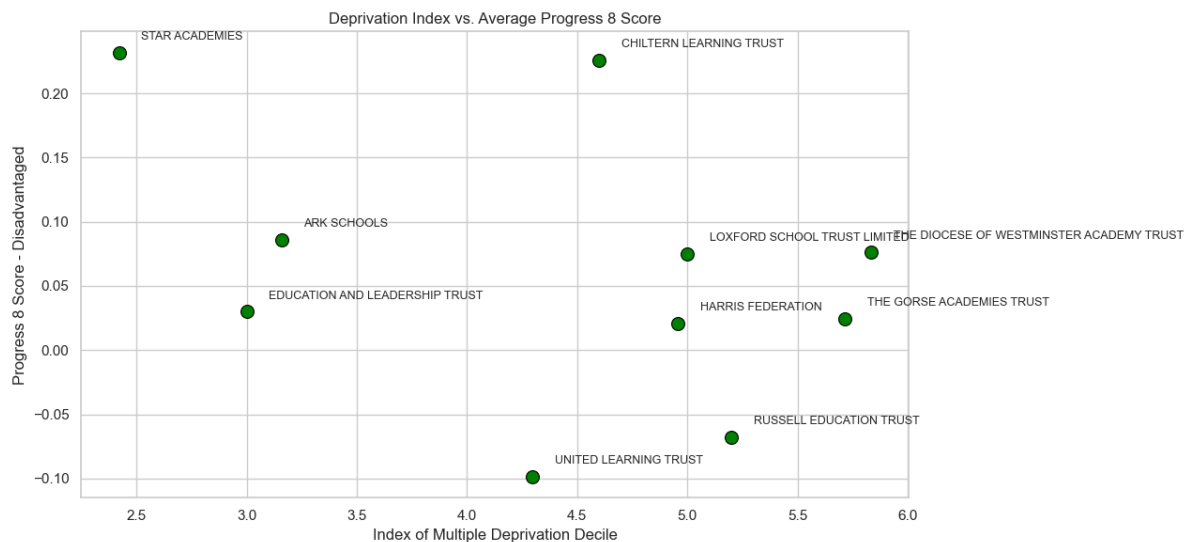
plt.figure(figsize=(10, 6))
sns.scatterplot(
    x='deprivation_index',
    y='prog8_score_disadv',
    data=top_10_MATs,
    s=100,
    color='green',
    edgecolor='black'
)
plt.title('Deprivation Index vs. Average Progress 8 Score')
plt.xlabel('Index of Multiple Deprivation Decile')
plt.ylabel('Progress 8 Score - Disadvantaged')
plt.tight_layout()

# Annotate MAT names
for idx, row in top_10_MATs.iterrows():
    plt.text(row['deprivation_index']+0.1, row['prog8_score_disadv']+0.01,
             row['Trust_Name'], fontsize=9)

images_dir = 'images'
image_path = os.path.join(images_dir, 'Obj3_Progress8 Disadv vs Deprivation Index for Top 10 MATs')
plt.savefig(image_path)

plt.show()

```



Before we can analyse the correlation coefficients I would need to standardise the data

```
# Columns to standardise
corr_columns = ['avg_progress8_score', 'prog8_score_disadv',
                'prog8_score_nondisadv', 'deprivation_index', 'progress8_gap',
                'attainment8_gap', 'maths_gap', 'english_gap', 'FiveGCSE_gap',
                'school_count']

scaler = StandardScaler() # this will give a mean of 0 and SD of 1

#filter data
top_10_MATs_standardized = top_10_MATs.copy()
top_10_MATs_standardized[corr_columns] = scaler.fit_transform(top_10_MATs[corr_columns])

print(top_10_MATs_standardized.head())
```

	Trust_Name	avg_progress8_score \
975	STAR ACADEMIES	1.586519
219	CHILTERN LEARNING TRUST	0.252807
57	ARK SCHOOLS	-1.075069
1096	THE DIOCESE OF WESTMINSTER ACADEMY TRUST	1.614075
628	LOXFORD SCHOOL TRUST LIMITED	-0.279996

	prog8_score_disadv	prog8_score_nondisadv	progress8_gap \
975	1.684115	-0.725355	-1.680816
219	1.629234	0.534547	-0.717045
57	0.249948	-1.272304	-1.105799
1096	0.160204	1.060226	0.670404
628	0.143809	-1.393449	-1.122556

	attainment8_gap	maths_gap	english_gap	FiveGCSE_gap \
975	-1.044491	-1.314392	-0.789312	-1.534426
219	-0.575425	-0.216238	-0.341835	-0.421951
57	-1.082402	-0.343113	-0.589626	-0.542685
1096	0.413372	-1.255853	-1.145190	0.206152
628	-1.428343	-1.193430	-1.206268	-0.708694

	deprivation_index	school_count
975	-1.782039	0.575651
219	0.162376	-0.745515

57	-1.124507	0.575651
1096	1.262959	-0.651146
628	0.519322	-0.839884

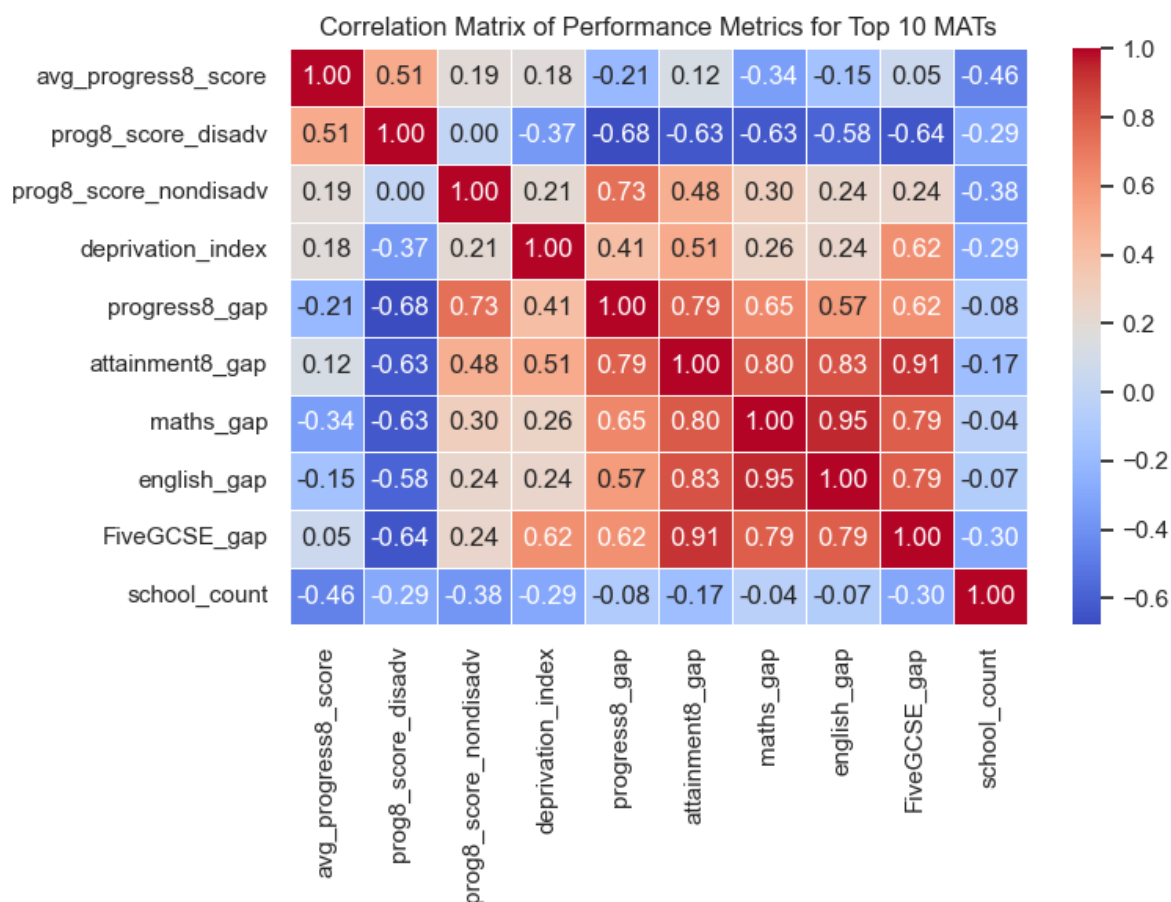
```
# Selecting the needed columns for correlation
corr_columns = ['avg_progress8_score', 'prog8_score_disadv',
                'prog8_score_nondisadv', 'deprivation_index', 'progress8_gap', 'attainment8_
                'maths_gap', 'english_gap', 'FiveGCSE_gap',
                'school_count']

# Compute the correlation matrix
corr_matrix = top_10_MATs[corr_columns].corr()

# Plot the heatmap
plt.figure(figsize=(8, 6))
sns.heatmap(corr_matrix, annot=True, cmap='coolwarm', fmt=".2f", linewidths=0.5)
plt.title('Correlation Matrix of Performance Metrics for Top 10 MATs')
plt.tight_layout()

#save the file in the data folder
images_dir = 'images'
image_path = os.path.join(images_dir, 'Obj3_Correlation Matrix top 10 MATS.png')
plt.savefig(image_path)

plt.show()
```



Project Outcome

Overview of Results

Objective 1: Evaluate National Disparities in Educational Performance

There significant gaps between non-disadvantaged and disadvantaged pupils including attainment 8, progress 8, Maths, English and strong passes in both. Disadvantaged pupils lag behind by approximately 1.45 GCSE grades per subject and have an attainment 8 gap of 11.6 points. Their Progress 8 scores are 0.6 grades lower across subjects than their peers, suggesting significant performance gaps.

Objective 2: Identify and Analyse Outlier Schools in Positive Progress 8 of Disadvantaged Pupils

Schools excelling in progress 8 for disadvantaged students, tend to support all students very well and have a strong positive correlation (0.85) between overall and disadvantaged pupils. Funding has a negative correlation with Progress 8 scores for disadvantaged pupils, and could be investigated further.

Objective 3: Identify and Evaluate Top Performing Multi-Academy Trusts (MATs)

High performing MAT have shown a strong positive correlation (0.51) between progress 8 scores for disadvantaged students and overall scores. Although socio-economic factors negatively correlate (-0.37) with progress, for high performing MATs this hasn't been seen to be a barrier; Star Academies for example is one of the highest performing MATs in the country, yet faces the highest deprivation average of all MATs, suggesting a robust pedagogical strategy and governance to run its schools. Such high performing MATs are good at closing the gap (smallest is 0.264 progress 8) between disadvantaged and advantaged students, demonstrating efficient use of funding and better equity.

Objective 1: Evaluate National Disparities in Educational Performance Between Advantaged and Disadvantaged Pupils

Explanation of Results:

There are positive gaps in all categories measured between advantaged and disadvantaged pupils, confirming that nationally, disadvantaged pupils are behind in every academic measure.

Attainment 8 Gap:

- Attainment 8 Disadvantaged: 40.22
- Attainment 8 Non-Disadvantaged: 51.83
- Attainment 8 Gap: 11.61

Analysis: The attainment 8 gap of 11.6 points between disadvantaged and advantaged pupils nationally, suggest approximately 1.45 GCSE grades lower per subject for disadvantaged students ($11.61/8 = 1.45125$ - as each subject is given a point based on the GCSE grade e.g. grade 9 = 9 points).

Progress 8 Gap:

- Progress 8 Disadvantaged: -0.47
- Progress 8 Non-Disadvantaged: 0.13
- Progress 8 Gap: 0.60

Analysis: Progress 8 gap of 0.60 that disadvantaged pupils are making 0.6 grades less progress across 8 subjects between keystage 2 and keystage 4 nationally. This would amount to 0.075 grade point less in each of the 8 subjects ($0.60/8=0.075$)

Subject Specific Gaps:

- Maths Disadvantaged: -0.44
- Maths Non-Disadvantaged: 0.11
- Maths Gap: 0.55
- English Disadvantaged: -0.46
- English Non-Disadvantaged: 0.12
- English Gap: 0.58

Analysis: Maths gap of 0.55 and English gap of 0.58 suggest, nationally, disadvantaged students are underperforming or making 0.55 grade less progress in maths and 0.58 less progress in English, between keystage 2 and keystage 4 nationally.

Percentage 9-5 Gap:

- Percentage Disadvantaged EngMaths_95: 28.09
- Percentage Nondisadv Student EngMaths_95: 50.01
- Percentage_95 Gap: 21.92

Analysis: Significant gap of 21.92 percentage points nationally between disadvantaged and advantaged students achieving grade 5 or above in English and Maths, suggests this needs to be addressed.

Visualisation

Distribution of Progress 8 Scores

The histogram shows an approximately normal distribution, as expected, since the results are standardised by exam boards. Most students would therefore have a progress 8 score of 0, with 68% of students falling within +1 or -1 standard deviations from the mean and 95% falling within +2 or -2 standard deviations from the mean.

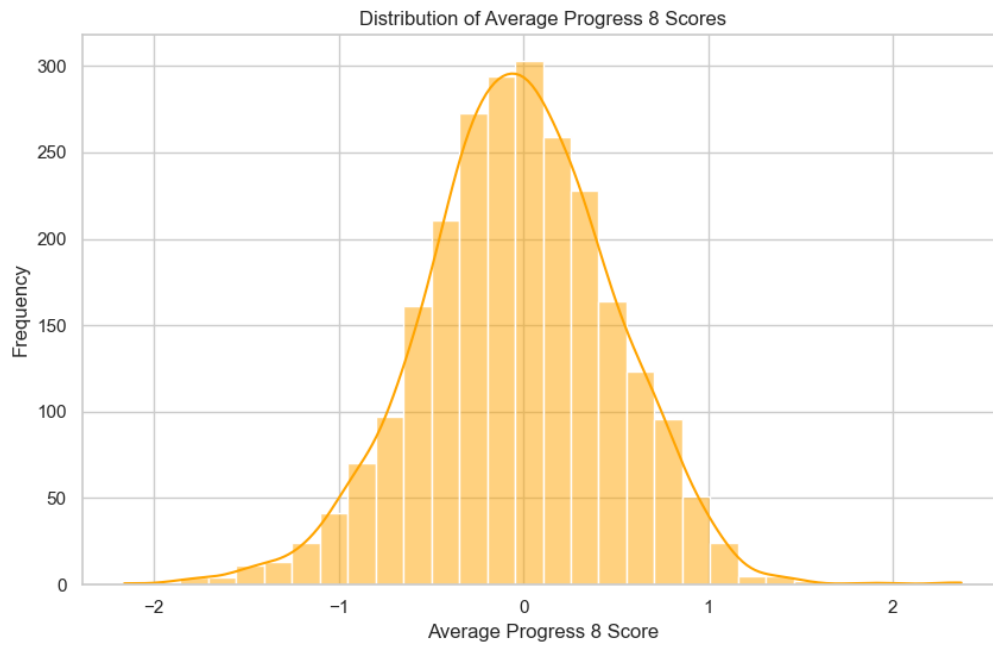


Figure 2: Obj1_progress8_distribution_nationally.png

Progress 8 and Attainment 8 Box Plots

Both box plots show disadvantaged students under performing. For progress 8, disadvantaged students have a negative progress 8 of -0.49 median score while advantaged students have a positive median score of 0.13, suggesting significant disparity. Both have a similar range and interquartile range with a number of outliers. For attainment 8, the gap and distribution is as expected given the results.

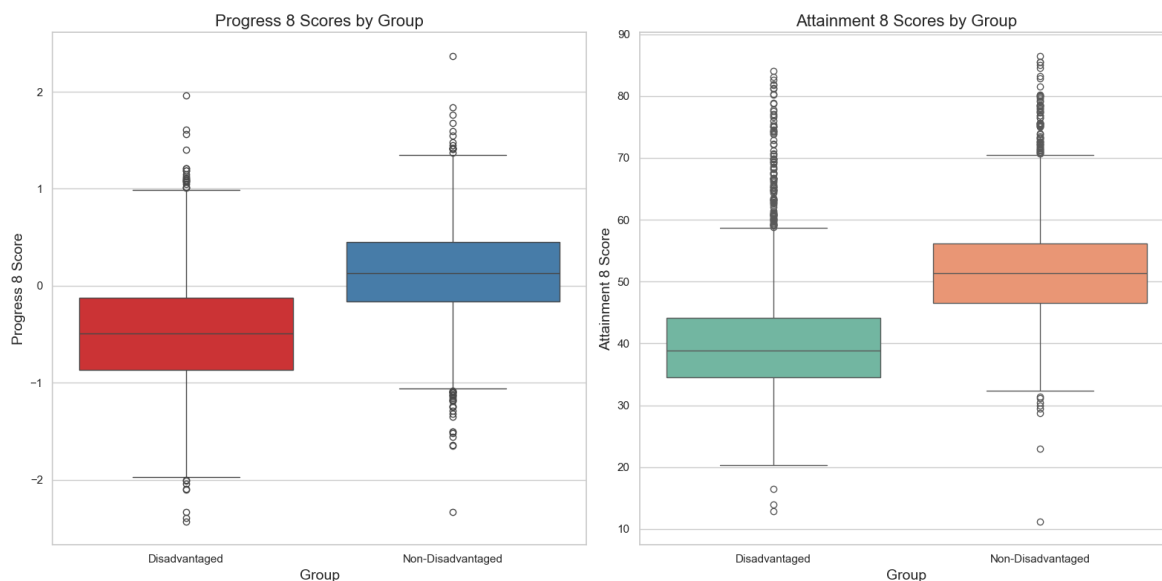


Figure 3: progress8_attainment8_boxplot.png

Percentage English and Mathematics Five Plus Box Plots

Maths Scores							
Summary	Median	Q1 (25%)	Q3 (75%)	IQR	Min	Max	Range
Disadvantaged	-0.47	-0.79	-0.12	0.67	-2.55	2.48	5.03
Non-Disadvantaged	0.11	-0.20	0.42	0.62	-1.65	2.95	4.60

English Scores							
Summary	Median	Q1 (25%)	Q3 (75%)	IQR	Min	Max	Range
Disadvantaged	-0.50	-0.90	-0.08	0.82	-3.06	2.19	5.25
Non-Disadvantaged	0.11	-0.20	0.44	0.64	-2.31	2.33	4.64

Both Maths and English have a negative median of -0.47 and -0.50 which is very concerning, given this is a national pattern, showing progress made by students between keystage 2 and keystage 4. English has a wider interquartile range for disadvantaged students, suggesting more variability. In both subjects, there is a greater difference between the minimum values, then between the maximum values, suggesting the disadvantaged students will significantly underperform than over perform.

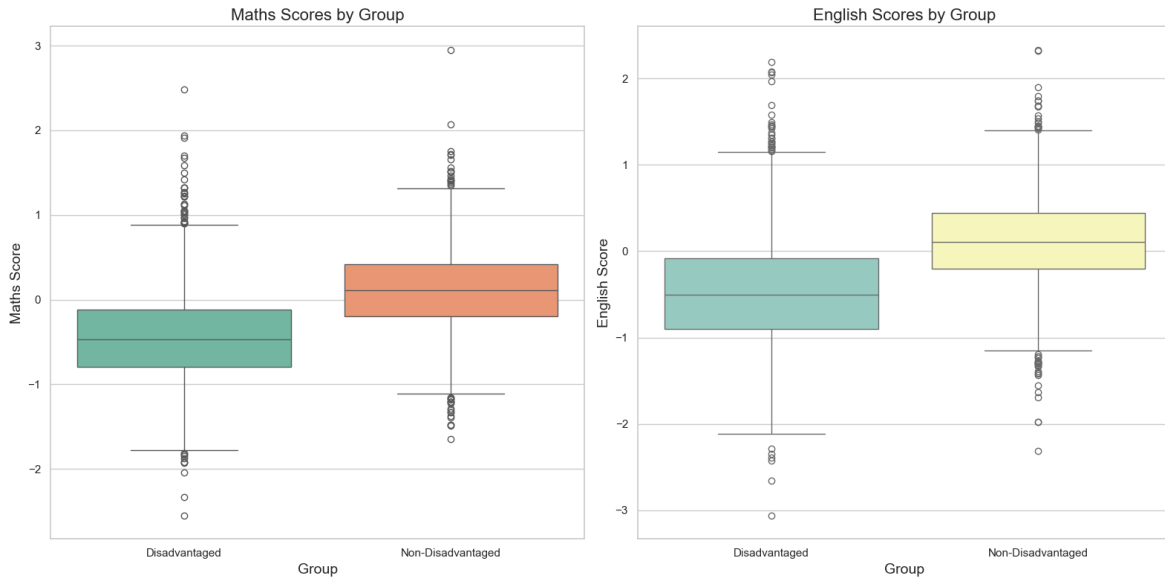


Figure 4: maths_english_scores_boxplot.png

Objective 2 Identify and Analyse Outlier Schools in Positive Progress 8 of Disadvantaged Pupils

Explanation of Results

Outlier schools for progress 8 were identified and then further categories as:

- a) Schools which are outliers only for non-disadvantaged pupils
- b) Schools which are outliers only for disadvantaged pupils
- c) schools which are outliers for both non-disadvantaged and disadvantaged
 - Overall schools which are outliers in both categories will do significantly better for disadvantaged pupils.
 - There is also a higher correlation (0.85) between progress 8 disadvantaged pupils and progress 8 in general, suggesting success breeds success.
 - Unexpectedly, funding (FSM(-0.45), total (-0.48) and pupil premium (-0.45)) all have negative correlation with progress 8 disadvantaged. This would need to be explored further as the range of the funding may be very small, and not being a good measure of proportionality.
 - Small positive correlation of progress 8 disadvantaged with percentage of disadvantage pupils (0.19) suggest disadvantaged pupils may do better where there are more such pupils.

- Index of multiple deprivation - has a negative correlation, suggesting lower values of the index ie. deprivation decreased, progress9 disadvantaged pupils will decline slightly, suggesting disadvantaged pupils' performance is expected to decrease when there is more deprivation.

Visualisation

Heatmap provides insights into outlier schools in areas such as progress 8 score, funding, deprivation index etc.



Figure 5: obj2_heatmap of outlier schools.png

Summary Statistics for 'Index of Multiple Deprivation Decile' by 'Outlier_Category'																				
Outlier_Category	Median	Q1 (25%)	Q3 (75%)	IQR	Min	Max	Range													
Both	7.0	4.5	8.50	4.00	1.0	10.0	9.0	None	6.0	3.0	8.00	5.00	1.0	10.0	9.0	Only_Disadv	4.5	3.0	7.00	4.00
Only_Disadv	4.5	3.0	7.00	4.00	1.0	9.0	8.0	Only_NonDisadv	3.0	2.0										