DALT7002 Data Science Foundations

Name: Ashwini Pant

Student Id: 19212510

Word Count: 2500

Table of Contents

Table of Figures

1 Introduction

In this coursework, I have taken the three dataset Housing, Broadband and Counsil Tax in different areas of Oxfordshire. These are original data published by UK government. Using these datasets, I have created a data model and I have analysed data. The housing data set contains information about different house prices of different ward of a different district, broadband dataset contains information about broadband speed and similarly, counsil tax dataset contains information about counsil tax charges, area, and band.

In this coursework, I have provided detail information of all the different processes involved in designing, development, implementation and testing of the application system.

2 Data Selection and cleaning

Data Cleaning is a form of data a management. Data selection and cleaning is the very essential step to move forward to normalization. It will remove error data taken from multiple sources. If the data are cleaned it can give high quality information and can increase the productivity.

2.1 For Housing dataset

I have searched and obtained the housing dataset from the following website (North, 16 December 2022).

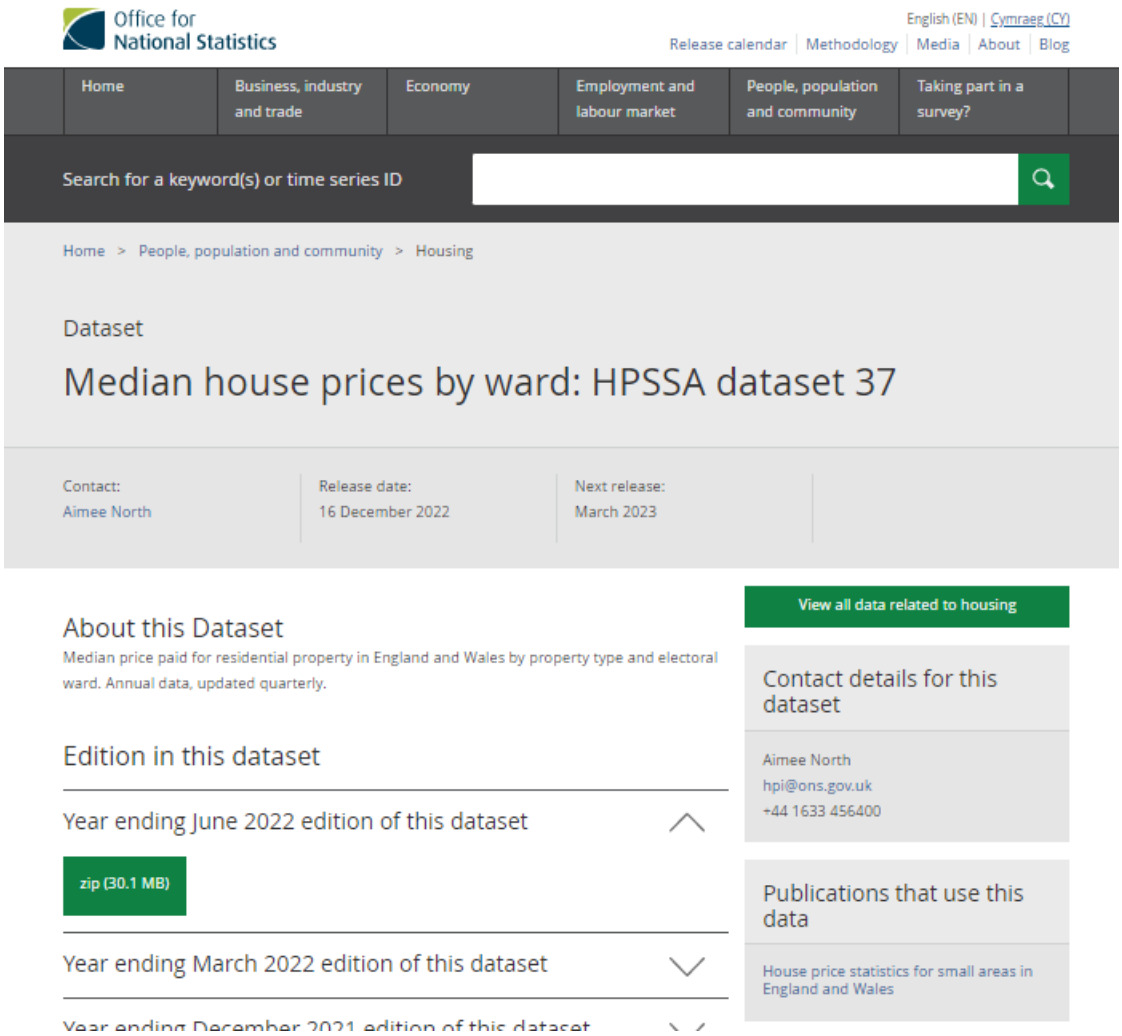
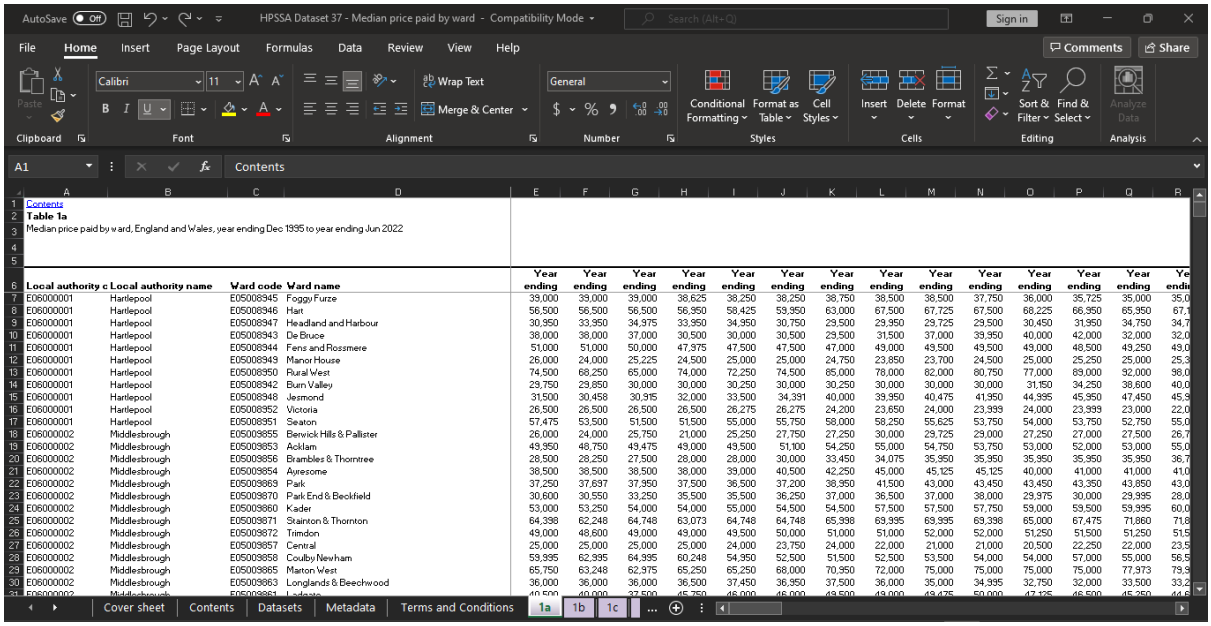


Figure 1: Website from where the housing dataset are collected.

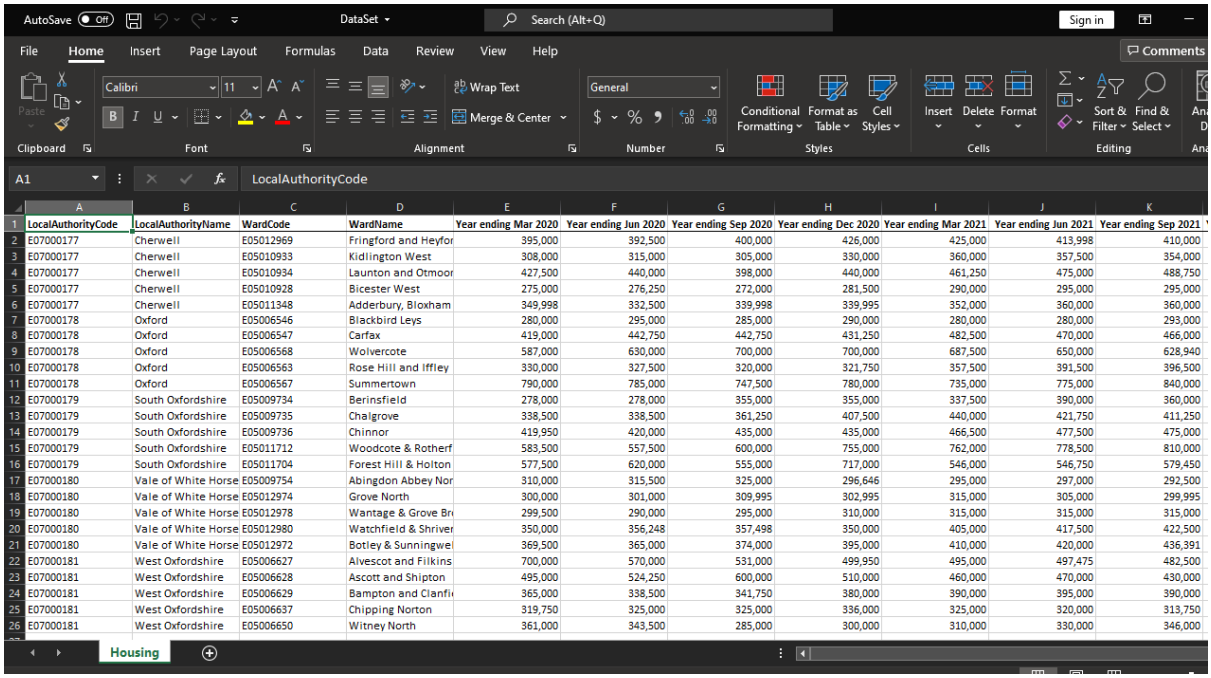
After downloading the dataset folder from the website, I extracted the folder then I open the excel file. There were tons of data which are as follows:



Contents																	
Table 1a																	
Median price paid by ward, England and Wales, year ending Dec 1995 to year ending Jun 2022																	
				Year ending	Year ending	Year ending	Year ending	Year ending	Year ending	Year ending	Year ending	Year ending	Year ending	Year ending	Year ending	Year ending	Ye
Local authority c	Local authority name	Ward code	Ward name														
E06000001	Hartlepool	E05008945	Foggy Furze	39,000	39,000	39,000	38,625	38,250	38,250	38,750	38,500	38,500	37,750	36,000	35,725	35,000	35,0
E06000001	Hartlepool	E05008946	Hart	56,500	56,500	56,500	56,950	58,425	59,950	63,000	67,500	67,725	67,500	68,225	66,950	65,950	67,1
E06000001	Hartlepool	E05008947	Headland and Harbour	30,950	33,950	34,975	33,950	34,950	30,750	29,500	29,950	29,725	29,500	30,450	31,950	34,750	34,7
E06000001	Hartlepool	E05008943	De Bruce	38,000	38,000	37,000	30,500	30,000	30,500	29,500	31,500	37,000	39,950	40,000	42,000	32,000	32,0
E06000001	Hartlepool	E05008944	Fens and Rossmere	51,000	51,000	50,000	47,375	47,500	47,500	47,000	49,000	49,500	49,500	49,000	48,500	49,250	49,0
E06000001	Hartlepool	E05008949	Manor House	26,000	24,000	25,225	24,500	25,000	25,000	24,750	23,850	23,700	24,500	25,000	25,250	25,000	25,3
E06000001	Hartlepool	E05008950	Rural West	74,500	68,250	65,000	74,000	72,250	74,500	65,000	78,000	82,000	80,750	77,000	69,000	92,000	98,0
E06000001	Hartlepool	E05008942	Burn Valley	29,750	29,850	30,000	30,000	30,250	30,000	30,250	30,000	30,000	30,000	31,150	34,250	38,600	40,0
E06000001	Hartlepool	E05008948	Jesmond	31,500	30,458	30,915	32,000	33,500	34,391	40,000	39,950	40,475	41,950	44,995	45,950	47,450	45,9
E06000001	Hartlepool	E05008952	Victoria	26,500	26,500	26,500	26,500	26,275	26,275	24,200	23,650	24,000	23,999	24,000	23,999	23,000	22,0
E06000001	Hartlepool	E05008951	Seaton	57,475	53,500	51,500	51,500	55,000	55,750	58,000	58,250	55,625	53,750	54,000	53,750	52,750	55,0
E06000002	Middlesbrough	E05009855	Berwick Hills & Pallister	26,000	24,000	25,750	21,000	25,250	27,750	27,250	30,000	29,725	29,000	27,250	27,000	27,500	26,7
E06000002	Middlesbrough	E05009853	Acklam	49,950	48,750	49,475	49,000	49,500	51,100	54,250	55,000	54,750	53,750	53,000	52,000	53,000	55,0
E06000002	Middlesbrough	E05009856	Brambles & Thorntree	28,500	28,250	27,500	28,000	28,000	30,000	33,450	34,075	35,950	35,950	35,950	35,950	35,950	36,7
E06000002	Middlesbrough	E05009854	Ayresome	38,500	38,500	38,500	38,000	39,000	40,500	42,250	45,000	45,125	45,125	40,000	41,000	41,000	41,0
E06000002	Middlesbrough	E05009869	Park	37,250	37,697	37,950	37,500	36,500	37,200	38,950	41,500	43,000	43,450	43,450	43,350	43,850	43,0
E06000002	Middlesbrough	E05009870	Park End & Beckfield	30,600	30,550	33,250	35,500	35,500	36,250	37,000	36,500	37,000	38,000	29,975	30,000	29,995	28,0
E06000002	Middlesbrough	E05009860	Kader	53,000	53,250	54,000	54,000	55,000	54,500	54,500	57,500	57,500	57,750	59,000	59,500	59,995	60,0
E06000002	Middlesbrough	E05009871	Stainton & Thornton	64,398	62,248	64,748	63,073	64,748	64,748	65,998	69,995	69,995	69,398	65,000	67,475	71,860	71,8
E06000002	Middlesbrough	E05009872	Trimdon	49,000	48,600	49,000	49,000	49,500	50,000	51,000	51,000	52,000	52,000	51,250	51,500	51,250	51,5
E06000002	Middlesbrough	E05009857	Central	25,000	25,000	25,000	25,000	24,000	23,750	24,000	22,000	21,000	21,000	20,500	22,250	22,000	23,5
E06000002	Middlesbrough	E05009858	Coulby Newham	59,995	62,995	64,995	60,248	54,950	52,500	51,500	52,500	53,500	54,000	54,000	57,000	55,000	56,5
E06000002	Middlesbrough	E05009865	Marton West	65,750	63,248	62,975	65,250	65,250	68,000	70,950	72,000	75,000	75,000	75,000	75,000	77,973	79,9
E06000002	Middlesbrough	E05009863	Longlands & Beechwood	36,000	36,000	36,000	36,500	37,450	36,350	37,500	36,000	35,000	34,995	32,750	32,000	33,500	33,2

Figure 2: Un-cleansed data from the housing dataset.

After that I used excel function filter to filter inaccurate, incomplete, and irrelevant data, then I used to remove duplicate function to remove duplicate data. After cleaning the data, now these data can be used for increasing productivity and the clean data will give quality information for decision making. Clean data can be seen below.



LocalAuthorityCode	LocalAuthorityName	WardCode	WardName	Year ending Mar 2020	Year ending Jun 2020	Year ending Sep 2020	Year ending Dec 2020	Year ending Mar 2021	Year ending Jun 2021	Year ending Sep 2021
E07000177	Cherwell	E05012969	Fringford and Heyfor	395,000	392,500	400,000	426,000	425,000	413,998	410,000
E07000177	Cherwell	E05010933	Kidlington West	308,000	315,000	305,000	330,000	360,000	357,500	354,000
E07000177	Cherwell	E05010934	Launton and Otmoor	427,500	440,000	398,000	440,000	461,250	475,000	488,750
E07000177	Cherwell	E05010928	Bicester West	275,000	276,250	272,000	281,500	290,000	295,000	295,000
E07000177	Cherwell	E05011348	Adderbury, Bloxham	349,998	332,500	339,998	339,995	352,000	360,000	360,000
E07000178	Oxford	E05006546	Blackbird Leys	280,000	295,000	285,000	290,000	280,000	280,000	293,000
E07000178	Oxford	E05006547	Carfax	419,000	442,750	442,750	431,250	482,500	470,000	466,000
E07000178	Oxford	E05006568	Wolvercote	587,000	630,000	700,000	700,000	687,500	650,000	628,940
E07000178	Oxford	E05006563	Rose Hill and Iffley	330,000	327,500	320,000	321,750	357,500	391,500	396,500
E07000178	Oxford	E05006567	Summertown	790,000	785,000	747,500	780,000	735,000	775,000	840,000
E07000179	South Oxfordshire	E05009734	Berinsfield	278,000	278,000	355,000	355,000	337,500	390,000	360,000
E07000179	South Oxfordshire	E05009735	Chalgrove	338,500	338,500	361,250	407,500	440,000	421,750	411,250
E07000179	South Oxfordshire	E05009736	Chinnor	419,950	420,000	435,000	435,000	466,500	477,500	475,000
E07000179	South Oxfordshire	E05011712	Woodcote & Rotherf	583,500	557,500	600,000	755,000	762,000	778,500	810,000
E07000179	South Oxfordshire	E05011704	Forest Hill & Holton	577,500	620,000	555,000	717,000	546,000	546,750	579,450
E07000180	Vale of White Horse	E05009754	Abingdon Abbey Nor	310,000	315,500	325,000	296,646	295,000	297,000	292,500
E07000180	Vale of White Horse	E05012974	Grove North	300,000	301,000	309,995	302,995	315,000	305,000	299,995
E07000180	Vale of White Horse	E05012978	Wantage & Grove Br	299,500	290,000	295,000	310,000	315,000	315,000	315,000
E07000180	Vale of White Horse	E05012980	Watchfield & Shriver	350,000	356,248	357,498	350,000	405,000	417,500	422,500
E07000180	Vale of White Horse	E05012972	Botley & Sunningwel	369,500	365,000	374,000	395,000	410,000	420,000	436,391
E07000181	West Oxfordshire	E05006627	Alvescot and Filkins	700,000	570,000	531,000	499,950	495,000	497,475	482,500
E07000181	West Oxfordshire	E05006628	Ascott and Shipton	495,000	524,250	600,000	510,000	460,000	470,000	430,000
E07000181	West Oxfordshire	E05006629	Bampton and Clanfi	365,000	338,500	341,750	380,000	390,000	395,000	390,000
E07000181	West Oxfordshire	E05006637	Chipping Norton	319,750	325,000	325,000	336,000	325,000	320,000	313,750
E07000181	West Oxfordshire	E05006650	Witney North	361,000	343,500	285,000	300,000	310,000	330,000	346,000

Figure 3: Cleansed housing dataset.

Through these data normalization process will be carried out to maintain data integrity.

2.2 For Broadband dataset

I searched and obtained the broadband data from the following websites (Baker, 21 October, 2022).

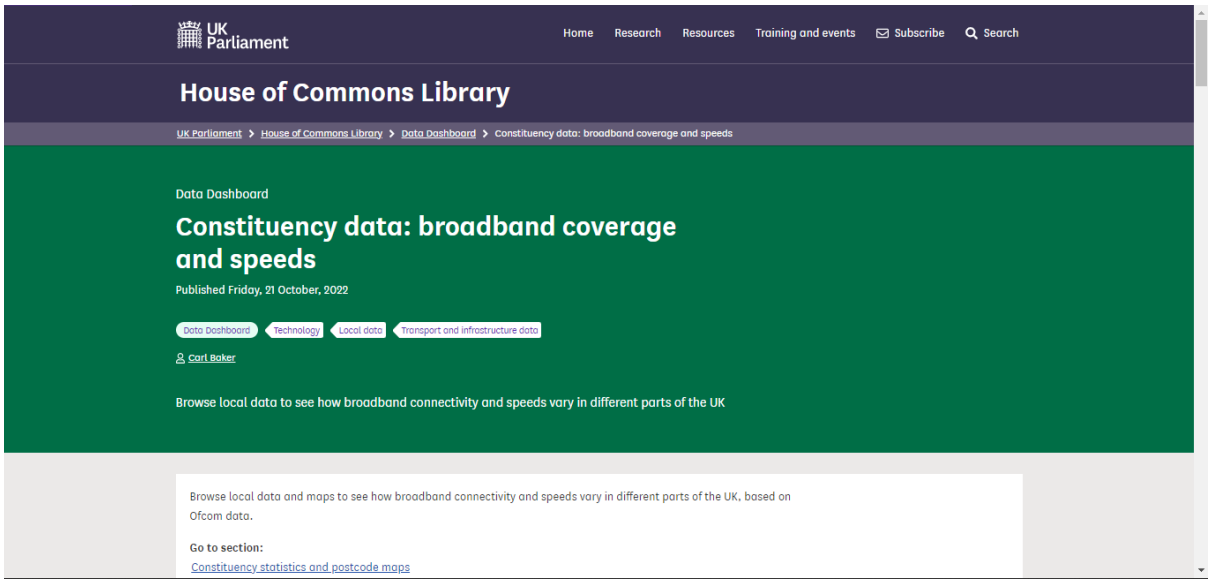
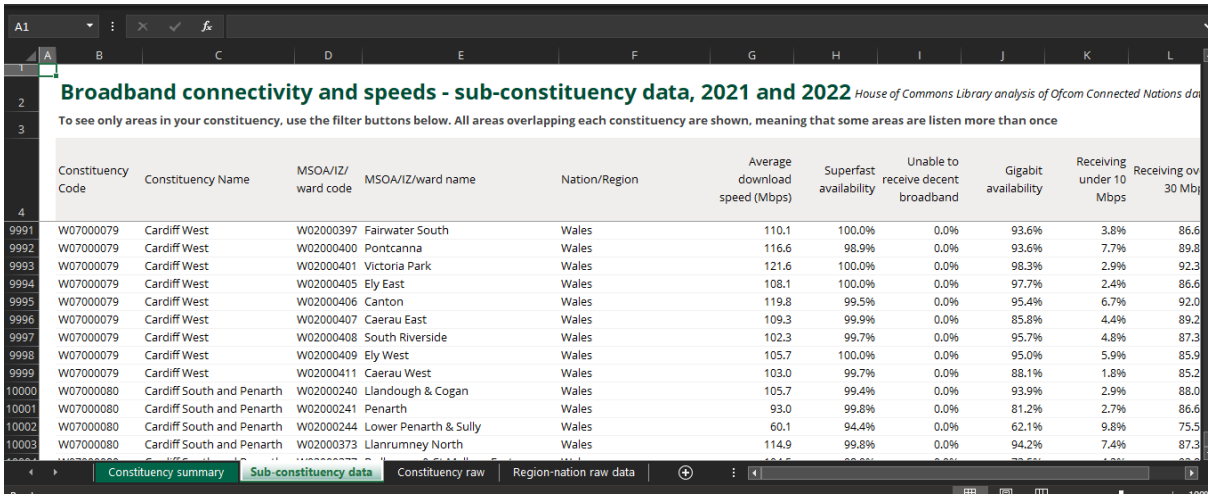


Figure 4 : Website from where the broadband dataset is collected.

After downloading the dataset from the website, I extracted the folder then I open the excel file. There were tons of data which are as follows:



Broadband connectivity and speeds - sub-constituency data, 2021 and 2022 *House of Commons Library analysis of Ofcom Connected Nations da*

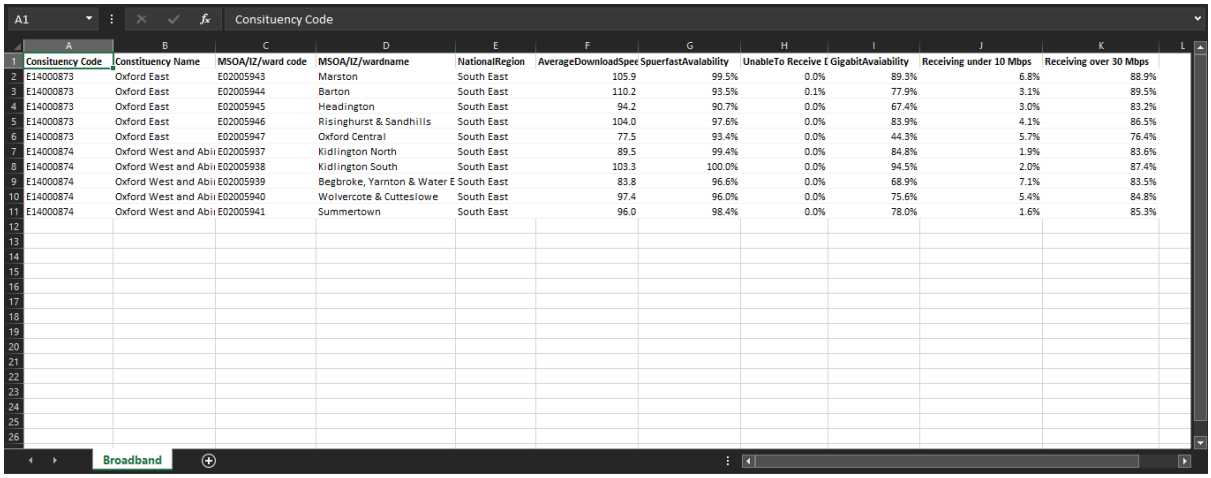
To see only areas in your constituency, use the filter buttons below. All areas overlapping each constituency are shown, meaning that some areas are listen more than once

Constituency Code	Constituency Name	MSOA/IZ/ ward code	MSOA/IZ/ward name	Nation/Region	Average download speed (Mbps)	Superfast availability	Unable to receive decent broadband	Gigabit availability	Receiving under 10 Mbps	Receiving ov 30 Mb
W07000079	Cardiff West	W02000397	Fairwater South	Wales	110.1	100.0%	0.0%	93.6%	3.8%	86.6
W07000079	Cardiff West	W02000400	Pontcanna	Wales	116.6	98.9%	0.0%	93.6%	7.7%	89.8
W07000079	Cardiff West	W02000401	Victoria Park	Wales	121.6	100.0%	0.0%	98.3%	2.9%	92.3
W07000079	Cardiff West	W02000405	Ely East	Wales	108.1	100.0%	0.0%	97.7%	2.4%	86.6
W07000079	Cardiff West	W02000406	Canton	Wales	119.8	99.5%	0.0%	95.4%	6.7%	92.0
W07000079	Cardiff West	W02000407	Caerau East	Wales	109.3	99.9%	0.0%	85.8%	4.4%	89.2
W07000079	Cardiff West	W02000408	South Riverside	Wales	102.3	99.7%	0.0%	95.7%	4.8%	87.3
W07000079	Cardiff West	W02000409	Ely West	Wales	105.7	100.0%	0.0%	95.0%	5.9%	85.9
W07000079	Cardiff West	W02000411	Caerau West	Wales	103.0	99.7%	0.0%	88.1%	1.8%	85.2
W07000080	Cardiff South and Penarth	W02000240	Llandough & Cogan	Wales	105.7	99.4%	0.0%	93.9%	2.9%	88.0
W07000080	Cardiff South and Penarth	W02000241	Penarth	Wales	93.0	99.8%	0.0%	81.2%	2.7%	86.6
W07000080	Cardiff South and Penarth	W02000244	Lower Penarth & Sully	Wales	60.1	94.4%	0.0%	62.1%	9.8%	75.5
W07000080	Cardiff South and Penarth	W02000373	Llanrumney North	Wales	114.9	99.8%	0.0%	94.2%	7.4%	87.3

Constituency summary | Sub-constituency data | Constituency raw | Region-nation raw data

Figure 5: Un-cleansed data from the broadband dataset.

After that I used excel function filter to filter inaccurate, incomplete, and irrelevant data, then I used to remove duplicate function to remove duplicate data. After cleaning the data, now these data can be used for increasing productivity and the clean data will give quality information for decision making. Clean data can be seen below.



Consituency Code	Constituency Name	MSOA/IZ/ward code	MSOA/IZ/wardname	NationalRegion	AverageDownloadSpee	SpuerfastAvalability	UnableTo Receive E	GigabitAvaiability	Receiving under 10 Mbps	Receiving over 30 Mbps
E14000873	Oxford East	E02005943	Marston	South East	105.9	99.5%	0.0%	89.3%	6.8%	88.9%
E14000873	Oxford East	E02005944	Barton	South East	110.2	93.5%	0.1%	77.9%	3.1%	89.5%
E14000873	Oxford East	E02005945	Headington	South East	94.2	90.7%	0.0%	67.4%	3.0%	83.2%
E14000873	Oxford East	E02005946	Risinghurst & Sandhills	South East	104.0	97.6%	0.0%	83.9%	4.1%	86.5%
E14000873	Oxford East	E02005947	Oxford Central	South East	77.5	93.4%	0.0%	44.3%	5.7%	76.4%
E14000874	Oxford West and Abi	E02005937	Kidlington North	South East	89.5	99.4%	0.0%	84.8%	1.9%	83.6%
E14000874	Oxford West and Abi	E02005938	Kidlington South	South East	103.3	100.0%	0.0%	94.5%	2.0%	87.4%
E14000874	Oxford West and Abi	E02005939	Begbroke, Yarnton & Water E	South East	83.8	96.6%	0.0%	68.9%	7.1%	83.5%
E14000874	Oxford West and Abi	E02005940	Wolvercote & Cutteslowe	South East	97.4	96.0%	0.0%	75.6%	5.4%	84.8%
E14000874	Oxford West and Abi	E02005941	Summertown	South East	96.0	98.4%	0.0%	78.0%	1.6%	85.3%

Figure 6: Cleansed broadband dataset.

Through these data normalization process will be carried out to maintain data integrity.

2.3 For CounsilTax dataset

I searched and obtained counsil tax data from the following websites. There were lots of data from which I filtered the data that the proposed system needed (Oxford, 31 December 2022, Council Tax Charges 2022-23).

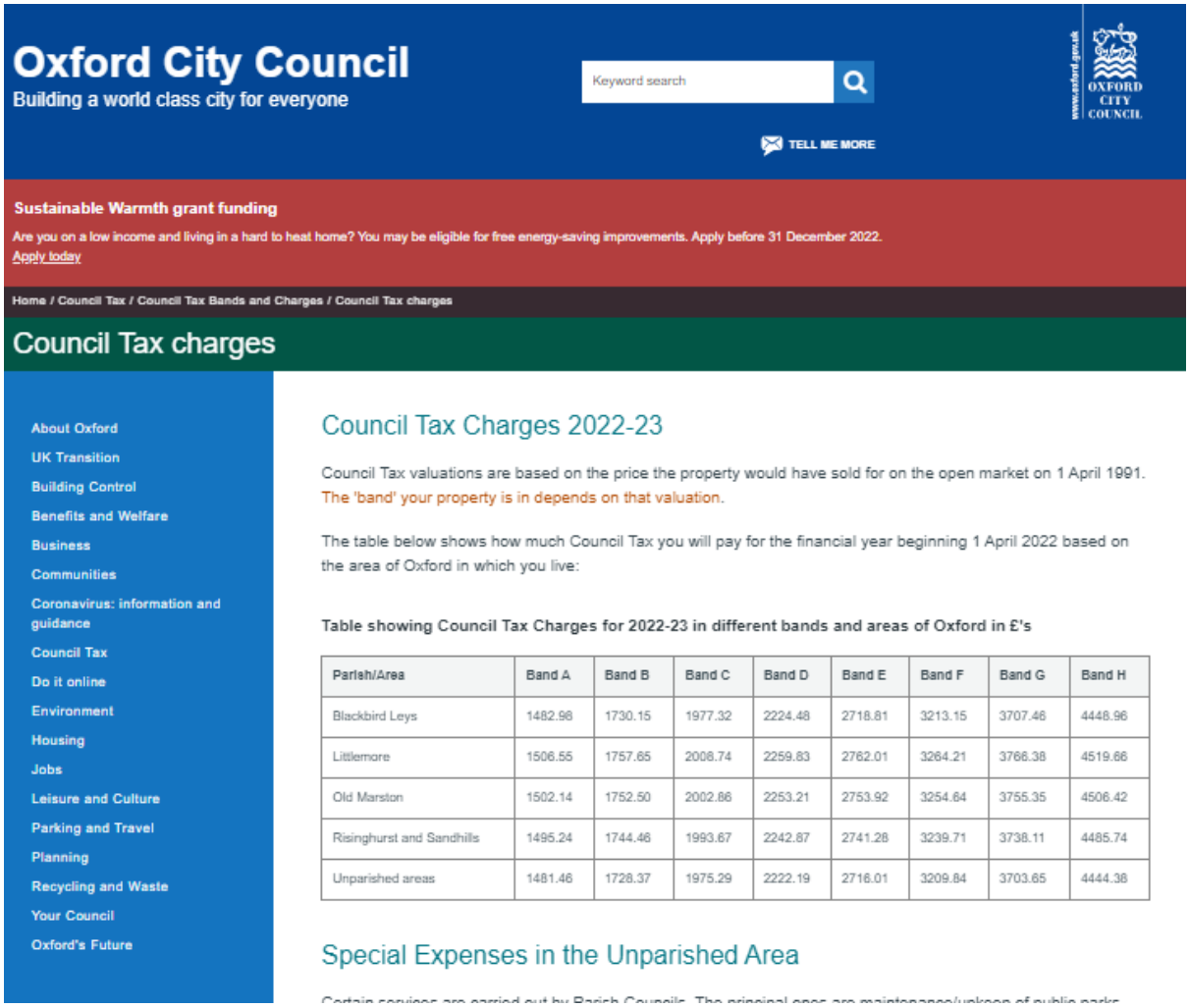


Figure 7 : Website from where the broadband dataset is collected for Oxford City Counsil.

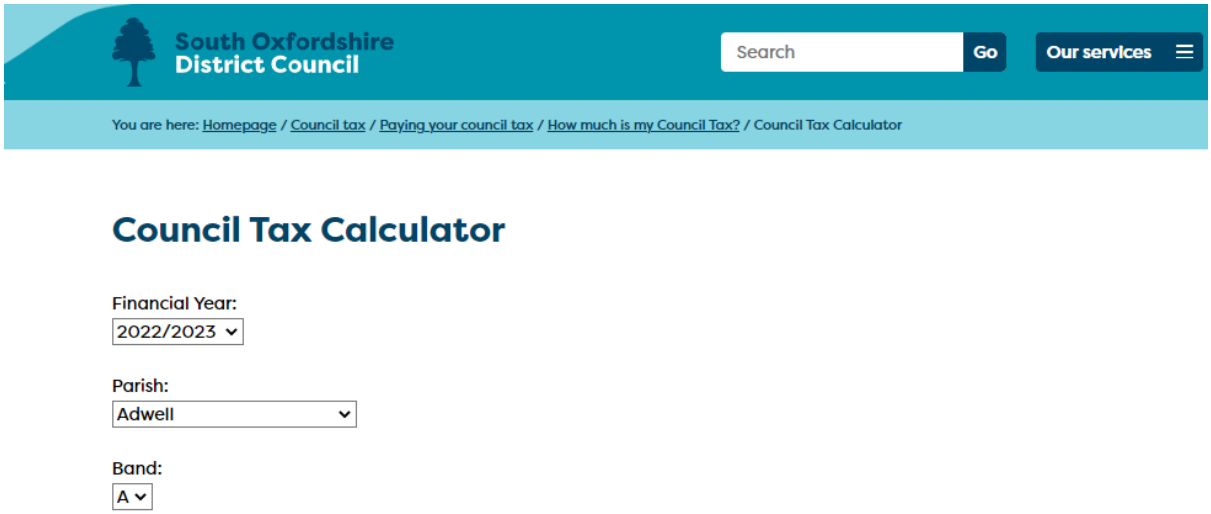


Figure 8: Website from where the broadband dataset is collected for South Oxfordshire District Counsil.

(South, Council Tax Calculator)

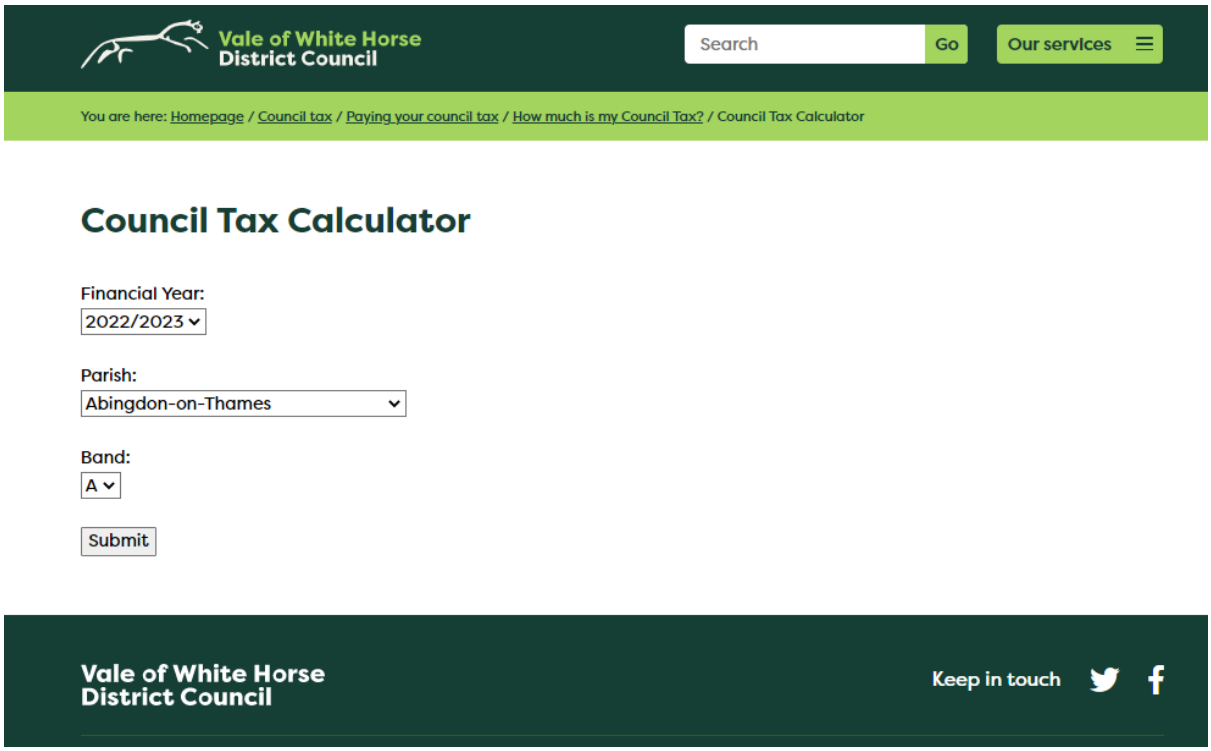


Figure 9: Website from where the broadband dataset is collected for Vale of White Horse District Counsil.

(Vale, 2022, Council Tax Calculator)

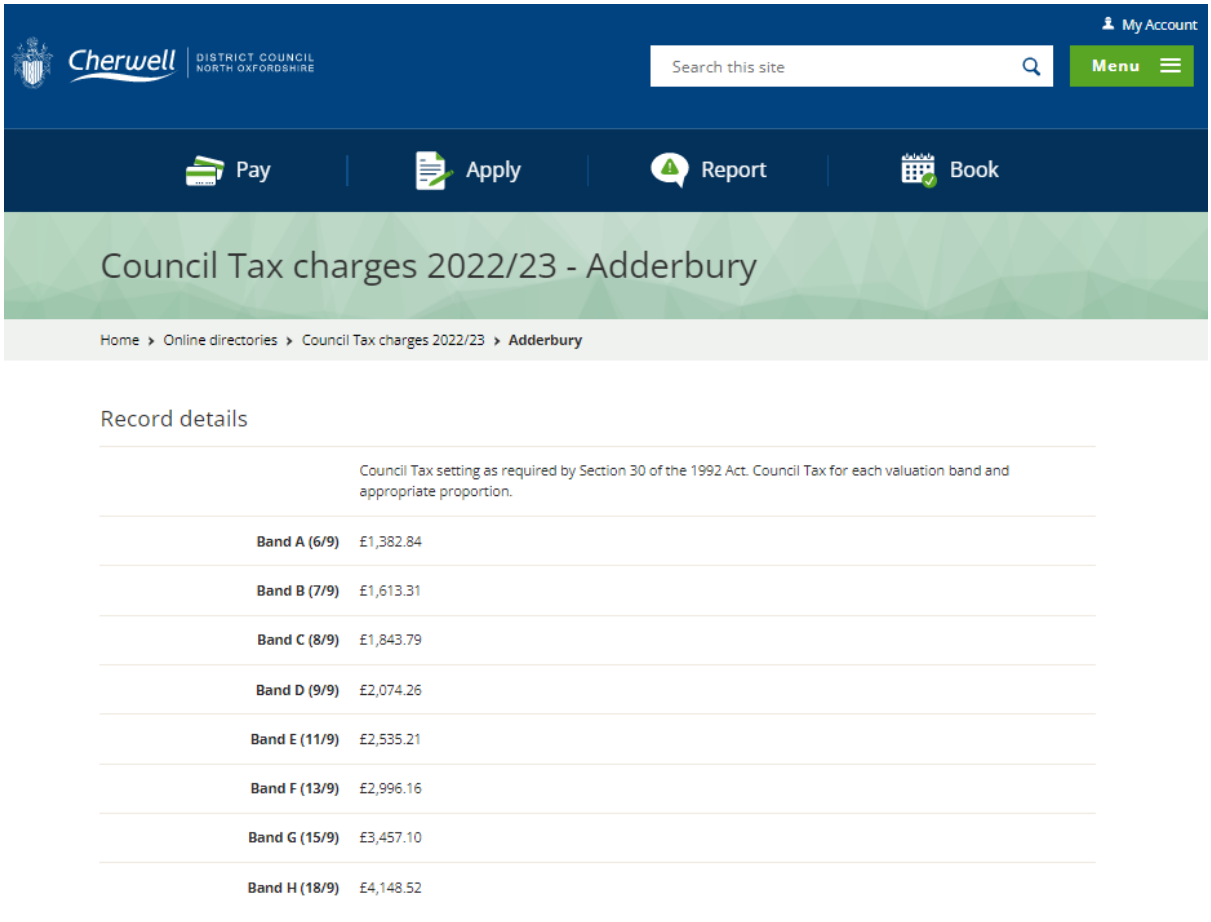


Figure 10: Website from where the broadband dataset is collected for Cherwell District Counsil.

(Cherwell, 2022, Cherwell Council Tax charges 2022/23)

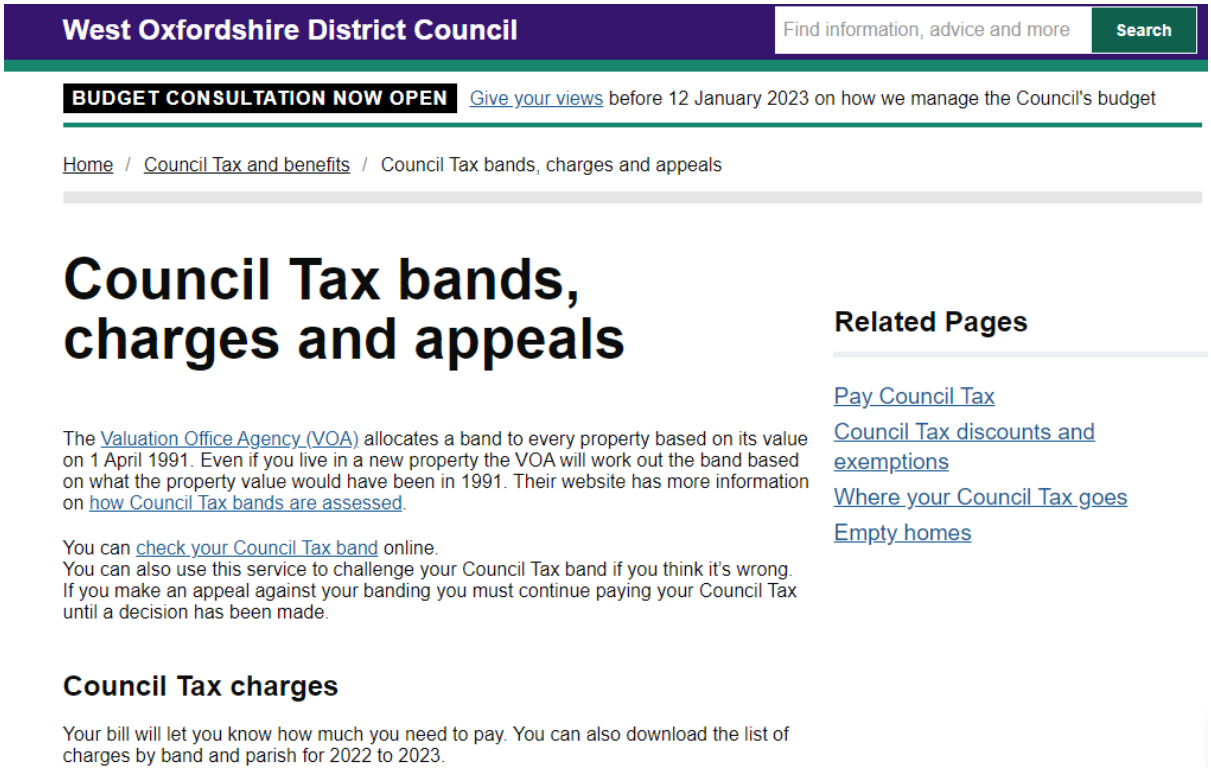
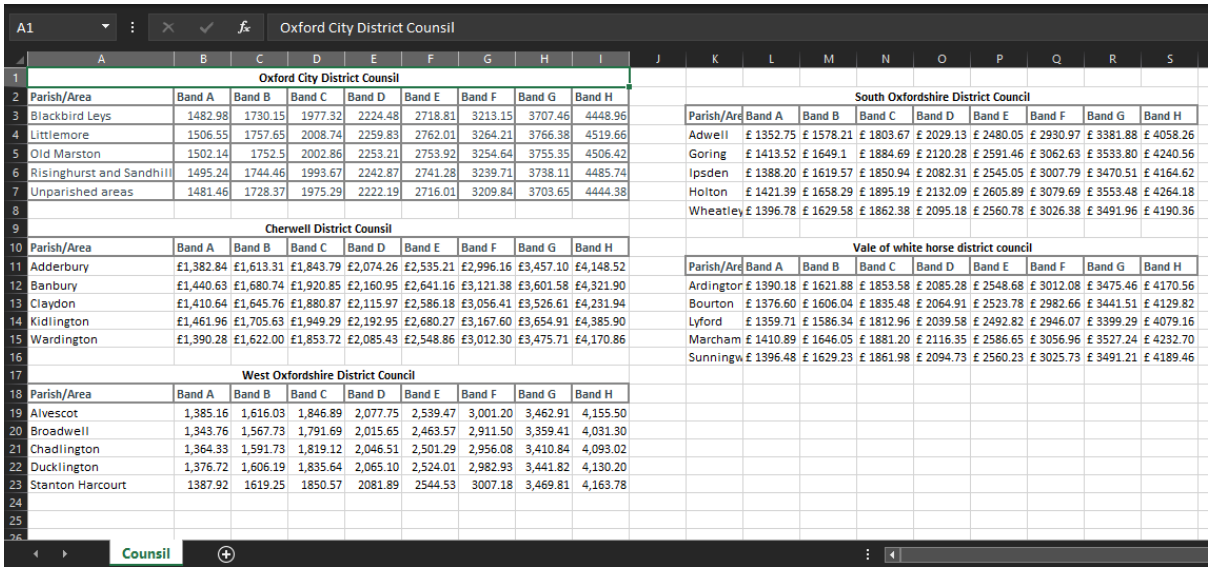


Figure 11 : Website from where the broadband dataset is collected for West Oxfordshire District Counsil.

After data cleaning, the clean data are as follows (West, Council Tax charges).



Oxford City District Counsil

Parish/Area	Band A	Band B	Band C	Band D	Band E	Band F	Band G	Band H
Blackbird Leys	1482.98	1730.15	1977.32	2224.48	2718.81	3213.15	3707.46	4448.96
Littlemore	1506.55	1757.65	2008.74	2259.83	2762.01	3264.21	3766.38	4519.66
Old Marston	1502.14	1752.5	2002.86	2253.21	2753.92	3254.64	3755.35	4506.42
Risinghurst and Sandhill	1495.24	1744.46	1993.67	2242.87	2741.28	3239.71	3738.11	4485.74
Unparished areas	1481.46	1728.37	1975.29	2222.19	2716.01	3209.84	3703.65	4444.38

Cherwell District Counsil

Parish/Area	Band A	Band B	Band C	Band D	Band E	Band F	Band G	Band H
Adderbury	£1,382.84	£1,613.31	£1,843.79	£2,074.26	£2,535.21	£2,996.16	£3,457.10	£4,148.52
Banbury	£1,440.63	£1,680.74	£1,920.85	£2,160.95	£2,641.16	£3,121.38	£3,601.58	£4,321.90
Claydon	£1,410.64	£1,645.76	£1,880.87	£2,115.97	£2,586.18	£3,056.41	£3,526.61	£4,231.94
Kidlington	£1,461.96	£1,705.63	£1,949.29	£2,192.95	£2,680.27	£3,167.60	£3,654.91	£4,385.90
Wardington	£1,390.28	£1,622.00	£1,853.72	£2,085.43	£2,548.86	£3,012.30	£3,475.71	£4,170.86

West Oxfordshire District Council

Parish/Area	Band A	Band B	Band C	Band D	Band E	Band F	Band G	Band H
Alvescot	1,385.16	1,616.03	1,846.89	2,077.75	2,539.47	3,001.20	3,462.91	4,155.50
Broadwell	1,343.76	1,567.73	1,791.69	2,015.65	2,463.57	2,911.50	3,359.41	4,031.30
Chadlington	1,364.33	1,591.73	1,819.12	2,046.51	2,501.29	2,956.08	3,410.84	4,093.02
Ducklington	1,376.72	1,606.19	1,835.64	2,065.10	2,524.01	2,982.93	3,441.82	4,130.20
Stanton Harcourt	1387.92	1619.25	1850.57	2081.89	2544.53	3007.18	3,469.81	4,163.78

South Oxfordshire District Council

Parish/Are	Band A	Band B	Band C	Band D	Band E	Band F	Band G	Band H
Adwell	£ 1352.75	£ 1578.21	£ 1803.67	£ 2029.13	£ 2480.05	£ 2930.97	£ 3381.88	£ 4058.26
Goring	£ 1413.52	£ 1649.1	£ 1884.69	£ 2120.28	£ 2591.46	£ 3062.63	£ 3533.80	£ 4240.56
Ipsden	£ 1388.20	£ 1619.57	£ 1850.94	£ 2082.31	£ 2545.05	£ 3007.79	£ 3470.51	£ 4164.62
Holton	£ 1421.39	£ 1658.29	£ 1895.19	£ 2132.09	£ 2605.89	£ 3079.69	£ 3553.48	£ 4264.18
Wheatley	£ 1396.78	£ 1629.58	£ 1862.38	£ 2095.18	£ 2560.78	£ 3026.38	£ 3491.96	£ 4190.36

Vale of white horse district council

Parish/Are	Band A	Band B	Band C	Band D	Band E	Band F	Band G	Band H
Ardington	£ 1390.18	£ 1621.88	£ 1853.58	£ 2085.28	£ 2548.68	£ 3012.08	£ 3475.46	£ 4170.56
Bourton	£ 1376.60	£ 1606.04	£ 1835.48	£ 2064.91	£ 2523.78	£ 2982.66	£ 3441.51	£ 4129.82
Lyford	£ 1359.71	£ 1586.34	£ 1812.96	£ 2039.58	£ 2492.82	£ 2946.07	£ 3399.29	£ 4079.16
Marcham	£ 1410.89	£ 1646.05	£ 1881.20	£ 2116.35	£ 2586.65	£ 3056.96	£ 3527.24	£ 4232.70
Sunningw	£ 1396.48	£ 1629.23	£ 1861.98	£ 2094.73	£ 2560.23	£ 3025.73	£ 3491.21	£ 4189.46

Figure 12: Cleansed counsil tax dataset.

Through these data normalization process will be carried out to maintain data integrity.

3 Legal and ethical issues

Accessing the shared data in net might be helpful for many people. But people should be aware of some law to use such data. Some laws are:

- legislation that applies to your proposed data use
- how to produce statistics
- data protection by design
- data minimisation
- information governance

When re-using published and unpublished information relating to public tasks, you must follow the Re-use of Public Sector Information Regulations 2015 (GOV.UK, Data Ethics Framework).

4 Structured Data Model and Semi-Structured Data Model

Structured Data Model

Structured data are highly organized and quantitative data. These are the data that we mostly used to work with. It is the data that neatly fit with the rows and column in sql database and excel. Examples of structured data includes name, address, date, contact number and more. This data is easily understood by machine language and hence straight forward to analyse. Structured data rely on the existence of a data model (Endenicher, Structured Data). A model of how data can be stored, processed, and accessed. Each column is distinct and can be accessed jointly or separately along with data from other fields location in the database. Using structured data in relational database we can input, manipulate, and search. This feature makes structured data more powerful. To manage structured data the programming language is called structured query language known as SQL.

Semi-Structured Data Model

Semi-Structured Data are those data which are semi organized or formatted in a conventional way. It doesn't follow format of a relational database or tabular data model. However, it contains some structural elements such as organizational semantic tags and metadata that make it easier to analyse and these data are not completely raw or unstructured. Compare to structured data it is simpler and more flexible to scale. Semi-structured data formats consist of JSON, Avro, and XML.Examples, when you see smart phone video or photo, which it contains semi-structured data (Teradata, 2022, Semi-Structured).

Hence, I have already mentioned above what are structured data and semi-structured data. Seeing all the character of both the data model. I think structured data model (SQL) fits perfectly for the proposed system. Because to get the desired result for the system. I have cleansed the data to get structured data. In the proposed system I am using relational database which required structured data. Structured data includes name, address, date, contact number and more. In the proposed system, I have created a similar table, column which required structured data. I need structured data in my proposed system because I must analyse the system requirement.

5 Data model and implementation

5.1 Normalization

Normalization is a multi-step process of removing duplicated from relational tables to enhance data and eliminate data redundancy and to maintain data integrity in the table.

5.1.1 Normalization for Housing

5.1.1.1 Un-Normalize Form (UNF)

LocalAuthority **({**LocalAuthorityCode, LocalAuthorityName**, {**WardCode, WardName}}**,** Quarter, Year, Month, Price)

Assumptions: The attribute from the houses price data set has been listed and repeating group has been highlighted using curly bracket. I have added LocalAuthorityCode as a Unique Identifier.

Repeating groups: ({LocalAuthorityCode, LocalAuthorityName**, {**WardCode, WardName}}

5.1.1.2 First Normal Form (1NF)

LocalAuthority (LocalAuthorityCode, LocalAuthorityName)

Ward (WardCode, Ward Name, LocalAuthorityCode*)

Quarter (QuarterCode, Year, Month, Price, WardCode*)

Assumptions:

In the Local Authority table, LocalAuthorityCode is declared as a primary key.

In the ward table, WardCode is declared as a primary key and LocalAuthorityCode is added a foreign key to form a relationship with local authority table which is formed after removing the repeating group.

In quarter table, QuarterCode is added as a primary key and WardCode is added a foreign key to form a relationship with ward table which is formed after removing the repeating group.

Dependencies in First Normal Form Entities

Here are the dependencies identified in the First Normal Form Entities:

Full Functional Dependencies in Ward

WardCode, LocalAuthorityCode* → WardName

Partial Dependencies in Quarter

QuarterCode → Year, Month, Price

5.1.1.3 Second Normal Form (2NF)

Removing the partial dependencies, we get the following entities in second normal form.

LocalAuthority (LocalAuthorityCode, LocalAuthorityName)

Ward (WardCode, WardName, LocalAuthorityCode*)

Quarter (QuarterCode, Year, Month, Price)

WardQuarter (WardQuarterCode, WardCode*, QuarterCode*)

Assumptions:

In the quarter table, after removing partial dependencies QuarterCode is declared as the new primary key.

In the ward quarter table, ward quarter code is added as the primary key and ward code and quarter code are declared as foreign key to form a relationship with ward and quarter table after removing partial dependencies.

Since there is no transitive dependency, so it is already in third normal form.

5.1.2 Normalization for Broadband

5.1.2.1 Un-Normalize Form (UNF)

NationalRegion ({RegionCode, RegionName, {ConstituencyCode, ConstituencyName}}, WardCode, WardName, AverageSpeed, SpuerfastAvalability, UnableToReceiveBroadband, GigabitAvaiability, ReceivingUnder10Mbps, ReceivingOver30Mbps)

Assumptions: The attribute from the broadband data set has been listed and repeating group has been highlighted using curly bracket. I have added RegionCode as a Unique Identifier.

Repeating groups: ({RegionCode, RegionName, {ConstituencyCode, ConstituencyName}}

5.1.2.2 First Normal Form (1NF)

NationalRegion (RegionCode, RegionName)

Constituency (ConstituencyCode, ConstituencyName, RegionCode*)

Ward (WardCode, WardName, BroadbandCode, AverageSpeed, SpuerfastAvalability, UnableToReceiveBroadband, GigabitAvaiability, ReceivingUnder10Mbps, ReceivingOver30Mbps, ConstituencyCode*)

In the ward table, BroadbandCode is added to form a Composite primary Key with WardCode and ConsituencyCode is added a foreign key to form a relationship with Constituency table which is formed after removing the repeating group.

Dependencies in First Normal Form Entities

Here are the dependencies identified in the First Normal Form Entities:

Full Functional Dependencies in Constituency

ConsituencyCode, RegionCode* → ConstituencyName

Full Functional Dependencies in Ward

WardCode, BroadbandCode, ConsituencyCode* → WardName

Partial Dependency in Ward

BroadbandCode - AverageSpeed, SpuerfastAvalability, UnableToReceiveBroadband, GigabitAvaiability, ReceivingUnder10Mbps, ReceivingOver30Mbps

5.1.2.3 Second Normal Form (2NF)

NationalRegion (RegionCode, RegionName)

Constituency (ConsituencyCode, ConstituencyName, RegionCode*)

Ward (WardCode, WardName, ConsituencyCode*, BroadbandCode*)

Broadband (BroadbandCode, AverageSpeed, SpuerfastAvalability, UnableToReceiveBroadband, GigabitAvaiability, ReceivingUnder10Mbps, ReceivingOver30Mbps)

Assumptions:

After removing partial dependencies from ward table WardCode is declared as primary key.

In the broadband table, broadband code is added as the primary key.

Transitive Dependency of Ward

ConsituencyCode*, BroadbandCode* --> WardCode -> WardName

Further,

ConsituencyCode*, BroadbandCode* -> WardCode

WardCode -> WardName

5.1.2.4 Third Normal Form (3NF)

NationalRegion (RegionCode, RegionName)

Constituency (ConsituencyCode, ConstituencyName, RegionCode*)

Broadband (BroadbandCode, AverageSpeed, SpuerfastAvalability, UnableToReceiveBroadband, GigabitAvaiability, ReceivingUnder10Mbps, ReceivingOver30Mbps)

Ward (WardCode, WardName)

WardBroadband (SerialNo, ConsituencyCode*, WardCode*, BroadbandCode*)

5.1.3 Normalization for Counsil Tax

5.1.3.1 Un-Normalize Form (UNF)

Council ({CounsilCode, CounsilName, {AreaCode, Area}}, BandDesc, Price)

Assumptions: The attribute from the counsil tax data set has been listed and repeating group has been highlighted using curly bracket. I have added Counsil Code as a Unique Identifier.

Repeating groups: ({CounsilCode, CounsilName, {AreaCode, Area}}

5.1.3.2 First Normal Form (1NF)

Council (CounsilCode, CounsilName)

Area (AreaCode, Area, CounsilCode*)

Band (BandCode, BandDesc, Price, AreaCode *)

Assumptions:

In the Counsil table, CounsilCode is declared as a primary key.

In the Area table, AreaCode is declared as a primary key and counsil code is added as a foreign key to form a relationship with Area table which is formed after removing the repeating group.

In the Band table, BandCode is added as a primary key and AreaCode is added as a foreign key to form a relationship with Area table which is formed after removing the repeating group.

Dependencies in First Normal Form Entities

Here are the dependencies identified in the First Normal Form Entities:

Full Functional Dependencies in Area

AreaCode, CounsilCode* -> Area

Full Functional Dependencies in Band

BandCode, AreaCode * -> Price

Partial Functional Dependencies in band table

BandCode -> BandDesc

5.1.3.3 Second Normal Form (2NF)

Council (CounsilCode, CounsilName)

Area (AreaCode, Area, CounsilCode*)

Band (BandCode, BandDesc)

AreaBand (AreaBandCode, Price, BandCode*, AreaCode*)

Assumptions:

After removing partial dependencies Area Band Code is added as the new primary key and band code and area code are declared as foreign key to form a relationship with band and area table after removing partial dependencies.

Since there is no transitive dependency, so it is already in third normal form.

5.2 Appropriate design of SQL database tables

5.2.1 ERD for Housing

Entity Relationship Diagram commonly known as ERD shows the relations between the table sets that are stored in the databases.

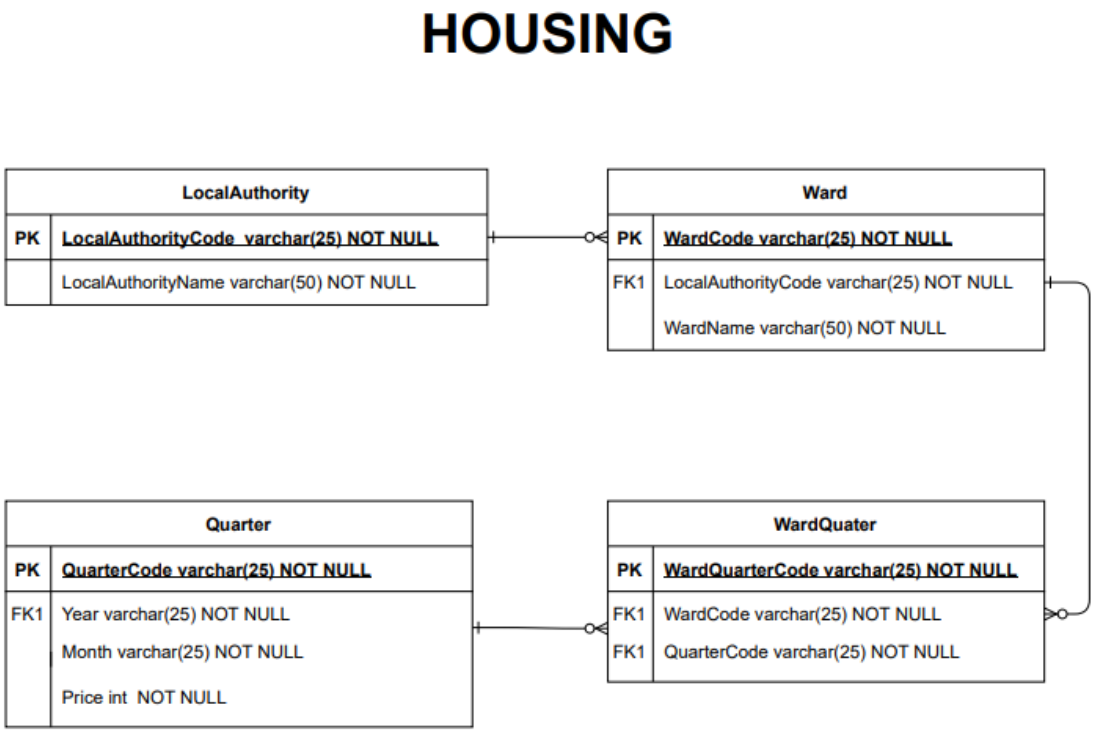


Figure 13 : ERD of Houses price data set.

The above ERD represents the final design of the Housing Database System. This Database System of the housing records all the house prices data per ward of each district quarterly and yearly. As per the ERD, the relationships of this database are as follows:

- The Local Authority holds multiple Ward. So, local authority table has one to many relations with ward table.
- There are number of wards in the local authority and each ward can have one or multiple quarter. Therefore, ward table has one to many relations with ward quarter table.
- Similarly, there are many quarters in a year where house prices may vary according to each ward. Thus, quarter table has one to many relations with ward quarter table.

Definition of keys and data types of Housing

Primary Key: It is a unique key in a table.

Foreign Key: When one ‘A’ table primary key is taken in reference in another ‘B’ table it is known foreign key in that table. It provides link between data in two tables.

Table Name: LocalAuthority

- LocalAuthorityCode column is a primary key in LocalAuthority table having string data type.
- LocalAuthorityName column contains name of the local authority’s having string data type.

Table Name: Ward

- WardCode column is a primary key in Ward table having string data type.
- LocalAuthorityCode column is a foreign key referenced to the primary key of the table name LocalAuthority.
- WardName column contains the name of ward in local authority having string data type.

Table Name: Quarter

- QuaterCode column is a primary key in Quater table having string data type.
- Year column contains year value which has string data type.
- Month column contains the name of month having string data type.
- Price column contains the price value for each ward in a particular month and year having integer data type.

Table Name: WardQuarter

- WardQuarterCode column is a primary key in WardQuarter table having string data type.
- WardCode column is a foreign key referenced to the primary key of the table name Ward.
- QuaterCode column is a foreign key referenced to the primary key of the table name Quarter.

5.2.2 ERD for Broadband

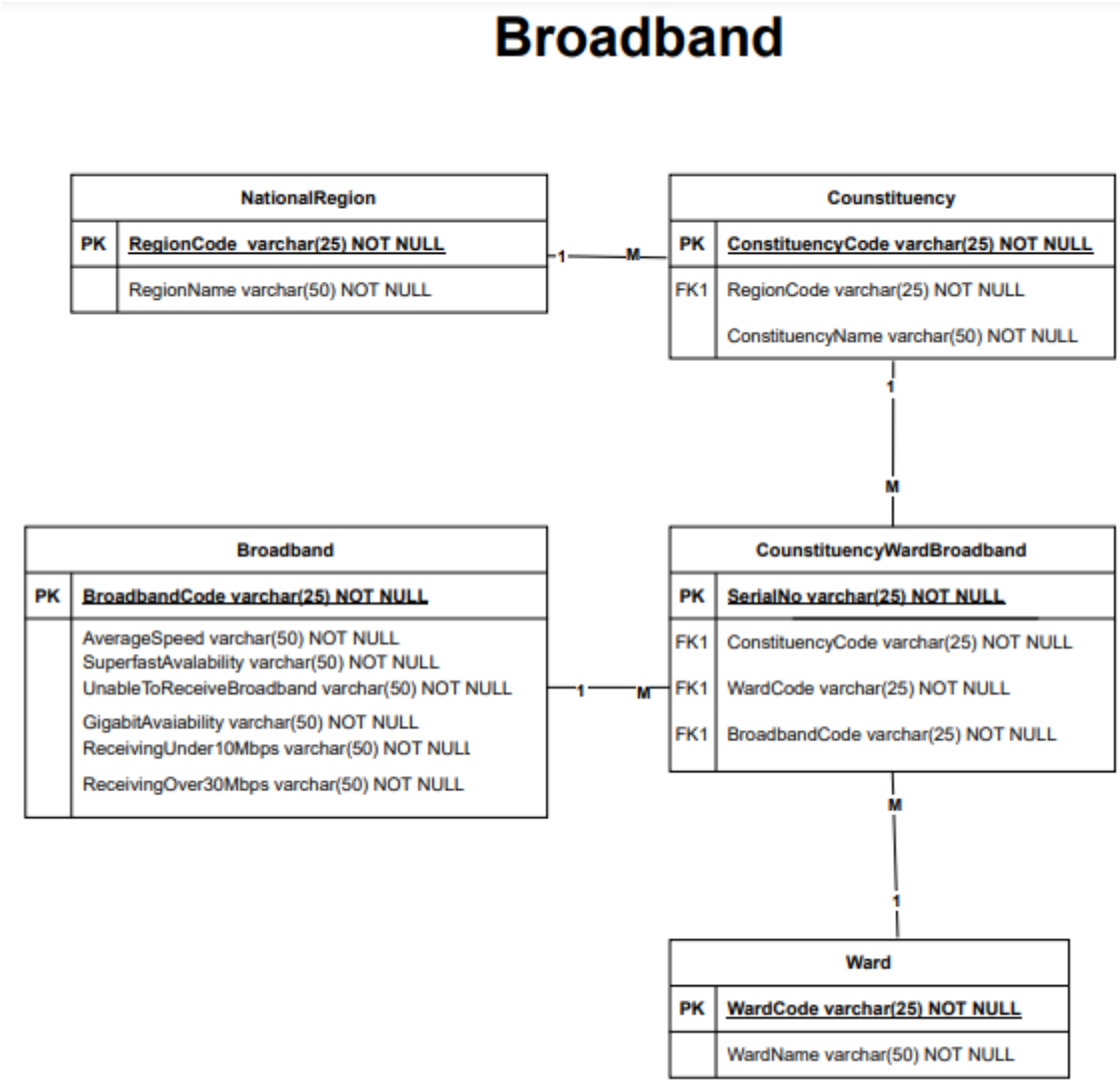


Figure 14 : ERD of Broadband dataset.

The above ERD represents the final design of the Broadband Database System. This Database System of the broadband records all the average download speed, superfast availability, and all other broadband data per ward of each district. As per the ERD, the relationships of this database are as follows:

- The National Region holds multiple Constituency. So, National Region table has one to many relations with ward table.
- There are number of Constituency in the National Region and each Constituency can have one or multiple wards and broadband. Therefore, constituency table has one to many relations with ConstituencyWardBroadband table.
- Similarly, there are number of wards in the Constituency and each ward can have one or multiple constituency and broadband. Therefore, ward table has one to many relations with ConstituencyWardBroadband table.

Definition of keys and data types of Broadband

Table Name: NationalRegion

- RegionCode column is a primary key in NationalRegion table having string data type.
- RegionName column contains name of the region having string data type.

Table Name: Constituency

- ConstituencyCode column is a primary key in Constituency table having string data type.
- RegionCode column is a foreign key referenced to the primary key of the table name NationalRegion.
- ConstituencyName column contains the name of constituency in national region having string data type.

Table Name: Ward

- WardCode column is a primary key in Ward table having string data type.
- WardName column contains name of the ward having string data type.

Table Name: Broadbands

- BroadbandCode column is a primary key in Broadband table having string data type.
- AverageSpeed column contains number of average speed value having string data type.
- SpuerfastAvalability column contains number of superfast availability having string data type.
- UnableToReceiveBroadband column contains number of the unable to receive broadband having string data type.
- GigabitAvalability column contains number of the gigabit avalability having string data type.
- ReceivingUnder10Mbps column contains number of values receiving under 10 mbps having string data type.
- ReceivingUnder30Mbps column contains number of values receiving under 30 mbps having string data type.

Table Name: ConstituencyWardBroadband

- SerialNo column is a primary key in ConstituencyWardBroadband table having string data type.

- RegionCode column is a foreign key referenced to the primary key of the table name NationalRegion.
- ConstituencyName column contains the name of constituency in national region having string data type.

5.2.3 ERD for CounsilTax

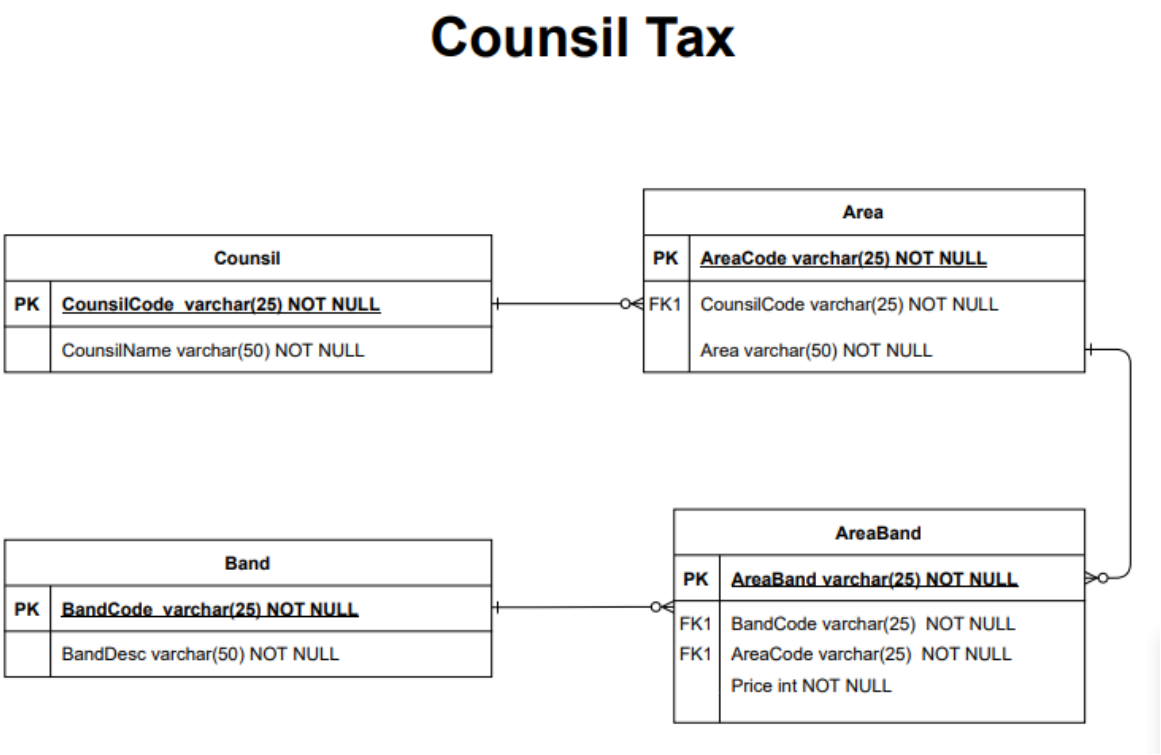


Figure 15: ERD of Counsil Tax.

The above ERD represents the final design of the Counsil Tax Database System. This Database System of the counsil records all the band prices data per area of each counsil from band A to band H. As per the ERD, the relationships of this database are as follows:

- The Counsil holds multiple Area. So, counsil table has one to many relations with area table.
- There are number of areas in the counsil, and each area can have one or multiple bands. Therefore, why area table has one to many relations with AreaBand table.
- Similarly, there are many bands in a area where band prices may vary according to each area. Thus, Band table has one to many relations with AreaBand table.

Definition of keys and data types of CounsilTax

Table Name: Counsil

- CounsilCode column is a primary key in Counsil table having string data type.
- CounsilName column contains name of the counsil having string data type.

Table Name: Area

- AreaCode column is a primary key in Area table having string data type.

- CounsilCode column is a foreign key referenced to the primary key of the table name Counsil.
- AreaName column contains the name of area of a counsil having string data type.

Table Name: Band

- BandCode column is a primary key in Band table having string data type.
- BandName column contains name of the band having string data type.

Table Name: AreaBand

- AreaBand column is a primary key in AreaBand table having string data type.
- AreaCode column is a foreign key referenced to the primary key of the table name Area.
- BandCode column is a foreign key referenced to the primary key of the table name Band.

6 R-Code

6.1 Steps to run code in RStudio

```
1  install.packages("RSQLite")
2  library(RSQLite)
3  setwd("D:/Master/Data Science")
4
5  conn <- dbConnect(SQLite(),"Coursework.db")
6
7  coursework <- dbSendQuery (conn,"Select * from LocalAuthority")
8
9  dbFetch(coursework)
```

Figure 16:Steps to run code in RStudio.

To run sql query in RStudio, I have installed the library RSQLite which uses RSQLite library using following code.

install.packages("RSQLite")

library(RSQLite)

Then I have set the working directory where my databases are stored using following code.

setwd("D:/Master/Data Science")

After that, I have run the code that connects my database to RStudio using dbConnect function using following code.

conn <- dbConnect(SQLite(),"Coursework.db")

Similarly, I have used dbSendQuery function which sends or submit a query to database using conn function using following code.

coursework <- dbSendQuery(conn, "Select * From LocalAuthority")

After that using dbFetch the results are fetched from the database and are displayed using following code.

dbFetch(coursework)

6.2 Design and Execution of R code

6.2.1 For question 3

```
question3 <- dbSendQuery(conn, "Select DISTINCT LocalAuthorityName,WardName,  substr(qa.QuarterCode, -2,2) Quarter,
AveragePrice from Ward w
inner join LocalAuthority la on la.LocalAuthorityCode=w.LocalAuthorityCodeId
inner join WardQuarter wq on wq.WardCodeId = w. WardCode
inner join ( select q.QuarterCode , (q.Price + qq.Price)/2 AveragePrice from Quarter qq
inner join Quarter q on substr(qq.QuarterCode, -4,4)= substr(q.QuarterCode, -4,4)where q.Year <> qq.Year) qa on
wq.QuarterCodeId = qa. QuarterCode")

dbFetch(question3)

```

Figure 17:Design for question 3.

In the figure, by matching local authority id from local authority table and ward table, similarly by matching ward code id from ward quarter and ward table and putting some formula to calculate average price of two years according to quarter in subquery. Since my quarter code is like '*2020FHQ1'* substr function is used to show only '*Q1*'. The average house price is shown for two years according to quarter.

6.2.2 Execution of question 3

	LocalAuthorityName	WardName	Quarter	AveragePrice
1	Cherwell	Fringford and Heyfords	Q1	410000
2	Cherwell	Fringford and Heyfords	Q2	403249
3	Cherwell	Fringford and Heyfords	Q3	405000
4	Cherwell	Fringford and Heyfords	Q4	413000
5	Cherwell	Kidlington West	Q1	334000
6	Cherwell	Kidlington West	Q2	336250
7	Cherwell	Kidlington West	Q3	329500
8	Cherwell	Kidlington West	Q4	339000
9	Cherwell	Launton and Otmoor	Q1	444375
10	Cherwell	Launton and Otmoor	Q2	457500
11	Cherwell	Launton and Otmoor	Q3	443375
12	Cherwell	Launton and Otmoor	Q4	470000
13	Cherwell	Bicester West	Q1	282500
14	Cherwell	Bicester West	Q2	285625
15	Cherwell	Bicester West	Q3	283500
16	Cherwell	Bicester West	Q4	285956
17	Cherwell	Adderbury, Bloxham and Bodicote	Q1	350999
18	Cherwell	Adderbury, Bloxham and Bodicote	Q2	346250
19	Cherwell	Adderbury, Bloxham and Bodicote	Q3	349999
20	Cherwell	Adderbury, Bloxham and Bodicote	Q4	353997
21	Oxford	Blackbird Leys	Q1	280000
22	Oxford	Blackbird Leys	Q2	287500
23	Oxford	Blackbird Leys	Q3	289000
24	Oxford	Blackbird Leys	Q4	291500
25	Oxford	Carfax	Q1	450750
26	Oxford	Carfax	Q2	456375
27	Oxford	Carfax	Q3	454375
28	Oxford	Carfax	Q4	446625
29	Oxford	Wolvercote	Q1	637250
30	Oxford	Wolvercote	Q2	640000
31	Oxford	Wolvercote	Q3	664470

Figure 18: Execution of question 3.

6.2.3 For question 4

```
#this query returns data for an average increase and decrease prices
#in percentage between two years for a given ward in particular district.
question4 <- dbSendQuery(conn, "Select distinct la.LocalAuthorityName,
wa.WardName,c.percentage AveragePercentage
from LocalAuthority la
inner JOIN ward wa
on la.LocalAuthorityCode =wa.LocalAuthorityCodeId
inner JOIN WardQuarter waq on
wa.WardCode = waq.WardCodeId
inner join
(
  select substr(qa.quartercode,-4,3) as quartercode,qa.year,qb.year,
  sum(qa.price),sum(qb.price), (qa.price-qb.price),
  (((qa.price-qb.price)*100)/(qa.price))||'%'  as percentage
  from quarter qa inner join
  quarter qb on substr(qa.quartercode, -4,3) = substr(qb.quartercode,-4,3)

  where qa.year = '2020' and qb.year = '2021'
  group by substr(qa.quartercode,-4,3), qa.year,qb.year
)c
on
substr(waq.QuarterCodeId,-4,3)=c.quartercode
order by 1,2,3")

#this fetch the data from database and display the results in RStudio.
dbFetch(question4)
```

Figure 19: Design for question 4.

In the figure, the increase and decrease in prices between two year is calculated and shown for a particular ward in a district.

6.2.4 Execution for question 4

Console | Terminal | Background Jobs

R 4.2.2 · D:/Master/Data Science/

	LocalAuthorityName	WardName	percentage
1	Cherwell	Adderbury, Bloxham and Bodicote	0%
2	Cherwell	Bicester West	-5%
3	Cherwell	Fringford and Heyfords	-7%
4	Cherwell	Kidlington West	-16%
5	Cherwell	Launton and Otmoor	-7%
6	Oxford	Blackbird Leys	0%
7	Oxford	Carfax	-15%
8	Oxford	Rose Hill and Iffley	-8%
9	Oxford	Summertown	6%
10	Oxford	Wolvercote	-17%
11	South Oxfordshire	Berinsfield	-21%
12	South Oxfordshire	Chalgrove	-29%
13	South Oxfordshire	Chinnor	-11%
14	South Oxfordshire	Forest Hill & Holton	-7%
15	South Oxfordshire	Woodcote & Rotherfield	-30%
16	Vale of White Horse	Abingdon Abbey Northcourt	4%
17	Vale of White Horse	Botley & Sunningwell	-10%
18	Vale of White Horse	Grove North	-5%
19	Vale of White Horse	Wantage & Grove Brook	-5%
20	Vale of White Horse	Watchfield & Shrivenham	-15%
21	West Oxfordshire	Alvescot and Filkins	29%
22	West Oxfordshire	Ascott and Shipton	7%
23	West Oxfordshire	Bampton and Clanfield	-6%
24	West Oxfordshire	Chipping Norton	-1%
25	West Oxfordshire	Witney North	14%

Figure 20: Execution of question 4.

6.2.5 For question 5

```
#this query returns data of a ward which has the highest houseprice in a particular
#(quarter of a) year in particular district.
question5 <- dbSendQuery(conn, "Select c.LocalAuthorityName,c.WardName,c.QuarterCodeId,
c.Year,c.Month,c.Price HighestHousePrice from
(select distinct la.LocalAuthorityName,wa.WardName,substr(waq.QuarterCodeId,-2,2) QuarterCodeId,
qb.Year,qb.Month,qb.Price,
dense_rank() over (PARTITION by la.LocalAuthorityName,substr(waq.QuarterCodeId,-2,2)
 order by price desc ) rn
from LocalAuthority la
inner JOIN ward wa
on la.LocalAuthorityCode =wa.LocalAuthorityCodeId
inner JOIN WardQuarter waq
on
wa.WardCode = waq.WardCodeId
inner join
quarter qb
on
substr(waq.QuarterCodeId,-4,4)=substr(qb.QuarterCode,-4,4)) c
where rn=1
order by 1,2,3")

#this fetch the data from database and display the results in RStudio.
dbFetch(question5)
```

Figure 21:Design for question 5.

In the figure, highest price in a particular quarter of a year is shown, considering all district.

6.2.6 Execution for question 5

Console | Terminal | Background Jobs

R 4.2.2 · D:/Master/Data Science/

	LocalAuthorityName	WardName	QuarterCodeId	Year	Month	Price
1	Cherwell	Launton and Otmoor	Q1	2021	Mar	461250
2	Cherwell	Launton and Otmoor	Q2	2021	Jun	475000
3	Cherwell	Launton and Otmoor	Q3	2021	Sep	488750
4	Cherwell	Launton and Otmoor	Q4	2021	Dec	500000
5	Oxford	Summertown	Q1	2020	Mar	790000
6	Oxford	Summertown	Q2	2020	Jun	785000
7	Oxford	Summertown	Q3	2021	Sep	840000
8	Oxford	Summertown	Q4	2021	Dec	795000
9	South Oxfordshire	Woodcote & Rotherfield	Q1	2021	Mar	762000
10	South Oxfordshire	Woodcote & Rotherfield	Q2	2021	Jun	778500
11	South Oxfordshire	Woodcote & Rotherfield	Q3	2021	Sep	810000
12	South Oxfordshire	Woodcote & Rotherfield	Q4	2021	Dec	757500
13	Vale of White Horse	Botley & Sunningwell	Q1	2021	Mar	410000
14	Vale of White Horse	Botley & Sunningwell	Q2	2021	Jun	420000
15	Vale of White Horse	Botley & Sunningwell	Q3	2021	Sep	436391
16	Vale of White Horse	Botley & Sunningwell	Q4	2021	Dec	441000
17	West Oxfordshire	Alvescot and Filkins	Q1	2020	Mar	700000
18	West Oxfordshire	Alvescot and Filkins	Q2	2020	Jun	570000
19	West Oxfordshire	Ascott and Shipton	Q3	2020	Sep	600000
20	West Oxfordshire	Ascott and Shipton	Q4	2020	Dec	510000

Figure 22:Execution of question 5.

6.2.7 For question 6

```
library(RSQLite)
setwd("D:/Master/Data Science")

conn <- dbConnect(SQLite(),"Coursework.db")

question6 <- dbSendQuery(conn, "Select c.ConstituencyName,wa.WardNames,b.AverageSpeed,b.SuperfastAvalability
from Constituency c
left join
WardBroadband w
on c.ConstituencyCode=w.ConstituencyCodeId
left join Wards wa
on w.WardCodeIds=wa.WardCodes
left join Broadbands b
on  wa.WardCodes=b.BroadbandCode
order by 1,2")

dbFetch(question6)
```

Figure 23:Design for question 6.

In the figure, average speed and superfast avalability is shown according to particular ward in a district.

6.2.8 Execution for question 6

Console | Terminal | Background Jobs

R 4.2.2 · D:/Master/Data Science/

	ConstituencyName	WardName	AverageSpeed	SuperfastAvalability
1	Oxford East	Barton	110.2	93.5%
2	Oxford East	Headington	94.2	90.7%
3	Oxford East	Marston	105.9	99.5%
4	Oxford East	Oxford Central	77.5	93.4%
5	Oxford East	Risinghurst & Sandhills	104	97.6%
6	Oxford West and Abingdon	Begbroke, Yarnton & Water Eaton	83.8	96.6%
7	Oxford West and Abingdon	Kidlington North	89.5	99.4%
8	Oxford West and Abingdon	Kidlington South	103.3	100.0%
9	Oxford West and Abingdon	Summertown	96	98.4%
10	Oxford West and Abingdon	Wolvercote & Cutteslowe	97.4	96.0%

Figure 24:Execution of question 6.

6.2.9 For question 7

```
question7 <- dbSendQuery(conn, "Select n.RegionName,c.ConstituencyName,wa.WardNames,b.AverageSpeed,b.SuperfastAvalability,
b.ReceivingUnder10Mbps,b.ReceivingOver30Mbps
from Constituency c, NationalRegion n
left join
WardBroadband w
on c.ConstituencyCode=w.ConstituencyCodeId
left join Wards wa
on w.WardCodeIds=wa.WardCodes
left join Broadbands b
on  wa.WardCodes=b.BroadbandCode
order by 1,2")

dbFetch(question7)
```

Figure 25: Design for question 7.

In the figure, the broadband speed information is displayed.

6.2.10 Execution for question 7

Console | Terminal | Background Jobs

R 4.2.2 · D:/Master/Data Science/

	RegionName	ConstituencyName	WardName	AverageSpeed	SuperfastAvalability	ReceivingUnder10Mbps	ReceivingOver30Mbps
1	South East	Oxford East	Marston	105.9	99.5%	6.8%	88.9%
2	South East	Oxford East	Barton	110.2	93.5%	3.1%	89.5%
3	South East	Oxford East	Headington	94.2	90.7%	3.0%	83.2%
4	South East	Oxford East	Risinghurst & Sandhills	104	97.6%	4.1%	86.5%
5	South East	Oxford East	Oxford Central	77.5	93.4%	5.7%	76.4%
6	South East	Oxford West and Abingdon	Kidlington North	89.5	99.4%	1.9%	83.6%
7	South East	Oxford West and Abingdon	Kidlington South	103.3	100.0%	2.0%	87.4%
8	South East	Oxford West and Abingdon	Begbroke, Yarnton & Water Eaton	83.8	96.6%	7.1%	83.5%
9	South East	Oxford West and Abingdon	Wolvercote & Cutteslowe	97.4	96.0%	5.4%	84.8%
10	South East	Oxford West and Abingdon	Summertown	96	98.4%	1.6%	85.3%

Figure 26:Execution for question 7.

6.2.11 For question 8

```
question8 <- dbSendQuery(conn, "Select aaa.AreaName, ag.AID, ag.BID, ag.CID, ag. AveragePrice from (
Select ap.AreaCodeId, AID, BID, BandCodeId CID, round((abc.Price+ap.Price)/2,2) AveragePrice from
(select a.AreaCodeId , a.BandCodeId AID , ab.BandCodeId BID, (a.Price+ AB.Price)/2 Price
from AreaBand a inner join AreaBand ab on a.AreaCodeId=ab.AreaCodeId
where a.BandCodeId<>ab.BandCodeId) ap inner join AreaBand abc on abc.AreaCodeId= ap.AreaCodeId
where abc.BandCodeId <> ap.AID and abc.BandCodeId<>ap.BID ) ag inner join Area aaa on ag.AreaCodeId = aaa.AreaCode
where ag.AID ='A' and ag.BID='B' and ag.CID='C'")

dbFetch(question8)
```

Figure 27: Design for question 8.

In the figure, same area, three band ‘A’, ‘B’ and ‘C’ average is calculated.

6.2.12 Execution for question 8

	AreaName	AID	BID	CID	AveragePrice
1	Adderbury	A	B	C	1670.93
2	Banbury	A	B	C	1740.77
3	Claydon	A	B	C	1704.54
4	Kidlington	A	B	C	1766.54
5	Wardington	A	B	C	1679.93
6	Alvescot	A	B	C	1673.74
7	Broadwell	A	B	C	1623.72
8	Chadlington	A	B	C	1648.58
9	Ducklington	A	B	C	1663.55
10	Stanton Harcourt	A	B	C	1677.08
11	Blackbird Leys	A	B	C	1791.94
12	Littlemore	A	B	C	1820.42
13	Old Marston	A	B	C	1815.09
14	Risinghurst and Sandhills	A	B	C	1806.76
15	Unparished areas	A	B	C	1790.10
16	Adwell	A	B	C	1634.58
17	Goring	A	B	C	1708.00
18	Ipsden	A	B	C	1677.41
19	Holton	A	B	C	1717.52
20	Wheatley	A	B	C	1687.78
21	Ardington	A	B	C	1679.81
22	Bourton	A	B	C	1663.40
23	Lyford	A	B	C	1642.99
24	Marcham	A	B	C	1704.84
25	Sunningwell	A	B	C	1687.42

Figure 28:Execution for question 8.

6.2.13 For question 9

```
question9 <- dbSendQuery(conn, "Select CounsilName, AreaCodeId, AreaName,BandCodeId, Price,
round((Lead(Price,-1) OVER
(Order by AreaCodeId) - Price),2) as Difference
From AreaBand B, Area A, Counsil C
where AreaCodeId in('A1','A2') and BandCodeId='A'
and B.AreaCodeId=A.AreaCode and C.CounsilCode =A.CounsilCodeId")

dbFetch(question9)
```

Figure 29: Design for question 9.

In figure, the difference in the same band ‘A’ for two area ‘Adderbury’ and ‘Banbury’ of same district is shown. There is 57.79 price difference in two bands.

6.2.14 Execution for question 9

```
                CounsilName AreaCodeId  AreaName BandCodeId   Price Difference
1 Cherwell District Council         A1 Adderbury          A 1382.84         NA
2 Cherwell District Council         A2   Banbury          A 1440.63     -57.79
```

Figure 30: Execution of question 9.

7 Testing of the system

To make system work properly testing should be done. The testing process of the system is carried out unit test. In unit test, adding data test, deleting data test and update data test is carried out.

7.1 Unit Testing

7.1.1 Adding data and displaying

Test number	1
Action	Values are inserted in the database using insert query.
Expected Result	Values should be added in the respective table.
Actual Result	Data added to database in the respective table.
Test Result	Successful

Table 1: Test for adding data.

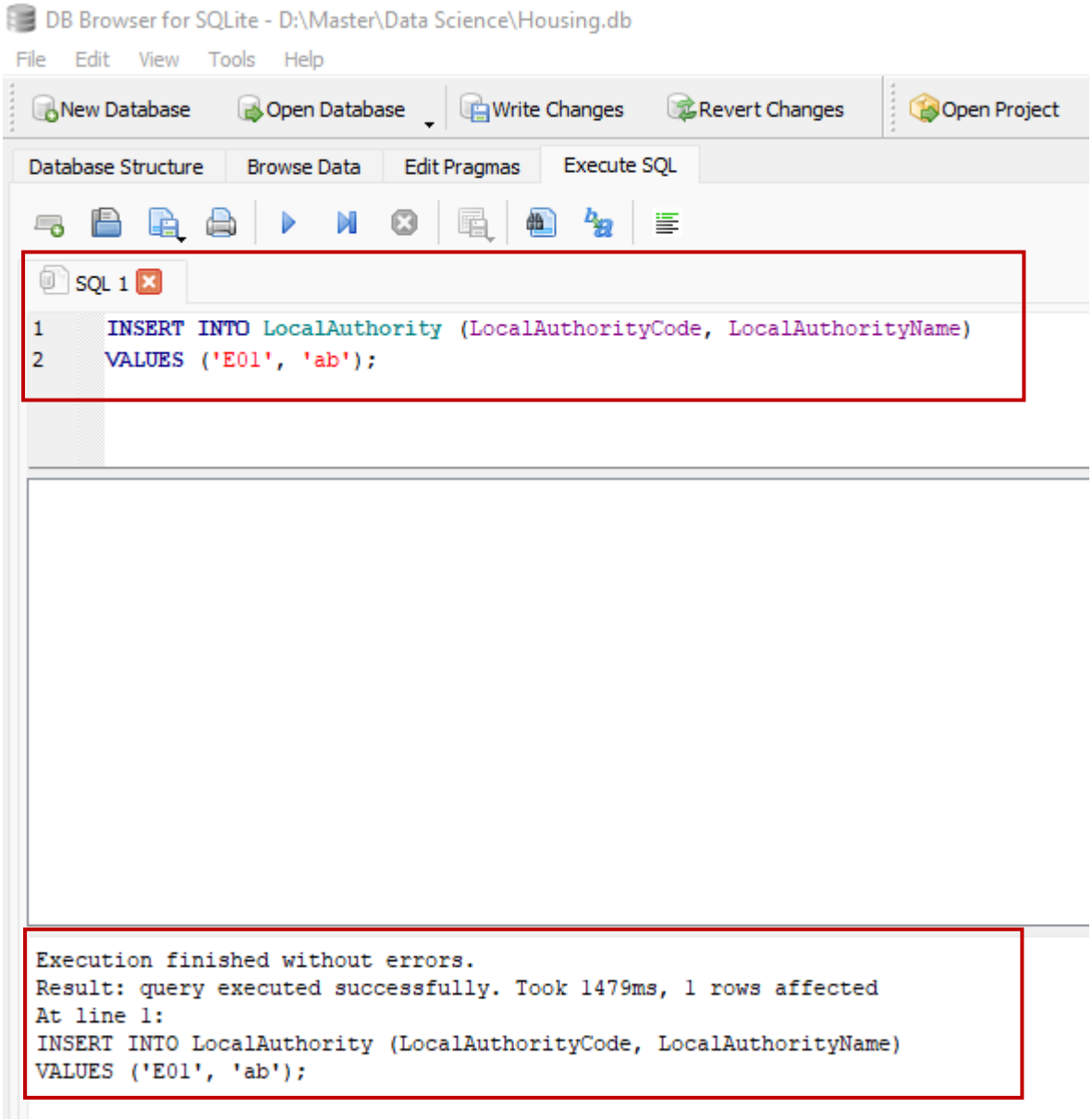


Figure 31 Executing insert query.

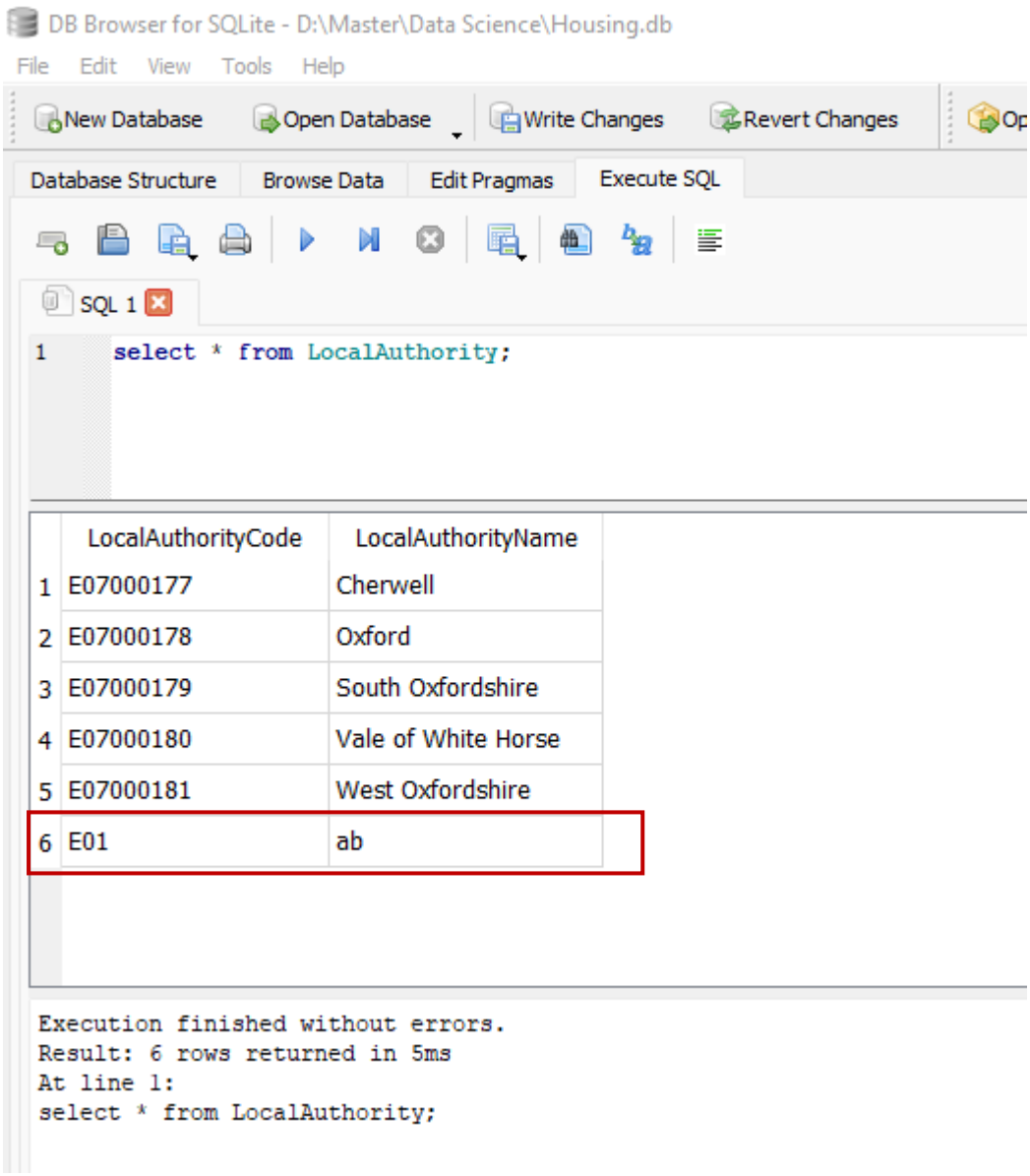


Figure 32: Data inserted in a table.

7.1.2 Deleting data and displaying

Test number	2
Action	Perform delete action to delete irrelevant data.
Expected Result	Data should be removed from the table.
Actual Result	Data is removed from the table.
Test Result	Successful

Table 2: Test for deleting data.

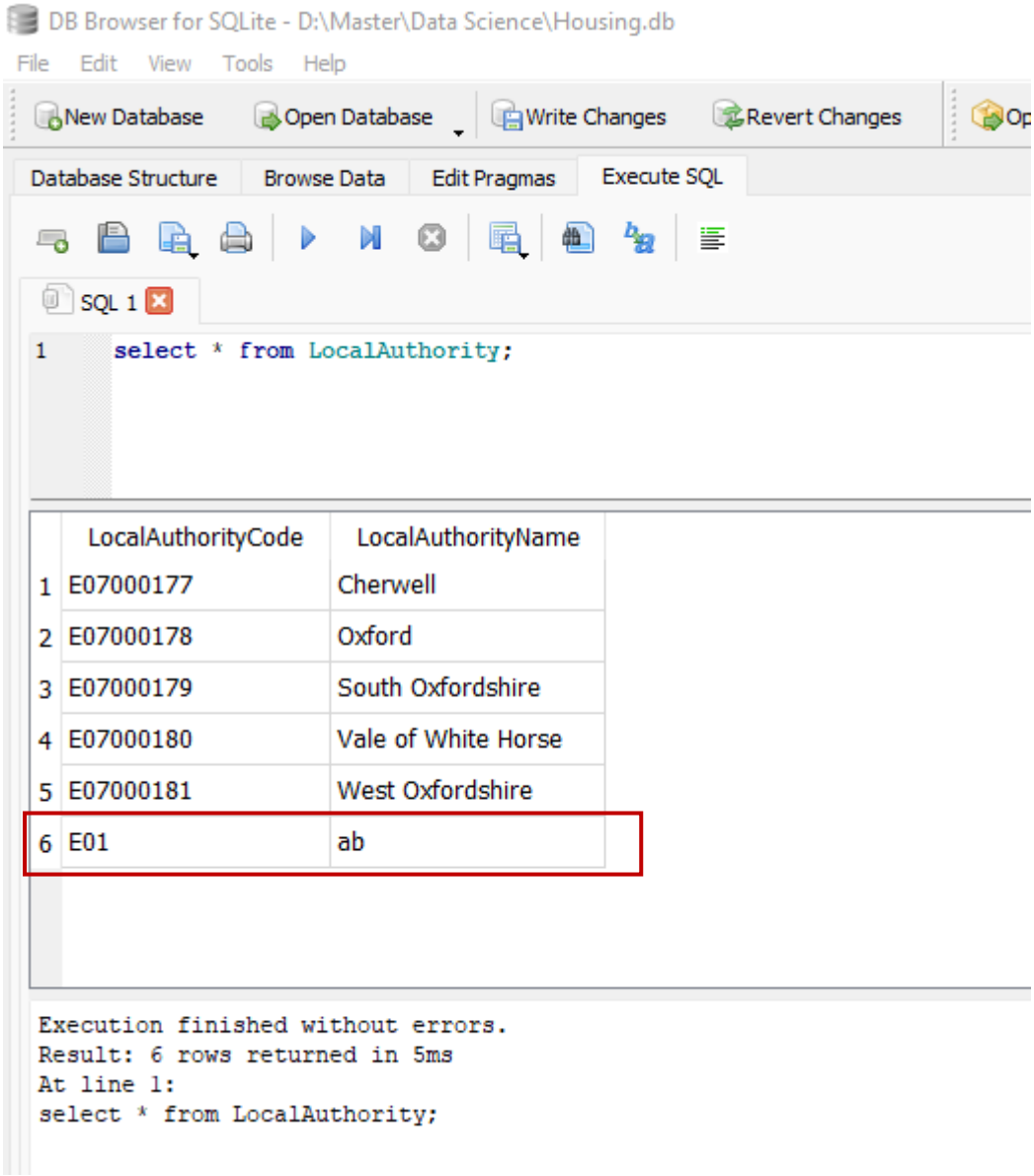


Figure 33: Displaying data from the table.

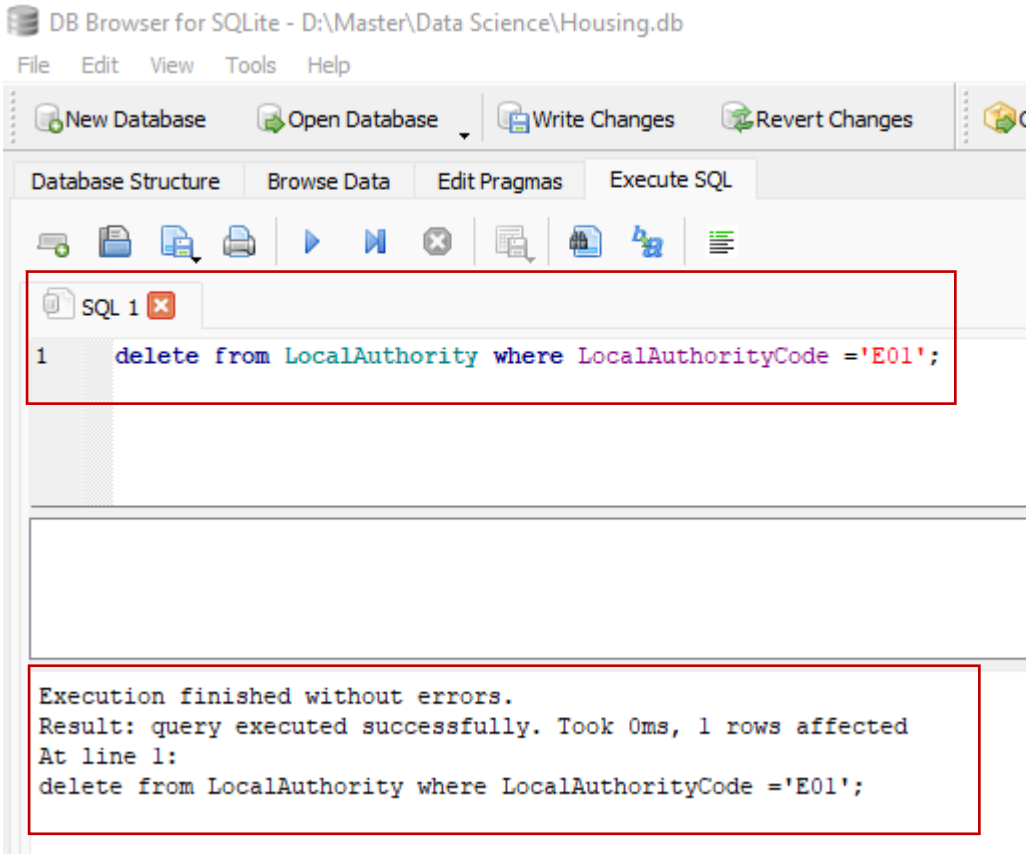


Figure 34: Deleted the underline data.

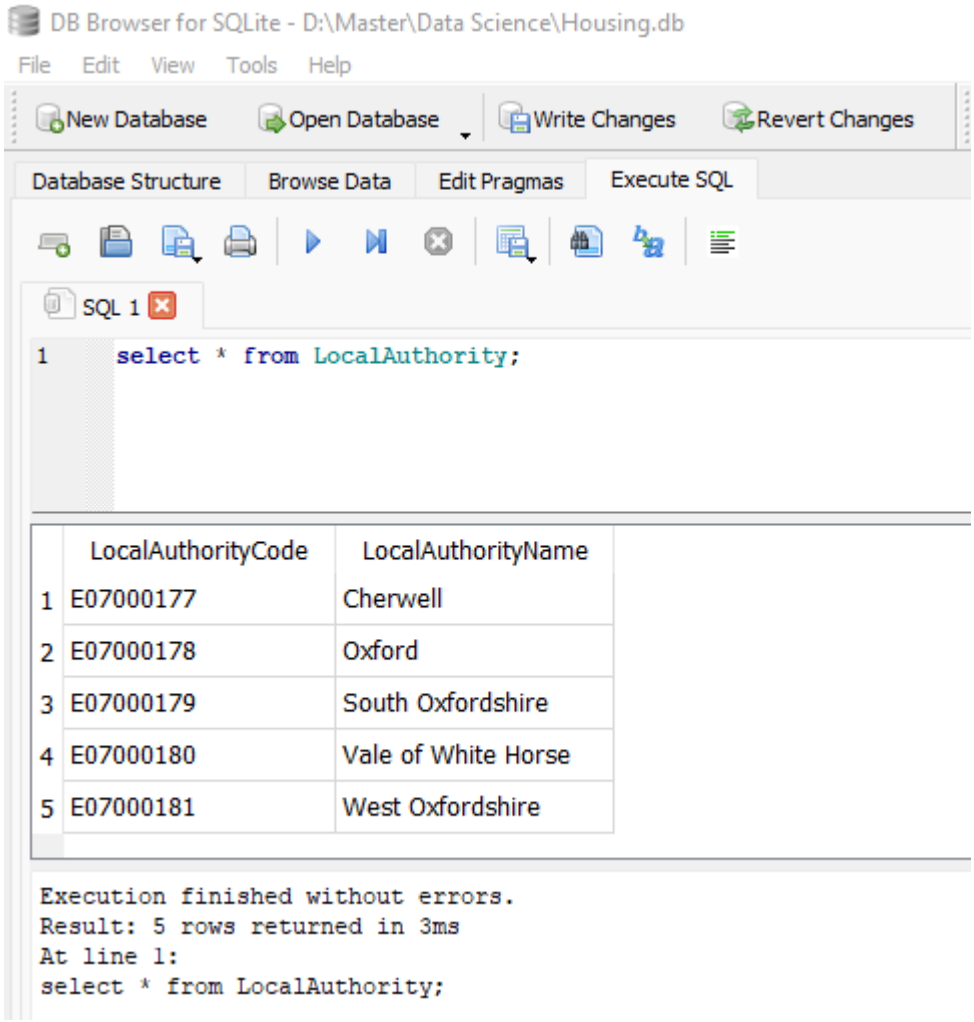
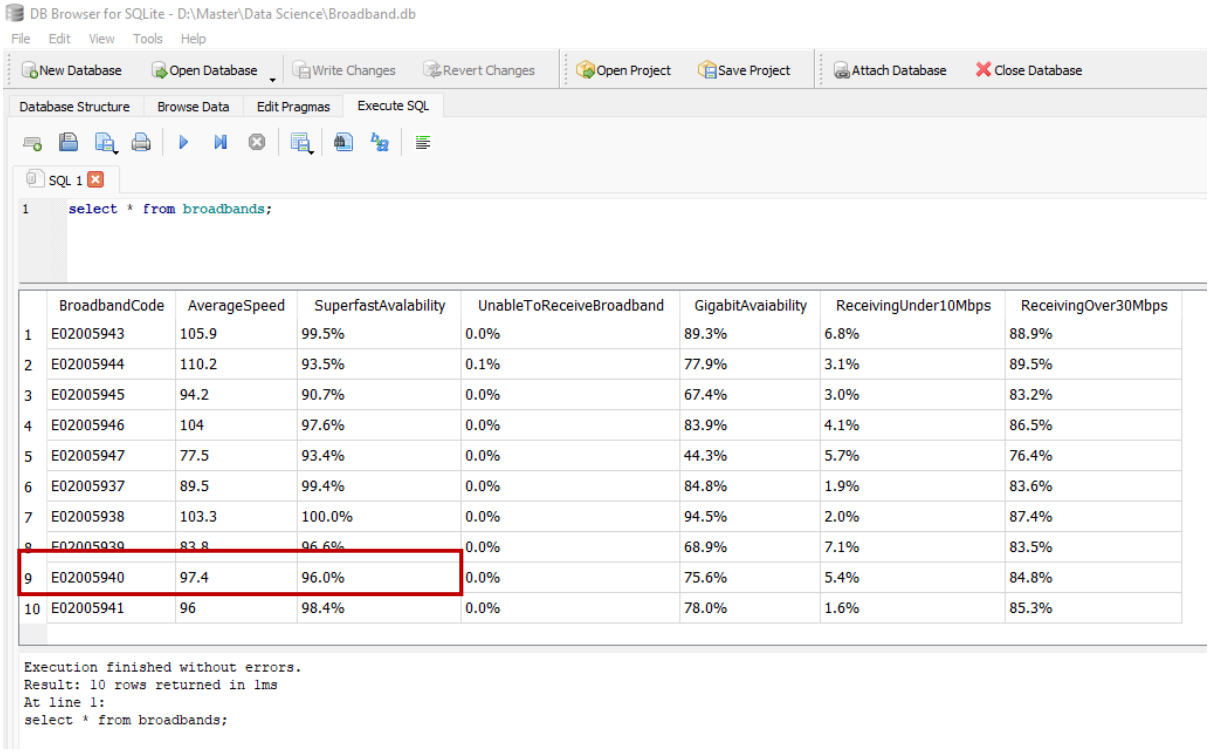


Figure 35: Displaying the data from the table after deletion of data.

7.1.3 Updating data and displaying

Test number	3
Action	Perform update action to update data.
Expected Result	Data should be updated from the table.
Actual Result	Data is updated in the table.
Test Result	Successful

Table 3: Test for updating data.



	BroadbandCode	AverageSpeed	SuperfastAvalability	UnableToReceiveBroadband	GigabitAvaiability	ReceivingUnder10Mbps	ReceivingOver30Mbps
1	E02005943	105.9	99.5%	0.0%	89.3%	6.8%	88.9%
2	E02005944	110.2	93.5%	0.1%	77.9%	3.1%	89.5%
3	E02005945	94.2	90.7%	0.0%	67.4%	3.0%	83.2%
4	E02005946	104	97.6%	0.0%	83.9%	4.1%	86.5%
5	E02005947	77.5	93.4%	0.0%	44.3%	5.7%	76.4%
6	E02005937	89.5	99.4%	0.0%	84.8%	1.9%	83.6%
7	E02005938	103.3	100.0%	0.0%	94.5%	2.0%	87.4%
8	E02005939	83.8	96.6%	0.0%	68.9%	7.1%	83.5%
9	E02005940	97.4	96.0%	0.0%	75.6%	5.4%	84.8%
10	E02005941	96	98.4%	0.0%	78.0%	1.6%	85.3%

Figure 36: Updating the underlined data.

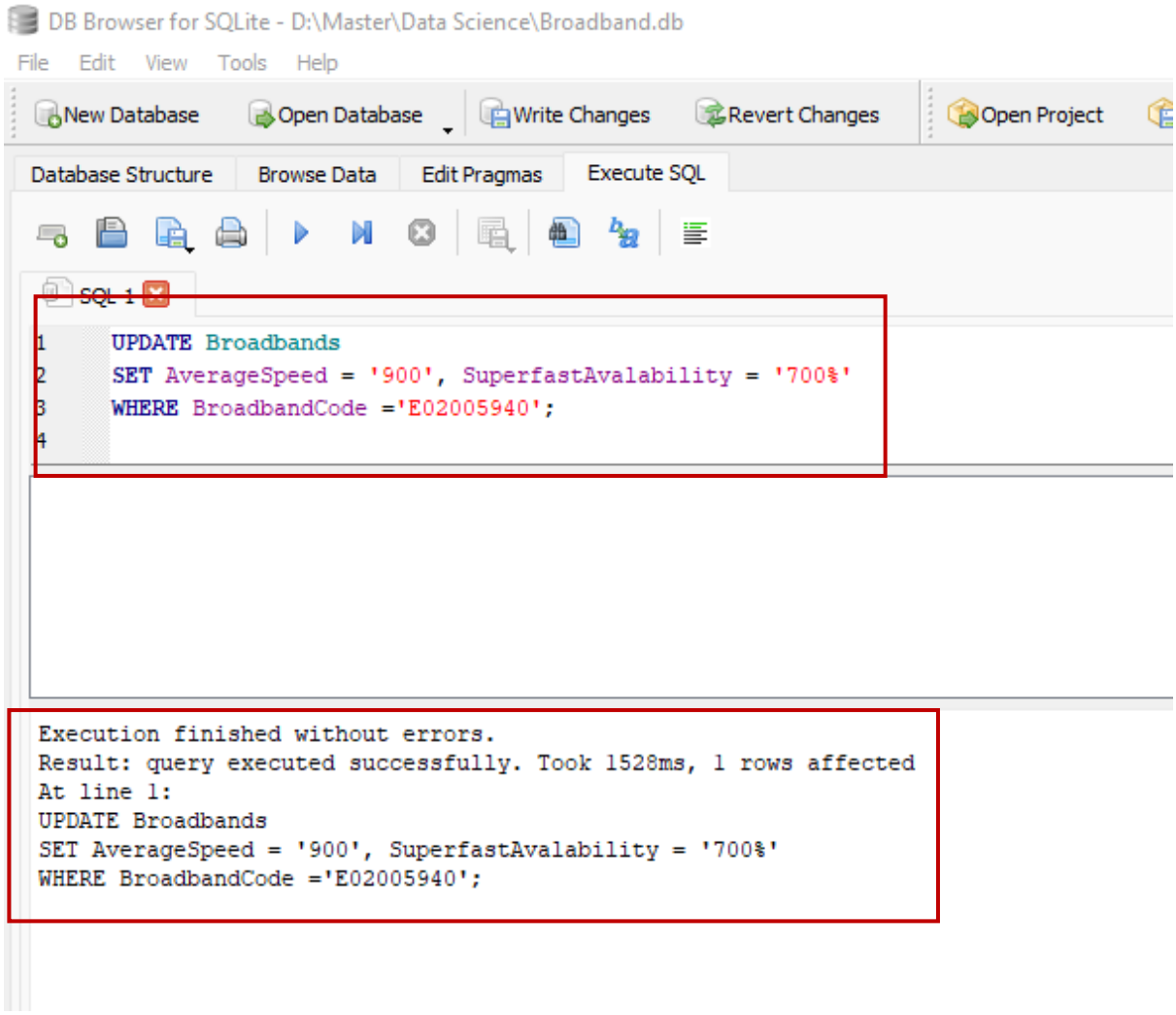
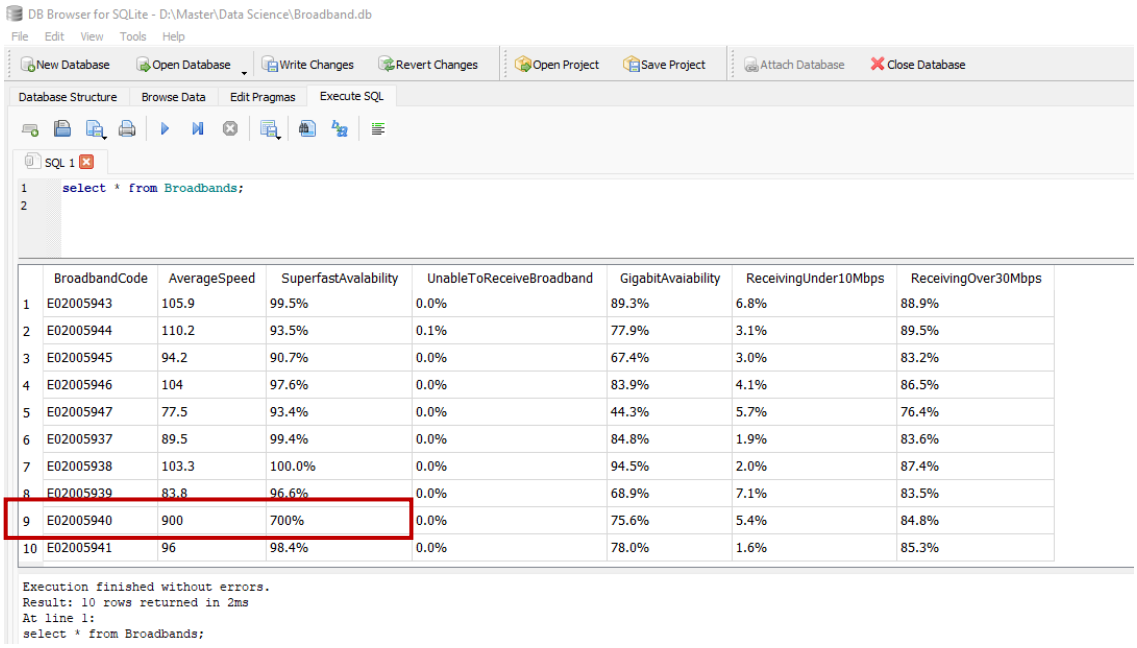


Figure 37:Executing update query.



	BroadbandCode	AverageSpeed	SuperfastAvalability	UnableToReceiveBroadband	GigabitAvaiability	ReceivingUnder10Mbps	ReceivingOver30Mbps
1	E02005943	105.9	99.5%	0.0%	89.3%	6.8%	88.9%
2	E02005944	110.2	93.5%	0.1%	77.9%	3.1%	89.5%
3	E02005945	94.2	90.7%	0.0%	67.4%	3.0%	83.2%
4	E02005946	104	97.6%	0.0%	83.9%	4.1%	86.5%
5	E02005947	77.5	93.4%	0.0%	44.3%	5.7%	76.4%
6	E02005937	89.5	99.4%	0.0%	84.8%	1.9%	83.6%
7	E02005938	103.3	100.0%	0.0%	94.5%	2.0%	87.4%
8	E02005939	83.8	96.6%	0.0%	68.9%	7.1%	83.5%
9	E02005940	900	700%	0.0%	75.6%	5.4%	84.8%
10	E02005941	96	98.4%	0.0%	78.0%	1.6%	85.3%

Figure 38: Data updated in the table.

8 References

Devin, P. (November 16, 2018) *Structured vs Unstructured Data – What's the Difference?* Available at: https://www.g2.com/articles/structured-vs-unstructured-data.

Endenicher *Structured Data.* Available at: https://www.bigdataframework.org/data-types-structured-vs-unstructured-data/#:~:text=Structured%20Data,Excel%20files%20or%20SQL%20databases.

Snowflake (2022) *STRUCTURED DATA VERSUS SEMI-STRUCTURED DATA.* Available at: https://www.snowflake.com/guides/structured-data-versus-semi-structured-data.

Teradata 2022. What is Semi-Structured Data?

GOV.UK *Data Ethics Framework.* Available at: https://www.gov.uk/government/publications/data-ethics-framework/data-ethics-framework-legislation-and-codes-of-practice-for-use-of-data.

Council, O. C. (31 December 2022) *Council Tax Charges 2022-23.* Available at: https://www.oxford.gov.uk/info/20152/council_tax_bands_and_charges/120/council_tax_charges.

Council, S. O. D. *Council Tax Calculator.* Available at: https://data.southoxon.gov.uk/ccm/support/Main.jsp?MODULE=Calculator.

Council, V. o. W. H. D. (2022) *Council Tax Calculator.* Available at: https://data.whitehorsedc.gov.uk/java/support/Main.jsp?MODULE=Calculator.

Council, W. O. D. *Council Tax charges.* Available at: https://www.westoxon.gov.uk/council-tax-and-benefits/council-tax-bands-charges-and-appeals/.

Counsil, C. D. (2022) *Cherwell Council Tax charges 2022/23.* Available at: https://www.cherwell.gov.uk/directory/22/council-tax-charges-2021-2022.

Baker, C. (21 October, 2022) *Constituency data : broadband coverage and speeds.* Available at: https://commonslibrary.parliament.uk/constituency-data-broadband-coverage-and-speeds/.

North, A. (16 December 2022) *Median house prices by ward: HPSSA dataset 37.* Available at: https://www.ons.gov.uk/peoplepopulationandcommunity/housing/datasets/medianpricepaidbywardhpssadataset37.

9 Appendix

```
library(RSQLite)

setwd("D:/Master/Data Science")


conn <- dbConnect(SQLite(),"Coursework.db")


question3 <- dbSendQuery(conn, "Select DISTINCT LocalAuthorityName,WardName,
substr(qa.QuarterCode, -2,2) Quarter,

AveragePrice from Ward w

inner join LocalAuthority la on la.LocalAuthorityCode=w.LocalAuthorityCodeId

inner join WardQuarter wq on wq.WardCodeId = w. WardCode

inner join ( select q.QuarterCode , (q.Price + qq.Price)/2 AveragePrice from Quarter qq

inner join Quarter q on substr(qq.QuarterCode, -4,4)= substr(q.QuarterCode, -4,4)where q.Year <>
qq.Year) qa on

wq.QuarterCodeId = qa. QuarterCode")

dbFetch(question3)


question4 <- dbSendQuery(conn, "Select distinct
la.LocalAuthorityName,wa.WardName,c.percentage

from LocalAuthority la

inner JOIN ward wa

on la.LocalAuthorityCode =wa.LocalAuthorityCodeId

inner JOIN WardQuarter waq

on

wa.WardCode = waq.WardCodeId

inner join

(

select substr(qa.quartercode,-4,3) as quartercode,qa.year,qb.year,sum(qa.price),sum(qb.price),
(qa.price-qb.price),

(((qa.price-qb.price)*100)/(qa.price))||'%'  as percentage

 from quarter qa inner join

 quarter qb on substr(qa.quartercode, -4,3) = substr(qb.quartercode,-4,3)
```

```
where qa.year = '2020' and qb.year = '2021'
 group by substr(qa.quartercode,-4,3), qa.year,qb.year
 )c on substr(waq.QuarterCodeId,-4,3)=c.quartercode
order by 1,2,3")
dbFetch(question4)

question5 <- dbSendQuery(conn, "Select
c.LocalAuthorityName,c.WardName,c.QuarterCodeId,c.year,c.month,c.price from (
select distinct la.LocalAuthorityName,wa.WardName,substr(waq.QuarterCodeId,-2,2)
QuarterCodeId,qb.year,qb.month,qb.price,
dense_rank() over (PARTITION by la.LocalAuthorityName,substr(waq.QuarterCodeId,-2,2)
 order by price desc ) rn
from LocalAuthority la
inner JOIN ward wa
on la.LocalAuthorityCode =wa.LocalAuthorityCodeId
inner JOIN WardQuarter waq
on wa.WardCode = waq.WardCodeId
inner join
quarter qb
on substr(waq.QuarterCodeId,-4,4)=substr(qb.QuarterCode,-4,4)) c
where rn=1
order by 1,2,3")
dbFetch(question5)

question6 <- dbSendQuery(conn, "Select
c.ConstituencyName,wa.WardNames,b.AverageSpeed,b.SuperfastAvalability
from Constituency c
left join
WardBroadband w
on c.ConstituencyCode=w.ConstituencyCodeId
left join Wards wa
on w.WardCodeIds=wa.WardCodes
```

```
left join Broadbands b

on  wa.WardCodes=b.BroadbandCode

order by 1,2")

dbFetch(question6)


question7 <- dbSendQuery(conn, "Select
n.RegionName,c.ConstituencyName,wa.WardNames,b.AverageSpeed,b.SuperfastAvalability,

b.ReceivingUnder10Mbps,b.ReceivingOver30Mbps

from Constituency c, NationalRegion n

left join

WardBroadband w

on c.ConstituencyCode=w.ConstituencyCodeId

left join Wards wa

on w.WardCodeIds=wa.WardCodes

left join Broadbands b

on  wa.WardCodes=b.BroadbandCode

order by 1,2")

dbFetch(question7)


question8 <- dbSendQuery(conn, "Select aaa.AreaName, ag.AID, ag.BID, ag.CID, ag. AveragePrice
from (

Select ap.AreaCodeId, AID, BID, BandCodeId CID, round((abc.Price+ap.Price)/2,2) AveragePrice from

(select a.AreaCodeId , a.BandCodeId AID , ab.BandCodeId BID, (a.Price+ AB.Price)/2 Price

from AreaBand a inner join AreaBand ab on a.AreaCodeId=ab.AreaCodeId

where a.BandCodeId<>ab.BandCodeId) ap inner join AreaBand abc on abc.AreaCodeId=
ap.AreaCodeId

where abc.BandCodeId <> ap.AID and abc.BandCodeId<>ap.BID ) ag inner join Area aaa on
ag.AreaCodeId = aaa.AreaCode

where ag.AID ='A' and ag.BID='B' and ag.CID='C'")

dbFetch(question8)
```

```
question9 <- dbSendQuery(conn, "Select CounsilName, AreaCodeId, AreaName,BandCodeId, Price,
round((Lead(Price,-1) OVER
(Order by AreaCodeId) - Price),2) as Difference
From AreaBand B, Area A, Counsil C
where AreaCodeId in('A1','A2') and BandCodeId='A'
and B.AreaCodeId=A.AreaCode and C.CounsilCode =A.CounsilCodeId")
dbFetch(question9)
```