1. Start to get a feel for the data

When dealing with new datasets, it is always useful to look over any relevant documentation to get a feel for the data you will be handling.

Investigate the *Connected Nations 2022: Interactive report* (Ofcom, 2023a) which is available at:

https://www.ofcom.org.uk/research-and-data/multi-sector-research/infrastructure-research/connected-nations-2022/interactive-report

One of the investigations you should do is to compare the fixed broadband coverage in your local authority with that of the UK as a whole. Take a single screenshot of what you have found and explain in no more than 100 words what the result says about broadband coverage in your local authority.

If you are not based in the UK then if possible choose a local authority with which you have some connection.

You must include your screenshot in the `images/` directory.

(3 marks)

Ofcom (2023a) *Connected Nations 2022: Interactive report*. Available at: https://www.ofcom.org.uk/research-and-data/multi-sector-research/infrastructure-research/connected-nations-2022/interactive-report (Accessed: 26 September 2023).

Write your answer and include your image in this markdown cell

 No description has been provided for this image

This indicates the coverage of premises by fixed broadband networks in Ealing compared to the UK. The bar graphs' shapes in Ealing resemble those of the UK, with '>= 10 Mbit/s' being the highest and <10 Mbit/s DL being the lowest. However, Full Fibre coverage in Ealing accounts for 31%, whereas the UK has 41%. Therefore, the Full Fibre networks in Ealing have improved, though not to the same extent as across the entire UK.

2. Store the data in a MongoDB database

The datafiles for fixed broadband coverage in the UK are found in the folder:

```
2023J_TMA02_data/Ofcom_fixed
```

There are five csv files - one for each year - the year is indicated in the filename. Each csv file has an associated pdf which includes metadata. The datafiles and

metadata pdfs were downloaded from Ofcom (2023b). The data is made available by Ofcom under the Open Government Licence v3.0.

Your first task is to import the data, clean it, and store it in MongoDB. You will need to use '2019–2023 structural changes to local government in England' (2023) because you will need to be able to compare broadband coverage between different years.

When doing this you should consider that:

- Not all the csv have the same column names.
- Not all the csv have the same number of columns.
- There is ambiguous data.
- The year is not included in the file, only in the filename.

As well as the data imported from the csv files, each MongoDB document should have a field with a value referencing the year that the data was collected.

You might find it useful to use python's glob or os modules which allow you to create a list of filenames in a directory.

Use comments and Markdown cells to discuss and justify any decisions you make when importing the data.

(10 marks)

'2019–2023 structural changes to local government in England' (2023) Wikipedia. Available at: https://en.wikipedia.org/wiki/2019%E2%80%932023_structural_changes_to_local_gover (Accessed: 26 September 2023).

Ofcom (2023b) *Connected Nations and infrastructure reports*. Available at: https://www.ofcom.org.uk/research-and-data/multi-sector-research/infrastructure-research (Accessed: 24 September 2023).

In [1]:
```
# Write your answer in this code cell.
# Use additional cells if necessary, including markdown
# cells to explain your decisions and code.

import pandas as pd
import matplotlib.pyplot as plt
import glob
import re
import chardet

import pymongo
import folium
```

In [2]:
```
!ls 2023J_TMA02_data/Ofcom_fixed
```

```
201909_fixed_laua_coverage_r01.csv
202009_fixed_laua_coverage_r01.csv
202109_fixed_laua_coverage_r01.csv
202209-about-fixed-local-and-unitary-authority.pdf
202209_fixed_laua_coverage_r02.csv
202305_fixed_laua_coverage_r02.csv
cn-2020-about-fixed-coverage-local-and-unitary-authority.pdf
cn-2021-about-fixed-laua-coverage.pdf
connected-nations-2019-about-fixed-local-unitary-authority-area.pdf
fixed-coverage-local-unitary-authority-202305-v2.pdf
```

2019

In [3]:
```
# Check encoding
chardet.detect(open('2023J_TMA02_data/Ofcom_fixed/201909_fixed_laua_cover
```

Out[3]:
```
{'encoding': 'ascii', 'confidence': 1.0, 'language': ''}
```

data file:

- encoding: ascii

In [4]:
```
# Read the CSV file into a DataFrame
df2019 = pd.read_csv('2023J_TMA02_data/Ofcom_fixed/201909_fixed_laua_cove
```

In [5]:
```
df2019.head()
```

Out[5]:

	laua	laua_name	All Premises	All Matched Premises	SFBB availability (% premises)	UFBB availability (% premises)	Full availa prem
0	S12000033	ABERDEEN CITY	125441	125311	73.3	20.1	
1	S12000034	ABERDEENSHIRE	125085	124305	78.5	2.8	
2	E07000223	ADUR	29770	29760	16.3	82.4	
3	E07000026	ALLERDALE	51385	51284	89.8	1.7	
4	E07000032	AMBER VALLEY	60674	60596	67.4	25.3	

5 rows × 38 columns

In [6]:
```
df2019.tail()
```

Out[6]:

	laua	laua_name	All Premises	All Matched Premises	SFBB availability (% premises)	UFBB availability (% premises)	Full Fi availabi premis
377	E07000238	WYCHAVON	62475	62114	82.9	8.6	
378	E07000007	WYCOMBE	76433	76345	68.5	26.5	
379	E07000128	WYRE	56343	56280	88.4	5.9	
380	E07000239	WYRE FOREST	48100	48061	49.3	46.7	
381	E06000014	YORK	98735	98548	23.0	70.8	4

5 rows × 38 columns

In [7]:
```
# Data volume
df2019.shape
```

Out[7]: (382, 38)

In [8]:
```
# Data type
df2019.dtypes
```

Out[8]:
```
laua                                                                object
laua_name                                                           object
All Premises                                                         int64
All Matched Premises                                                 int64
SFBB availability (% premises)                                     float64
UFBB availability (% premises)                                     float64
Full Fibre availability (% premises)                               float64
% of premises unable to receive 2Mbit/s                            float64
% of premises unable to receive 5Mbit/s                            float64
% of premises unable to receive 10Mbit/s                           float64
% of premises unable to receive 30Mbit/s                           float64
% of premises below the USO                                        float64
% of premises with NGA                                             float64
% of premises able to receive decent broadband from FWA            float64
% of premises able to receive SFBB from FWA                        float64
Number of premises with SFBB availability                            int64
Number of premises with UFBB availability                            int64
Number of premises with Full Fibre availability                      int64
Number of premises unable to receive 2Mbit/s                         int64
Number of premises unable to receive 5Mbit/s                         int64
Number of premises unable to receive 10Mbit/s                        int64
Number of premises unable to receive 30Mbit/s                        int64
Number of premises below the USO                                     int64
Number of premises with NGA                                          int64
Number of premises able to receive decent broadband from FWA         int64
Number of premises able to receive SFBB from FWA                     int64
Number of premises with 30<300Mbit/s download speed                  int64
Number of premises with >=300Mbit/s download speed                   int64
Number of premises with 0<2Mbit/s download speed                     int64
Number of premises with 2<5Mbit/s download speed                     int64
Number of premises with 5<10Mbit/s download speed                    int64
Number of premises with 10<30Mbit/s download speed                   int64
% of premises with 30<300Mbit/s download speed                     float64
% of premises with >=300Mbit/s download speed                      float64
% of premises with 0<2Mbit/s download speed                        float64
% of premises with 2<5Mbit/s download speed                        float64
% of premises with 5<10Mbit/s download speed                       float64
% of premises with 10<30Mbit/s download speed                      float64
dtype: object
```

In [9]:
```
# Count null values
df2019.isnull().sum()
```

Out[9]:
```
laua                                                    0
laua_name                                               0
All Premises                                            0
All Matched Premises                                    0
SFBB availability (% premises)                          0
UFBB availability (% premises)                          0
Full Fibre availability (% premises)                    0
% of premises unable to receive 2Mbit/s                 0
% of premises unable to receive 5Mbit/s                 0
% of premises unable to receive 10Mbit/s                0
% of premises unable to receive 30Mbit/s                0
% of premises below the USO                             0
% of premises with NGA                                  0
% of premises able to receive decent broadband from FWA 0
% of premises able to receive SFBB from FWA             0
Number of premises with SFBB availability               0
Number of premises with UFBB availability               0
Number of premises with Full Fibre availability         0
Number of premises unable to receive 2Mbit/s            0
Number of premises unable to receive 5Mbit/s            0
Number of premises unable to receive 10Mbit/s           0
Number of premises unable to receive 30Mbit/s           0
Number of premises below the USO                        0
Number of premises with NGA                             0
Number of premises able to receive decent broadband from FWA  0
Number of premises able to receive SFBB from FWA        0
Number of premises with 30<300Mbit/s download speed     0
Number of premises with >=300Mbit/s download speed      0
Number of premises with 0<2Mbit/s download speed        0
Number of premises with 2<5Mbit/s download speed        0
Number of premises with 5<10Mbit/s download speed       0
Number of premises with 10<30Mbit/s download speed      0
% of premises with 30<300Mbit/s download speed          0
% of premises with >=300Mbit/s download speed           0
% of premises with 0<2Mbit/s download speed             0
% of premises with 2<5Mbit/s download speed             0
% of premises with 5<10Mbit/s download speed            0
% of premises with 10<30Mbit/s download speed           0
dtype: int64
```

In [10]:
```python
# Select only numerical columns
numerical_columns = df2019.select_dtypes(include=['number'])

# Use describe on numerical columns
numerical_description = numerical_columns.describe()

# Display the numerical description
display(numerical_description)
```

	All Premises	All Matched Premises	SFBB availability (% premises)	UFBB availability (% premises)	Full Fibre availability (% premises)	pre una re 2M
count	382.000000	382.000000	382.000000	382.000000	382.000000	382.00
mean	80678.520942	80532.005236	46.587435	46.857068	8.586649	0.40
std	52521.559617	52472.885870	25.255563	28.551654	10.401911	0.79
min	1681.000000	1678.000000	1.100000	0.000000	0.000000	0.00
25%	48100.250000	48070.250000	24.225000	18.450000	2.400000	0.00
50%	64867.500000	64782.500000	42.450000	51.800000	4.950000	0.10
75%	98485.750000	98413.250000	69.850000	72.175000	11.100000	0.40
max	469208.000000	468772.000000	97.800000	97.000000	97.000000	7.90

8 rows × 36 columns

2020

In [11]:
```
# Check encoding
chardet.detect(open('2023J_TMA02_data/Ofcom_fixed/202009_fixed_laua_cover
```

Out[11]:
```
{'encoding': 'ascii', 'confidence': 1.0, 'language': ''}
```

data file:

- encoding: ascii

In [12]:
```
# Read the CSV file into a DataFrame
df2020 = pd.read_csv('2023J_TMA02_data/Ofcom_fixed/202009_fixed_laua_cove
df2020.head()
```

Out[12]:

	laua	laua_name	All Premises	All Matched Premises	SFBB availability (% premises)	UFBB (100Mbit/s) availability (% premises)	avail pre
0	S12000033	ABERDEEN CITY	126176	125948	94.6	49.0	
1	S12000034	ABERDEENSHIRE	126065	125176	82.9	7.2	
2	E07000223	ADUR	29779	29755	98.8	85.8	
3	E07000026	ALLERDALE	51647	51483	92.3	2.8	
4	E07000032	AMBER VALLEY	61134	60972	94.7	30.2	

5 rows × 40 columns

In [13]:
```
df2020.tail()
```

Out[13]:

	laua	laua_name	All Premises	All Matched Premises	SFBB availability (% premises)	UFBB (100Mbit/s) availability (% premises)	availa prem
374	W06000006	WREXHAM	65867	65212	94.4	37.3	
375	E07000238	WYCHAVON	62536	62215	93.8	19.2	
376	E07000128	WYRE	56527	56411	95.1	22.7	
377	E07000239	WYRE FOREST	48237	48173	96.8	47.9	
378	E06000014	YORK	95949	95674	94.1	75.5	

5 rows × 40 columns

In [14]:
```
# Data volume
df2020.shape
```

Out[14]:
```
(379, 40)
```

In [15]:
```
# Data types
df2020.dtypes
```

Out[15]:
```
laua                                                                object
laua_name                                                           object
All Premises                                                         int64
All Matched Premises                                                 int64
SFBB availability (% premises)                                     float64
UFBB (100Mbit/s) availability (% premises)                         float64
UFBB availability (% premises)                                     float64
Full Fibre availability (% premises)                               float64
Gigabit availability (% premises)                                  float64
% of premises unable to receive 2Mbit/s                            float64
% of premises unable to receive 5Mbit/s                            float64
% of premises unable to receive 10Mbit/s                           float64
% of premises unable to receive 30Mbit/s                           float64
% of premises below the USO                                        float64
% of premises with NGA                                             float64
% of premises able to receive decent broadband from FWA            float64
Number of premises with SFBB availability                            int64
Number of premises with UFBB (100Mbit/s) availability                int64
Number of premises with UFBB availability                            int64
Number of premises with Full Fibre availability                      int64
Number of premises with Gigabit availability                         int64
Number of premises unable to receive 2Mbit/s                         int64
Number of premises unable to receive 5Mbit/s                         int64
Number of premises unable to receive 10Mbit/s                        int64
Number of premises unable to receive 30Mbit/s                        int64
Number of premises below the USO                                     int64
Number of premises with NGA                                          int64
Number of premises able to receive decent broadband from FWA         int64
Number of premises with 30<300Mbit/s download speed                  int64
Number of premises with >=300Mbit/s download speed                   int64
Number of premises with 0<2Mbit/s download speed                     int64
Number of premises with 2<5Mbit/s download speed                     int64
Number of premises with 5<10Mbit/s download speed                    int64
Number of premises with 10<30Mbit/s download speed                   int64
% of premises with 30<300Mbit/s download speed                     float64
% of premises with >=300Mbit/s download speed                      float64
% of premises with 0<2Mbit/s download speed                        float64
% of premises with 2<5Mbit/s download speed                        float64
% of premises with 5<10Mbit/s download speed                       float64
% of premises with 10<30Mbit/s download speed                      float64
dtype: object
```

In [16]:
```
# Count null values
df2020.isnull().sum()
```

Out[16]:
```
laua                                                    0
laua_name                                               0
All Premises                                            0
All Matched Premises                                    0
SFBB availability (% premises)                          0
UFBB (100Mbit/s) availability (% premises)              0
UFBB availability (% premises)                          0
Full Fibre availability (% premises)                    0
Gigabit availability (% premises)                       0
% of premises unable to receive 2Mbit/s                 0
% of premises unable to receive 5Mbit/s                 0
% of premises unable to receive 10Mbit/s                0
% of premises unable to receive 30Mbit/s                0
% of premises below the USO                             0
% of premises with NGA                                  0
% of premises able to receive decent broadband from FWA 0
Number of premises with SFBB availability               0
Number of premises with UFBB (100Mbit/s) availability   0
Number of premises with UFBB availability               0
Number of premises with Full Fibre availability         0
Number of premises with Gigabit availability            0
Number of premises unable to receive 2Mbit/s            0
Number of premises unable to receive 5Mbit/s            0
Number of premises unable to receive 10Mbit/s           0
Number of premises unable to receive 30Mbit/s           0
Number of premises below the USO                        0
Number of premises with NGA                             0
Number of premises able to receive decent broadband from FWA    0
Number of premises with 30<300Mbit/s download speed     0
Number of premises with >=300Mbit/s download speed      0
Number of premises with 0<2Mbit/s download speed        0
Number of premises with 2<5Mbit/s download speed        0
Number of premises with 5<10Mbit/s download speed       0
Number of premises with 10<30Mbit/s download speed      0
% of premises with 30<300Mbit/s download speed          0
% of premises with >=300Mbit/s download speed           0
% of premises with 0<2Mbit/s download speed             0
% of premises with 2<5Mbit/s download speed             0
% of premises with 5<10Mbit/s download speed            0
% of premises with 10<30Mbit/s download speed           0
dtype: int64
```

In [17]:
```python
# Select only numerical columns
numerical_columns = df2020.select_dtypes(include=['number'])

# Use describe on numerical columns
numerical_description = numerical_columns.describe()

# Display the numerical description
display(numerical_description)
```

	All Premises	All Matched Premises	SFBB availability (% premises)	UFBB (100Mbit/s) availability (% premises)	UFBB availability (% premises)	Full I availal premi
count	379.000000	379.000000	379.000000	379.000000	379.000000	379.00(
mean	82073.034301	81814.664908	94.187071	54.782850	52.300000	14.92
std	54691.926020	54579.787696	5.414738	28.169708	28.214368	14.85
min	1677.000000	1666.000000	56.500000	0.000000	0.000000	0.00(
25%	48481.000000	48342.000000	93.000000	30.500000	26.100000	4.55(
50%	65648.000000	65115.000000	95.900000	61.700000	58.100000	10.20(
75%	99125.000000	98969.000000	97.600000	78.950000	77.750000	20.10(
max	474257.000000	473084.000000	99.600000	97.500000	97.500000	97.50(

8 rows × 38 columns

In [18]:
```
# Find columns in df2019 but not in df2020
columns_only_in_df2019 = set(df2019.columns) - set(df2020.columns)

# Find columns in df2020 but not in df2019
columns_only_in_df2020 = set(df2020.columns) - set(df2019.columns)

# Display the results
print("Columns only in df2019:", columns_only_in_df2019)
print("Columns only in df2020:", columns_only_in_df2020)
```

```
Columns only in df2019: {'% of premises able to receive SFBB from FWA', 'N
umber of premises able to receive SFBB from FWA'}
Columns only in df2020: {'Gigabit availability (% premises)', 'Number of p
remises with UFBB (100Mbit/s) availability', 'UFBB (100Mbit/s) availabilit
y (% premises)', 'Number of premises with Gigabit availability'}
```

2021

In [19]:
```
# Check encoding
chardet.detect(open('2023J_TMA02_data/Ofcom_fixed/202109_fixed_laua_cover
```

Out[19]:
```
{'encoding': 'ascii', 'confidence': 1.0, 'language': ''}
```

data file:

- encoding: ascii

In [20]:
```
# Read the CSV file into a DataFrame
df2021 = pd.read_csv('2023J_TMA02_data/Ofcom_fixed/202109_fixed_laua_cove
df2021.head()
```

Out[20]:

	laua	laua_name	All Premises	All Matched Premises	SFBB availability (% premises)	UFBB (100Mbit/s) availability (% premises)	avail pre
0	S12000033	ABERDEEN CITY	127714	126771	94.7	66.6	
1	S12000034	ABERDEENSHIRE	126481	125378	82.8	13.8	
2	E07000223	ADUR	29884	29793	98.6	85.9	
3	E07000026	ALLERDALE	51933	51622	92.3	3.4	
4	E07000032	AMBER VALLEY	61555	61161	95.1	31.4	

5 rows × 40 columns

In [21]:
```
df2021.tail()
```

Out[21]:

	laua	laua_name	All Premises	All Matched Premises	SFBB availability (% premises)	UFBB (100Mbit/s) availability (% premises)	availa prem
369	W06000006	WREXHAM	66192	65306	94.6	42.0	
370	E07000238	WYCHAVON	63359	62695	94.2	23.8	
371	E07000128	WYRE	57413	57099	95.4	46.4	
372	E07000239	WYRE FOREST	48472	48204	96.7	48.5	
373	E06000014	YORK	96147	95638	94.2	77.5	

5 rows × 40 columns

In [22]:
```
# Data volume
df2021.shape
```

Out[22]: (374, 40)

In [23]:
```
# Count null values
df2021.isnull().sum()
```

Out[23]:

```
laua                                                    0
laua_name                                               0
All Premises                                            0
All Matched Premises                                    0
SFBB availability (% premises)                          0
UFBB (100Mbit/s) availability (% premises)              0
UFBB availability (% premises)                          0
Full Fibre availability (% premises)                    0
Gigabit availability (% premises)                       0
% of premises unable to receive 2Mbit/s                 0
% of premises unable to receive 5Mbit/s                 0
% of premises unable to receive 10Mbit/s                0
% of premises unable to receive 30Mbit/s                0
% of premises below the USO                             0
% of premises with NGA                                  0
% of premises able to receive decent broadband from FWA 0
Number of premises with SFBB availability               0
Number of premises with UFBB (100Mbit/s) availability   0
Number of premises with UFBB availability               0
Number of premises with Full Fibre availability         0
Number of premises with Gigabit availability            0
Number of premises unable to receive 2Mbit/s            0
Number of premises unable to receive 5Mbit/s            0
Number of premises unable to receive 10Mbit/s           0
Number of premises unable to receive 30Mbit/s           0
Number of premises below the USO                        0
Number of premises with NGA                             0
Number of premises able to receive decent broadband from FWA    0
Number of premises with 30<300Mbit/s download speed     0
Number of premises with >=300Mbit/s download speed      0
Number of premises with 0<2Mbit/s download speed        0
Number of premises with 2<5Mbit/s download speed        0
Number of premises with 5<10Mbit/s download speed       0
Number of premises with 10<30Mbit/s download speed      0
% of premises with 30<300Mbit/s download speed          0
% of premises with >=300Mbit/s download speed           0
% of premises with 0<2Mbit/s download speed             0
% of premises with 2<5Mbit/s download speed             0
% of premises with 5<10Mbit/s download speed            0
% of premises with 10<30Mbit/s download speed           0
dtype: int64
```

In [24]:

```
# Select only numerical columns
numerical_columns2 = df2021.select_dtypes(include=['number'])

# Use describe on numerical columns
numerical_description2 = numerical_columns2.describe()

# Display the numerical description
display(numerical_description2)
```

	All Premises	All Matched Premises	SFBB availability (% premises)	UFBB (100Mbit/s) availability (% premises)	UFBB availability (% premises)	Full F availab premi
count	374.000000	374.000000	374.000000	374.000000	374.000000	374.000
mean	83731.264706	83184.727273	94.344652	59.760963	57.726203	23.960
std	55452.666179	55099.509244	5.246693	26.495761	26.509567	18.610
min	1683.000000	1661.000000	56.500000	1.100000	1.100000	0.900
25%	49355.500000	49097.750000	93.225000	39.275000	36.875000	10.225
50%	66536.000000	66171.500000	95.900000	67.750000	65.600000	19.500
75%	101537.000000	100732.500000	97.500000	82.275000	80.500000	33.575
max	474961.000000	471159.000000	99.500000	97.600000	97.600000	97.600

8 rows × 38 columns

In [25]:
```
# Find columns in df2019 but not in df2020
columns_only_in_df2019 = set(df2019.columns) - set(df2021.columns)

# Find columns in df2021 but not in df2019
columns_only_in_df2021 = set(df2021.columns) - set(df2019.columns)

# Display the results
print("Columns only in df2019:", columns_only_in_df2019)
print("Columns only in df2021:", columns_only_in_df2021)
```

```
Columns only in df2019: {'% of premises able to receive SFBB from FWA', 'N
umber of premises able to receive SFBB from FWA'}
Columns only in df2021: {'Gigabit availability (% premises)', 'Number of p
remises with UFBB (100Mbit/s) availability', 'UFBB (100Mbit/s) availabilit
y (% premises)', 'Number of premises with Gigabit availability'}
```

In [26]:
```
# Find columns in df2020 but not in df2021
columns_only_in_df2020 = set(df2020.columns) - set(df2021.columns)

# Find columns in df2021 but not in df2020
columns_only_in_df2021 = set(df2021.columns) - set(df2020.columns)

# Display the results
print("Columns only in df2019:", columns_only_in_df2020)
print("Columns only in df2021:", columns_only_in_df2021)
```

```
Columns only in df2019: set()
Columns only in df2021: set()
```

it has the same columns in 2021 and 2020

2022

In [27]:
```
# Check encoding
chardet.detect(open('2023J_TMA02_data/Ofcom_fixed/202209_fixed_laua_cover
```

Out[27]:
```
{'encoding': 'ascii', 'confidence': 1.0, 'language': ''}
```

In [28]:
```
# Read the CSV file into a DataFrame
df2022 = pd.read_csv('2023J_TMA02_data/Ofcom_fixed/202209_fixed_laua_cove
```

In [29]:
```
df2022.head()
```

Out[29]:

	laua	laua_name	All Premises	All Matched Premises	SFBB availability (% premises)	UFBB (100Mbit/s) availability (% premises)	avail pre
0	S12000033	ABERDEEN CITY	128708	128294	95.8	79.1	
1	S12000034	ABERDEENSHIRE	127941	127265	84.2	20.6	
2	E07000223	ADUR	29971	29920	99.1	91.0	
3	E07000026	ALLERDALE	52309	52133	92.7	5.4	
4	E07000032	AMBER VALLEY	62170	61902	96.1	49.4	

5 rows × 40 columns

In [30]:
```
df2022.tail()
```

Out[30]:

	laua	laua_name	All Premises	All Matched Premises	SFBB availability (% premises)	UFBB (100Mbit/s) availability (% premises)	availa prem
369	W06000006	WREXHAM	66672	65735	95.2	48.6	
370	E07000238	WYCHAVON	64057	63530	95.1	35.8	
371	E07000128	WYRE	58069	57900	97.0	60.3	
372	E07000239	WYRE FOREST	48894	48679	97.3	55.7	
373	E06000014	YORK	96526	96317	94.7	75.8	

5 rows × 40 columns

In [31]:
```
# Data volume
df2022.shape
```

Out[31]:
```
(374, 40)
```

In [32]:
```
# Count null values
df2022.isnull().sum()
```

Out[32]:
```
laua                                                    0
laua_name                                               0
All Premises                                            0
All Matched Premises                                    0
SFBB availability (% premises)                          0
UFBB (100Mbit/s) availability (% premises)              0
UFBB availability (% premises)                          0
Full Fibre availability (% premises)                    0
Gigabit availability (% premises)                       0
% of premises unable to receive 2Mbit/s                 0
% of premises unable to receive 5Mbit/s                 0
% of premises unable to receive 10Mbit/s                0
% of premises unable to receive 30Mbit/s                0
% of premises below the USO                             0
% of premises with NGA                                  0
% of premises able to receive decent broadband from FWA 0
Number of premises with SFBB availability               0
Number of premises with UFBB (100Mbit/s) availability   0
Number of premises with UFBB availability               0
Number of premises with Full Fibre availability         0
Number of premises with Gigabit availability            0
Number of premises unable to receive 2Mbit/s            0
Number of premises unable to receive 5Mbit/s            0
Number of premises unable to receive 10Mbit/s           0
Number of premises unable to receive 30Mbit/s           0
Number of premises below the USO                        0
Number of premises with NGA                             0
Number of premises able to receive decent broadband from FWA    0
Number of premises with 30<300Mbit/s download speed     0
Number of premises with >=300Mbit/s download speed      0
Number of premises with 0<2Mbit/s download speed        0
Number of premises with 2<5Mbit/s download speed        0
Number of premises with 5<10Mbit/s download speed       0
Number of premises with 10<30Mbit/s download speed      0
% of premises with 30<300Mbit/s download speed          0
% of premises with >=300Mbit/s download speed           0
% of premises with 0<2Mbit/s download speed             0
% of premises with 2<5Mbit/s download speed             0
% of premises with 5<10Mbit/s download speed            0
% of premises with 10<30Mbit/s download speed           0
dtype: int64
```

In [33]:
```python
# Select only numerical columns
numerical_columns3 = df2022.select_dtypes(include=['number'])

# Use describe on numerical columns
numerical_description3 = numerical_columns3.describe()

# Display the numerical description
display(numerical_description3)
```

	All Premises	All Matched Premises	SFBB availability (% premises)	UFBB (100Mbit/s) availability (% premises)	UFBB availability (% premises)	Full F availab premi
count	374.000000	374.000000	374.000000	374.000000	374.000000	374.000
mean	84761.181818	84349.454545	95.325668	67.605615	66.110428	38.207
std	56035.842818	55767.364674	4.584913	22.297820	22.464340	20.553
min	1686.000000	1669.000000	58.700000	1.600000	1.600000	1.600
25%	50147.500000	49881.250000	94.500000	52.950000	50.775000	22.300
50%	67870.000000	67463.500000	96.750000	74.400000	72.200000	37.800
75%	102291.250000	102037.000000	98.000000	85.975000	84.800000	51.275
max	477617.000000	475094.000000	99.500000	97.700000	97.700000	97.700

8 rows × 38 columns

In [34]:
```
# Find columns in df2019 but not in df2020
columns_only_in_df2019 = set(df2019.columns) - set(df2022.columns)

# Find columns in df2020 but not in df2019
columns_only_in_df2022 = set(df2022.columns) - set(df2019.columns)

# Display the results
print("Columns only in df2019:", columns_only_in_df2019)
print("Columns only in df2020:", columns_only_in_df2022)
```

```
Columns only in df2019: {'% of premises able to receive SFBB from FWA', 'N
umber of premises able to receive SFBB from FWA'}
Columns only in df2020: {'Gigabit availability (% premises)', 'Number of p
remises with UFBB (100Mbit/s) availability', 'UFBB (100Mbit/s) availabilit
y (% premises)', 'Number of premises with Gigabit availability'}
```

In [35]:
```
# Find columns in df2021 but not in df2020
columns_only_in_df2021 = set(df2021.columns) - set(df2022.columns)

# Find columns in df2020 but not in df2021
columns_only_in_df2022 = set(df2022.columns) - set(df2021.columns)

# Display the results
print("Columns only in df2021:", columns_only_in_df2021)
print("Columns only in df2020:", columns_only_in_df2022)
```

```
Columns only in df2021: set()
Columns only in df2020: set()
```

Columns: 2020, 2021, 2022 are the same

2023

In [36]:
```
# Check encoding
chardet.detect(open('2023J_TMA02_data/Ofcom_fixed/202305_fixed_laua_cover
```

Out[36]:
```
{'encoding': 'ascii', 'confidence': 1.0, 'language': ''}
```

In [37]:
```
# Read the CSV file into a DataFrame
df2023 = pd.read_csv('2023J_TMA02_data/Ofcom_fixed/202305_fixed_laua_cove
df2023.head()
```

Out[37]:

	laua	laua_name	All Premises	All Matched Premises	SFBB availability (% premises)	UFBB (100Mbit/s) availability (% premises)	avail pre
0	S12000033	ABERDEEN CITY	129315	129197	97.2	84.8	
1	S12000034	ABERDEENSHIRE	128408	128070	85.9	25.5	
2	E07000223	ADUR	29985	29953	99.1	92.8	
3	E07000026	ALLERDALE	52482	52364	93.1	6.0	
4	E07000032	AMBER VALLEY	62512	62430	97.2	62.4	

5 rows × 40 columns

In [38]:
```
df2023.head()
```

Out[38]:

	laua	laua_name	All Premises	All Matched Premises	SFBB availability (% premises)	UFBB (100Mbit/s) availability (% premises)	avail pre
0	S12000033	ABERDEEN CITY	129315	129197	97.2	84.8	
1	S12000034	ABERDEENSHIRE	128408	128070	85.9	25.5	
2	E07000223	ADUR	29985	29953	99.1	92.8	
3	E07000026	ALLERDALE	52482	52364	93.1	6.0	
4	E07000032	AMBER VALLEY	62512	62430	97.2	62.4	

5 rows × 40 columns

In [39]:
```
# Data volume
df2023.shape
```

Out[39]:
```
(374, 40)
```

In [40]:
```
# Count null values
df2023.isnull().sum()
```

Out[40]:
```
laua                                                    0
laua_name                                               0
All Premises                                            0
All Matched Premises                                    0
SFBB availability (% premises)                          0
UFBB (100Mbit/s) availability (% premises)              0
UFBB availability (% premises)                          0
Full Fibre availability (% premises)                    0
Gigabit availability (% premises)                       0
% of premises unable to receive 2Mbit/s                 0
% of premises unable to receive 5Mbit/s                 0
% of premises unable to receive 10Mbit/s                0
% of premises unable to receive 30Mbit/s                0
% of premises below the USO                             0
% of premises with NGA                                  0
% of premises able to receive decent broadband from FWA 0
Number of premises with SFBB availability               0
Number of premises with UFBB (100Mbit/s) availability   0
Number of premises with UFBB availability               0
Number of premises with Full Fibre availability         0
Number of premises with Gigabit availability            0
Number of premises unable to receive 2Mbit/s            0
Number of premises unable to receive 5Mbit/s            0
Number of premises unable to receive 10Mbit/s           0
Number of premises unable to receive 30Mbit/s           0
Number of premises below the USO                        0
Number of premises with NGA                             0
Number of premises able to receive decent broadband from FWA    0
Number of premises with 30<300Mbit/s download speed     0
Number of premises with >=300Mbit/s download speed      0
Number of premises with 0<2Mbit/s download speed        0
Number of premises with 2<5Mbit/s download speed        0
Number of premises with 5<10Mbit/s download speed       0
Number of premises with 10<30Mbit/s download speed      0
% of premises with 30<300Mbit/s download speed          0
% of premises with >=300Mbit/s download speed           0
% of premises with 0<2Mbit/s download speed             0
% of premises with 2<5Mbit/s download speed             0
% of premises with 5<10Mbit/s download speed            0
% of premises with 10<30Mbit/s download speed           0
dtype: int64
```

In [41]:
```python
# Select only numerical columns
numerical_columns4 = df2023.select_dtypes(include=['number'])

# Use describe on numerical columns
numerical_description4 = numerical_columns4.describe()

# Display the numerical description
display(numerical_description4)
```

	All Premises	All Matched Premises	SFBB availability (% premises)	UFBB (100Mbit/s) availability (% premises)	UFBB availability (% premises)	Full availa prem
count	374.000000	374.000000	374.000000	374.000000	374.000000	374.00
mean	85116.336898	84886.005348	96.072995	71.809091	70.550802	47.62
std	56252.163193	56070.215034	4.211346	20.414938	20.496678	20.76
min	1689.000000	1678.000000	59.100000	1.800000	1.800000	1.80
25%	50502.250000	50250.000000	95.400000	59.075000	57.550000	32.17
50%	68136.500000	67915.000000	97.300000	77.050000	75.350000	48.45
75%	102462.000000	102371.250000	98.300000	88.175000	87.200000	62.05
max	478734.000000	476604.000000	99.800000	98.400000	98.400000	98.40

8 rows × 38 columns

In [42]:
```
# Find columns in df2019 but not in df2023
columns_only_in_df2019 = set(df2019.columns) - set(df2023.columns)

# Find columns in df2023 but not in df2019
columns_only_in_df2023 = set(df2023.columns) - set(df2019.columns)

# Display the results
print("Columns only in df2019:", columns_only_in_df2019)
print("Columns only in df2023:", columns_only_in_df2023)
```

```
Columns only in df2019: {'% of premises able to receive SFBB from FWA', 'N
umber of premises able to receive SFBB from FWA'}
Columns only in df2023: {'Gigabit availability (% premises)', 'Number of p
remises with UFBB (100Mbit/s) availability', 'UFBB (100Mbit/s) availabilit
y (% premises)', 'Number of premises with Gigabit availability'}
```

In [43]:
```
# Find columns in df2020 but not in df2023
columns_only_in_df2020 = set(df2020.columns) - set(df2023.columns)

# Find columns in df2023 but not in df2020
columns_only_in_df2023 = set(df2023.columns) - set(df2020.columns)

# Display the results
print("Columns only in df2020:", columns_only_in_df2020)
print("Columns only in df2023:", columns_only_in_df2023)
```

```
Columns only in df2020: set()
Columns only in df2023: set()
```

In [44]:
```
df2019.dtypes
```

Out[44]:
```
laua                                                    object
laua_name                                               object
All Premises                                             int64
All Matched Premises                                     int64
SFBB availability (% premises)                         float64
UFBB availability (% premises)                         float64
Full Fibre availability (% premises)                   float64
% of premises unable to receive 2Mbit/s                float64
% of premises unable to receive 5Mbit/s                float64
% of premises unable to receive 10Mbit/s               float64
% of premises unable to receive 30Mbit/s               float64
% of premises below the USO                            float64
% of premises with NGA                                 float64
% of premises able to receive decent broadband from FWA float64
% of premises able to receive SFBB from FWA            float64
Number of premises with SFBB availability                int64
Number of premises with UFBB availability                int64
Number of premises with Full Fibre availability          int64
Number of premises unable to receive 2Mbit/s             int64
Number of premises unable to receive 5Mbit/s             int64
Number of premises unable to receive 10Mbit/s            int64
Number of premises unable to receive 30Mbit/s            int64
Number of premises below the USO                         int64
Number of premises with NGA                              int64
Number of premises able to receive decent broadband from FWA int64
Number of premises able to receive SFBB from FWA         int64
Number of premises with 30<300Mbit/s download speed      int64
Number of premises with >=300Mbit/s download speed       int64
Number of premises with 0<2Mbit/s download speed         int64
Number of premises with 2<5Mbit/s download speed         int64
Number of premises with 5<10Mbit/s download speed        int64
Number of premises with 10<30Mbit/s download speed       int64
% of premises with 30<300Mbit/s download speed         float64
% of premises with >=300Mbit/s download speed          float64
% of premises with 0<2Mbit/s download speed            float64
% of premises with 2<5Mbit/s download speed            float64
% of premises with 5<10Mbit/s download speed           float64
% of premises with 10<30Mbit/s download speed          float64
dtype: object
```

The dataframe of df2019 has not the column related to Gigabit('Gigabit availability (% premises)' and 'Number of premises with Gigabit availability').

The columns names of df2019: 'Number of premises able to receive SFBB from FWA' '% of premises able to receive SFBB from FWA' of df2019 are the same as 'SFBB availability (% premises)' and 'Number of premises with SFBB availability' respectively. In addition, the columns from 'UFBB availability (% premises)' and 'Number of premises with UFBB (100Mbit/s) availability' from df2019 are the same as 'UFBB (100Mbit/s) availability (% premises)' and 'Number of premises with UFBB (100Mbit/s) availability' from other dataframes (2020 - 2023).

Therefore, the names of df2019 are changed

In [45]:
```
df2019.rename(columns={'Number of premises able to receive SFBB from FWA'
                       '% of premises able to receive SFBB from FWA': 'SF
                       'UFBB availability (% premises)': 'UFBB (100Mbit/s
                       'Number of premises with UFBB (100Mbit/s) availabi
```

```
                        },
                inplace=True)
```

In [46]:
```
# Data types
df2023.dtypes
```

Out[46]:
```
laua                                                    object
laua_name                                               object
All Premises                                             int64
All Matched Premises                                     int64
SFBB availability (% premises)                         float64
UFBB (100Mbit/s) availability (% premises)             float64
UFBB availability (% premises)                         float64
Full Fibre availability (% premises)                   float64
Gigabit availability (% premises)                      float64
% of premises unable to receive 2Mbit/s                float64
% of premises unable to receive 5Mbit/s                float64
% of premises unable to receive 10Mbit/s               float64
% of premises unable to receive 30Mbit/s               float64
% of premises below the USO                            float64
% of premises with NGA                                 float64
% of premises able to receive decent broadband from FWA float64
Number of premises with SFBB availability                int64
Number of premises with UFBB (100Mbit/s) availability    int64
Number of premises with UFBB availability                int64
Number of premises with Full Fibre availability          int64
Number of premises with Gigabit availability             int64
Number of premises unable to receive 2Mbit/s             int64
Number of premises unable to receive 5Mbit/s             int64
Number of premises unable to receive 10Mbit/s            int64
Number of premises unable to receive 30Mbit/s            int64
Number of premises below the USO                         int64
Number of premises with NGA                              int64
Number of premises able to receive decent broadband from FWA int64
Number of premises with 30<300Mbit/s download speed      int64
Number of premises with >=300Mbit/s download speed       int64
Number of premises with 0<2Mbit/s download speed         int64
Number of premises with 2<5Mbit/s download speed         int64
Number of premises with 5<10Mbit/s download speed        int64
Number of premises with 10<30Mbit/s download speed       int64
% of premises with 30<300Mbit/s download speed         float64
% of premises with >=300Mbit/s download speed          float64
% of premises with 0<2Mbit/s download speed            float64
% of premises with 2<5Mbit/s download speed            float64
% of premises with 5<10Mbit/s download speed           float64
% of premises with 10<30Mbit/s download speed          float64
dtype: object
```

In [47]:
```
# Find 'laua_name' values in 2019 but not in 2020
laua_names_only_in_2019 = set(df2019['laua_name']) - set(df2020['laua_nam

# Display the results
display("laua_name values only in 2019:", laua_names_only_in_2019)

# Find 'laua_name' values in 2020 but not in 2019
laua_names_only_in_2020 = set(df2020['laua_name']) - set(df2019['laua_nam

# Display the results
display("laua_name values only in 2020:", laua_names_only_in_2020)
```

```
'laua_name values only in 2019:'
{'AYLESBURY VALE', 'CHILTERN', 'SOUTH BUCKS', 'WYCOMBE'}
'laua_name values only in 2020:'
{'BUCKINGHAMSHIRE'}
```

In [48]:
```
# Find 'laua_name' values in 2019 but not in 2021
laua_names_only_in_2019 = set(df2019['laua_name']) - set(df2021['laua_nam

# Display the results
display("laua_name values only in 2019:", laua_names_only_in_2019)

# Find 'laua_name' values in 2021 but not in 2019
laua_names_only_in_2021 = set(df2021['laua_name']) - set(df2019['laua_nam

# Display the results
display("laua_name values only in 2021:", laua_names_only_in_2021)
```

```
'laua_name values only in 2019:'
{'AYLESBURY VALE',
 'CHILTERN',
 'CORBY',
 'DAVENTRY',
 'EAST NORTHAMPTONSHIRE',
 'KETTERING',
 'NORTHAMPTON',
 'SOUTH BUCKS',
 'SOUTH NORTHAMPTONSHIRE',
 'WELLINGBOROUGH',
 'WYCOMBE'}
'laua_name values only in 2021:'
{'BUCKINGHAMSHIRE', 'NORTH NORTHAMPTONSHIRE', 'WEST NORTHAMPTONSHIRE'}
```

In [49]:
```
# Find 'laua_name' values in 2019 but not in 2022
laua_names_only_in_2019 = set(df2019['laua_name']) - set(df2022['laua_nam

# Display the results
display("laua_name values only in 2019:", laua_names_only_in_2019)

# Find 'laua_name' values in 2022 but not in 2019
laua_names_only_in_2022 = set(df2022['laua_name']) - set(df2019['laua_nam

# Display the results
display("laua_name values only in 2023:", laua_names_only_in_2022)
```

```
'laua_name values only in 2019:'
{'AYLESBURY VALE',
 'CHILTERN',
 'CORBY',
 'DAVENTRY',
 'EAST NORTHAMPTONSHIRE',
 'KETTERING',
 'NORTHAMPTON',
 'SOUTH BUCKS',
 'SOUTH NORTHAMPTONSHIRE',
 'WELLINGBOROUGH',
 'WYCOMBE'}
'laua_name values only in 2023:'
{'BUCKINGHAMSHIRE', 'NORTH NORTHAMPTONSHIRE', 'WEST NORTHAMPTONSHIRE'}
```

In [50]:
```
# Find 'laua_name' values in 2019 but not in 2023
laua_names_only_in_2019 = set(df2019['laua_name']) - set(df2023['laua_nam
```

```
# Display the results
display("laua_name values only in 2019:", laua_names_only_in_2019)

# Find 'laua_name' values in 2023 but not in 2019
laua_names_only_in_2023 = set(df2023['laua_name']) - set(df2019['laua_nam

# Display the results
display("laua_name values only in 2023:", laua_names_only_in_2023)
```

```
'laua_name values only in 2019:'
{'AYLESBURY VALE',
 'CHILTERN',
 'CORBY',
 'DAVENTRY',
 'EAST NORTHAMPTONSHIRE',
 'KETTERING',
 'NORTHAMPTON',
 'SOUTH BUCKS',
 'SOUTH NORTHAMPTONSHIRE',
 'WELLINGBOROUGH',
 'WYCOMBE'}
'laua_name values only in 2023:'
{'BUCKINGHAMSHIRE', 'NORTH NORTHAMPTONSHIRE', 'WEST NORTHAMPTONSHIRE'}
```

In [51]:
```
# Find 'laua_name' values in 2020 but not in 2021
laua_names_only_in_2020 = set(df2020['laua_name']) - set(df2021['laua_nam

# Display the results
display("laua_name values only in 2020:", laua_names_only_in_2020)

# Find 'laua_name' values in 2021 but not in 2020
laua_names_only_in_2021 = set(df2021['laua_name']) - set(df2020['laua_nam

# Display the results
display("laua_name values only in 2023:", laua_names_only_in_2021)
```

```
'laua_name values only in 2020:'
{'CORBY',
 'DAVENTRY',
 'EAST NORTHAMPTONSHIRE',
 'KETTERING',
 'NORTHAMPTON',
 'SOUTH NORTHAMPTONSHIRE',
 'WELLINGBOROUGH'}
'laua_name values only in 2023:'
{'NORTH NORTHAMPTONSHIRE', 'WEST NORTHAMPTONSHIRE'}
```

Some values in laua_name are missing. However, in storing data in MongoDB, advantage was taken of its flexible structure, allowing documents to have different fields. Therefore, filling in missing values with null was not chosen.

Store the data

In [52]:
```
MONGO_CONNECTION_STRING = f"mongodb://localhost:27017/"
print(f"MONGO_CONNECTION_STRING = {MONGO_CONNECTION_STRING}")
```

```
MONGO_CONNECTION_STRING = mongodb://localhost:27017/
```

In [53]:
```
from pymongo import MongoClient
mongo_client = MongoClient(MONGO_CONNECTION_STRING)
DB_NAME = "Q1_TMA02_TM351"
print(f"DB_NAME = {DB_NAME}")

mongo_db = mongo_client[DB_NAME]
```

```
DB_NAME = Q1_TMA02_TM351
```

In [54]:
```
import os
from pymongo import MongoClient


# List of DataFrames and corresponding years
dataframes = [df2019, df2020, df2021, df2022, df2023]
years = [2019, 2020, 2021, 2022, 2023]

for df, year in zip(dataframes, years):
    # Drop the 'year' column if it already exists
    if 'year' in df.columns:
        df = df.drop(columns=['year'])

    # Add a field for the year
    df['year'] = year

    # Convert DataFrame to list of dictionaries
    records = df.to_dict(orient='records')

    # Specify the collection name based on the year
    collection_name = f'data_{year}'

    # Insert data into MongoDB
    mongo_db[collection_name].insert_many(records)
```

```
/tmp/ipykernel_412/1316622539.py:18: UserWarning: DataFrame columns are no
t unique, some columns will be omitted.
  records = df.to_dict(orient='records')
```

As you prepared the data for entry into mongoDB you would find that the dataset contained dirty and missing data. Give three examples where you identified problems with the data. Explain how you resolved these problems and what the implications might be when you analyse the data.

(5 marks)

Write your answer in this markdown cell

In the process of preparing data for entry into MongoDB, three instances of data quality issues were identified.

Firstly, within each CSV dataset representing a specific year, the 'year' information was absent. To address this, the integration process ensured the addition of 'year' information.

Secondly, there are missing values. While columns related to 'Gigabit' were present from 2020 to 2023, there was a gap in the information for the year 2019. Regarding

the 'laua_name' column, some local authorities are missing in each dataframe. However, in storing data in MongoDB, advantage was taken of its flexible structure, allowing documents to have different fields. Therefore, filling in missing values with null was not chosen. This decision aligns with MongoDB's ability to handle sparse data effectively, providing flexibility for future changes in the data structure without disrupting existing documents.

Thirdly, a discrepancy was observed in the column names for 'SFBB' and 'UFBB' between 2019 and the subsequent years (2020-2023). To establish consistency, the column names in 2019 were changed to match those from 2020 to 2023.