|  |  |  |
| --- | --- | --- |
| Version | Reason | Author |
| 0.1 | Initial draft | MR |
| 0.2 | Review, Proof read and formatting | GL |
| 0.3 | Add additional content respond to points raised | MR |
| 0.4 | Visio diagram added | mr |

UPRN Seeding Workflow

# Introduction

The objective of this process is to get UPRN or Parent UPRN based on accurate and comparable variables where possible from each address string equivalent to those imputed from the Address base file. The data then goes to the oracle table for actual seeding linkage stage.

A user has some text strings which are addresses with or without postcodes. There is an assumption here that the given postcodes do at least vaguely relate to the address string if present. The user wishes to assign UPRN (Unique property Reference number) to gain benefit of precise point level location and consistent addressing to group person level records into households.

# Data Challenges

Incoming addressing data will come to us in a variety of formats, sometimes with business names or partial addressing, address ranges, and buildings that no longer exist or have been altered into new sub-units. Consistency within address strings can be variable too so street number, name and flat position ordering within the string can be unexpected.

|  |  |  |  |
| --- | --- | --- | --- |
| Inclusion logic | Address String Contents | Comments / example | Utility |
| Exclude | Nonsense / unhelpful or misleading text | c/o followed by name or other non-address text. | Negative to matching |
| Best course is do 2 runs with / without and aggregate | Company/Institution name | Alan Bros Ltd | Generally more difficult to work with as these are often inconsistently written both in address base and incoming data. Often better to exclude this and focus on other aspects of address. |
| Initial match with sub unit then to parent | Flat / Or Unit Or Room | 2/1 or 2-1 or unit b etc | Written very inconsistently. Subtle standardisation helps if renumbering has happened or it is very inconsistent. Assignment of Parent UPRN is the goal to get building level match |
| Generally key metric few exceptions where building name means people don’t write street number | street number or number range | Any number or number letter combination 12, 12c etc | Assumption is this is included if relevant. Massively important to get this element correct |
| If exists key, but significant variations | Building Name | The Morrison Building or South Lodge | Usually includes massive effect on scoring. Can be excluded if an address also has a number |
| Possible not improvement is to match ambiguous addressing to USRN centres | Street Name | Camphill Avenue | Almost always there and very useful to find an address! Often abbreviated |
| Used for non prostcoded addresses to assign start of search | Town or locality | Newton Mearns, Stewarton, shawlands | Can be included or missed depending on habit |
| Not particularly useful | City | Glasgow, Dundee etc | Normally included |
| Not particularly useful | County | North Lanarkshire | Very optional but can affect similarity scoring a lot as they can contain long names |
| Can be misleading, prone to typos and problems with validation used only as a fuzzy geo search start point | postcode | G41 3AU | Can be unreliable, old or moved over time |

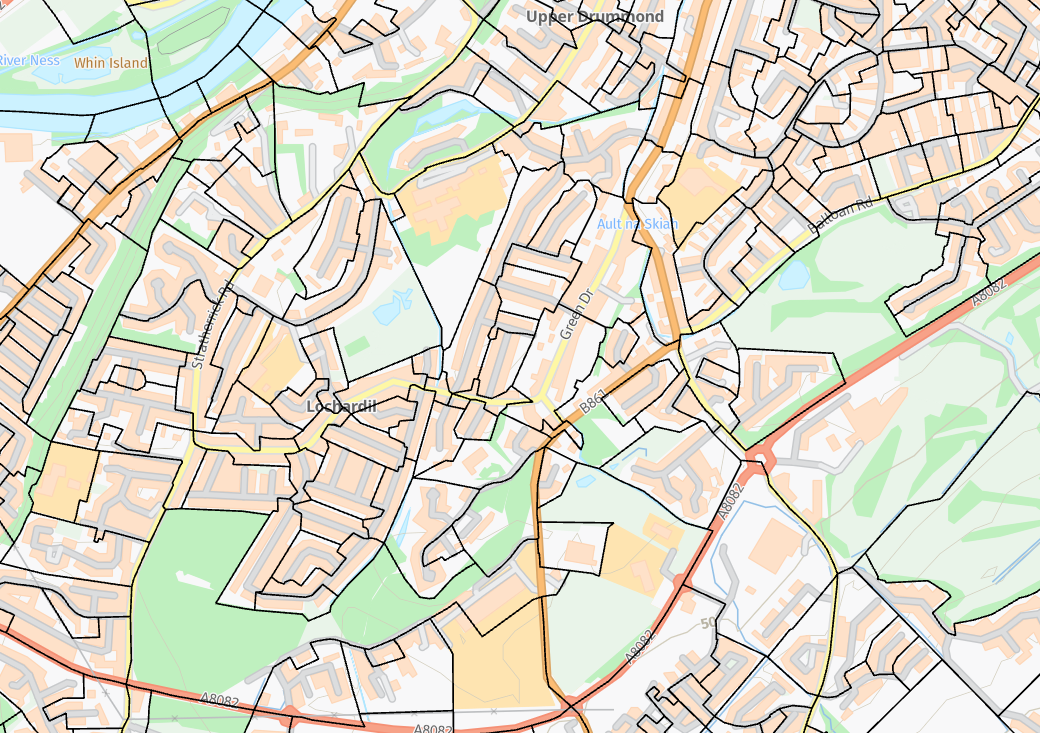
# Step 1 - Cleaning Incoming data to be seeded

To get incoming addressing into a similar state or assign equivalent features as has been done on the addressbase file, a data cleaning script is used. It needs adjustments depending on how addresses are supplied. H3 tiles are defined for increasing search from a given centroid.

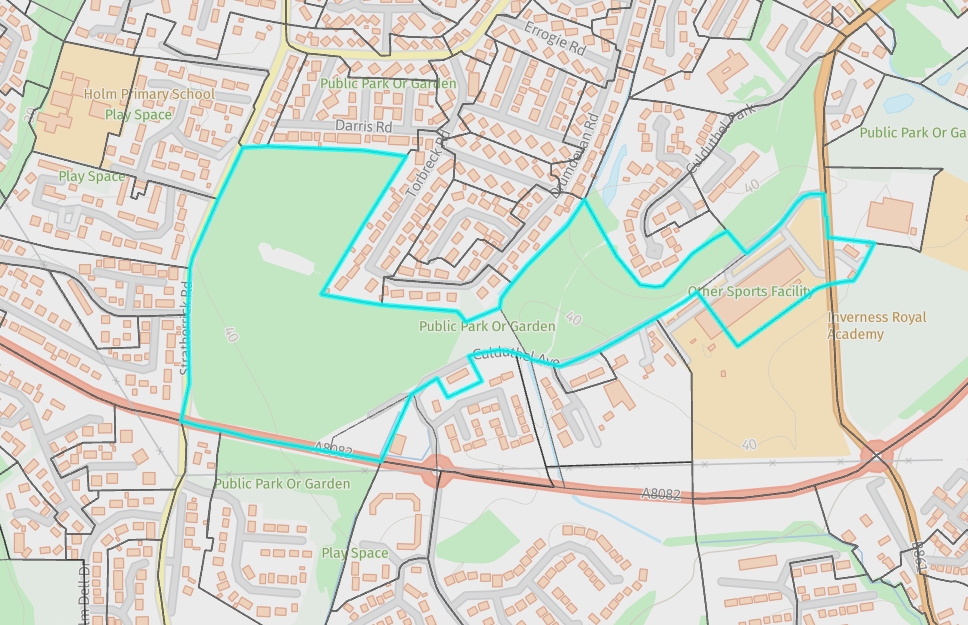
[\\cl-out\Martin\_R\UPRN Seeding\PRE\_PROCESS\_CHI\_ADDRESSING\_CREATE\_SQL\_TABLE\_FOR SEEDING\_65\_2.R](file:///\\cl-out\Martin_R\UPRN%20Seeding\PRE_PROCESS_CHI_ADDRESSING_CREATE_SQL_TABLE_FOR%20SEEDING_65_2.R) is an example cleaning script for CHI addressing as incoming data. As mentioned above, the quality and formats of incoming addresses usually will vary with different datasets. Over coming projects we will amass a set of these cleaning scripts for incoming address data to deal with and adapt for the slight variations between the incoming datasets.

Therefore, the cleaned results are postcode centroids derived from supplied postcodes, and used to assign H3 tiles to the incoming records or a town trimmed from the end of the address string if no postcode is supplied.

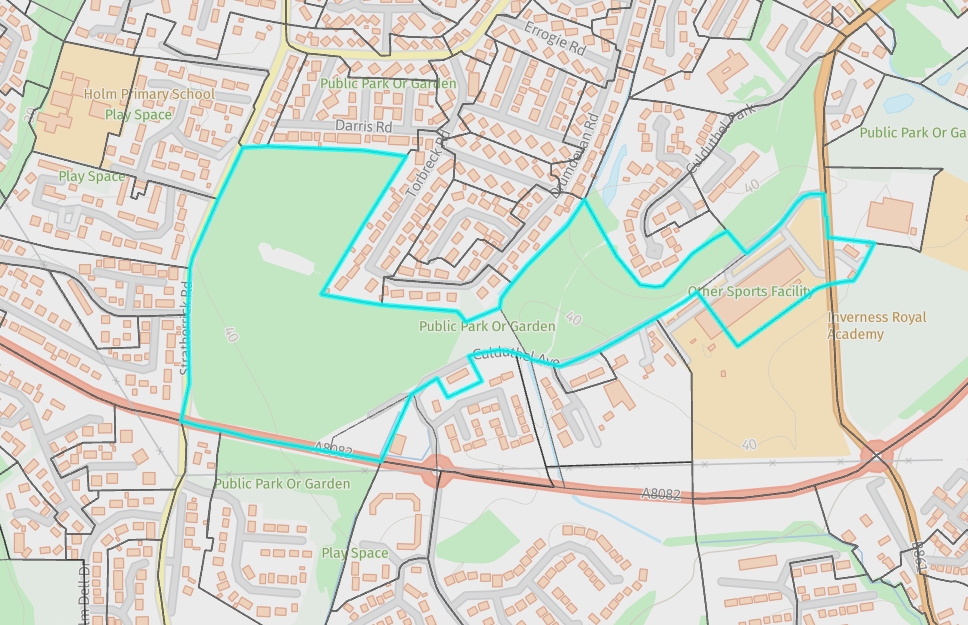
# Uber’s H3 tiles



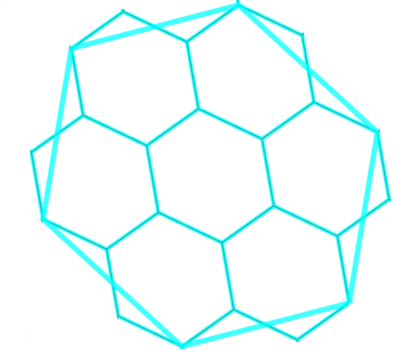
In the Scotland we are all familiar with Postcode areas they are ubiquitous, what we might not appreciate is the flux they are in, how irregular they are and prone to error at input. The map above is intended to show the diversity of area even ins a predominantly urban zone. Institutions can get their own postcodes as large users, “normal” residential addressing can on average have 15 other addresses within it but there is considerable variation in this. Over time postcodes change to facilitate efficient mail delivery, merging, altering size, position, being deleted and new areas introduced.



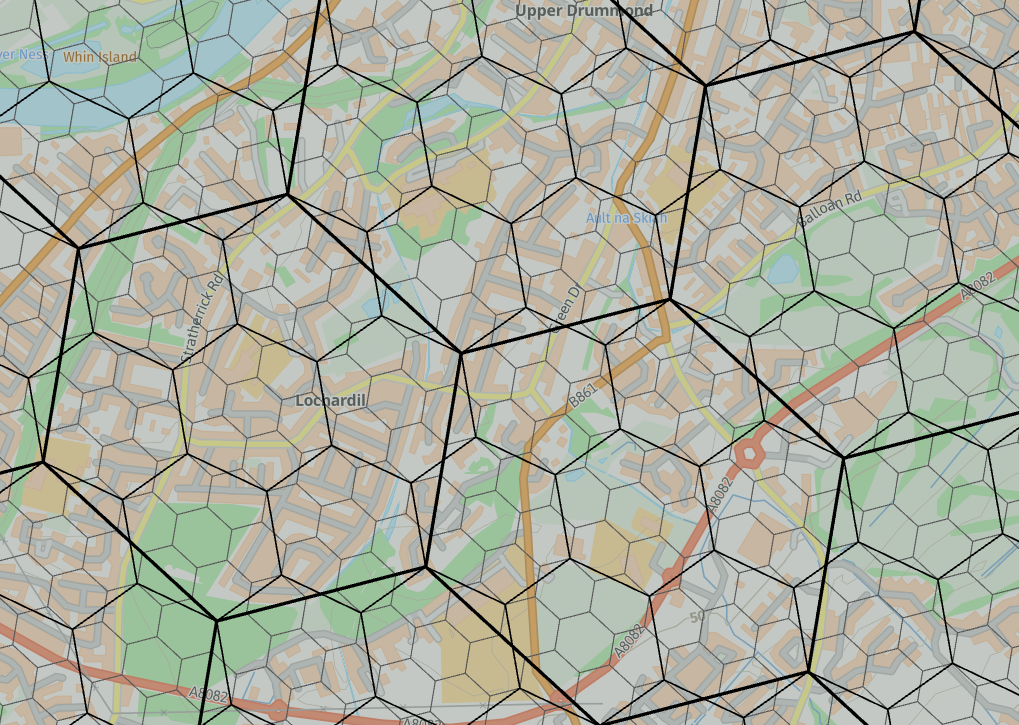
Select a postcode at random, and overlaying 2 iterations of the postcode shape file to shows how the highlighted blue area is now split by 4 new postcode areas. Furthermore in such an area the centroid for the sports facility might move from :



Our health records are recorded over a long time typically a home address is entered a point of GP registration with whatever the understanding of the address and its postcode was at that time. This is a frozen recording it doesn’t auto update if the home address is caught in the administrative changes described above. Further more if a postcode is mistyped by even 1 character is can mislocate based associate centroid coordinates. Postcodes that are similar are not grouped together. Enter Ubers H3 grid:



The above example diagram shows how H3 nest , there are 0-15 size tile in the hierarchy. Tiles sized 10 to 4 are used in the uprn seeding process to give optimal search areas for different areas.

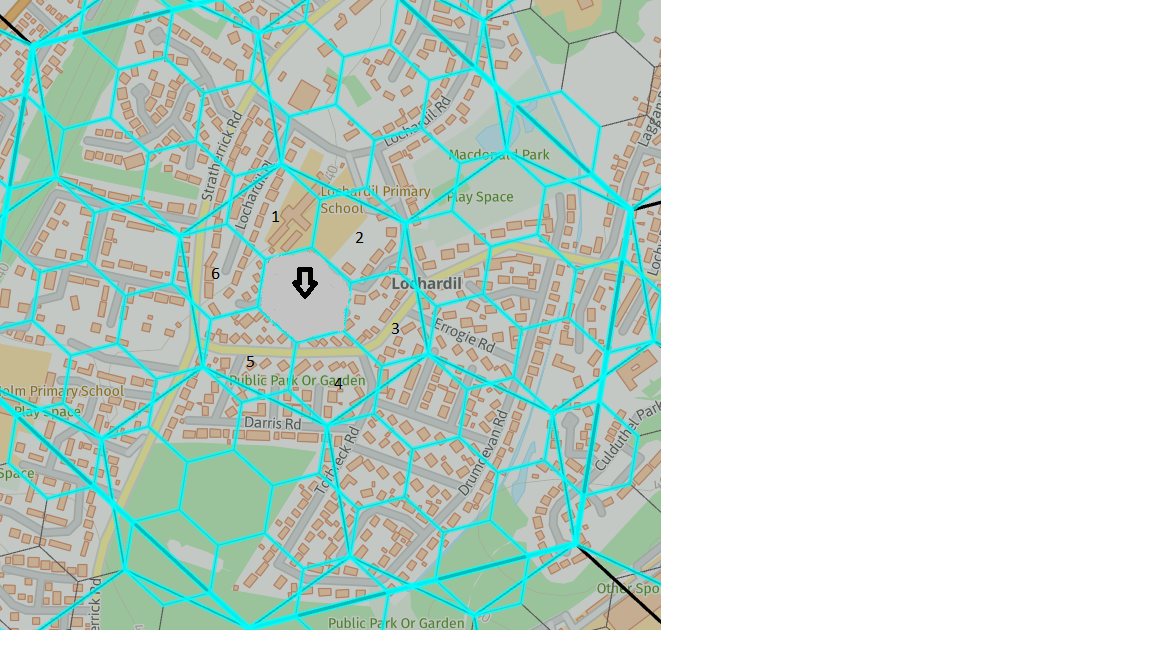


The H3 indexing system places a hexagonal tile, close in its index to those entities it is in proximity to on the ground. The above map shows H3 tiles 8-10 nesting against a real world map. Neighbouring tiles can be computed easily and will be used to increase search areas systematically until a suitable match is found.



Critically H3 tiles are computed using an algorithm which means they are immutable. By using an approximate starting location the area of area for an address is controlled until a suitability strict match can be accepted. Matches are accepted based proximity to a search start point and similarity in terms of address text features. Also taken into account are the number of feasible match candidates (see below for more on matches).

# Search discussion.



The above map shows a *potential* target tile where a postcode centroid is located and it is assumed the proximity of the actual location in the first instance is in that tile unless no suitable match is found. If no match is found a second query to expand the search area to include the area defined by tiles labelled 1-6, followed by a larger ring of outer neighbouring tiles. In a rural area a larger tile size is used later in the staged seeding process.

The reference data also geo binned in the same manner as the incoming data we need to UPRN seed. The benefit of applying this both sides of the linkage equation not only handles the search area but also allows the size of comparisons to be controlled and therefore avoids comparing addresses in areas which are not relevant to the search.

# UPRN Seeding

## What is it?

This step is a record linkage exercise where incoming address strings are compared for similar characteristics against a cleasned version of Addressbase and “possible matches” are generated in a staging table. Addressbase is cleansed too as it has many entities in health we aren’t interested in and only add noise and size to the data. In short, and surprisingly, things do not match due to human involvement, context of data entry, and change in the physical environment overtime. The Process to generate possible matches then accepting matches is written using ORACLE. This was chosen mainly for scalability allowing relatively large files to be seeded.

## Generation of Possible Matches

From the “Possible matches” “accepted matches” are generated based on meeting certain geographical proximity textual similarity thresholds and ranking of near rivals.

Once all "possible matches” are generated at each stage, they need to be evaluated to determine the matches to be marked as accepted matches for each stage. At each stage, once the best or accepted matches are agreed or identified, the initial “possible matches” table is deleted to save space. Then to save processing and space, the agreed matches are removed from the remaining pool of addresses to be seeded. The way we handle evaluation of possible matches into agreed matches is varied, but generally what is done is firstly a grouping of the possible matches by unique incoming ID. Then, if there are multiple records and the scores are over a threshold, the top score is selected (the score is a similarity metric based on all or certain features of the address strings). If there is only one record and the score is over a threshold, this is accepted.

## Carrying out seeding

1. Pre process the most recent Addressbase file
2. Cleanse and format incoming addresses file
3. Run seeding
4. Evaluate where matches are made and run sampling script
5. Review an agreed sample size and calculate estimated error rate
6. Return agreed data / map to customer.
7. Reset Oracle environment

The method attempts to go from strictest matches to less similar matches in hierarchy in such a way to avoid accepting false positives in great numbers. These iterations of many match keys (criteria in the where statement and scoring logic) attempt to account for typical variations in incoming data versus reference Addressbase.

## Methodology

Stages are ordered to run from most strict to less strict, typically each stage using a combination of string searching and string distance metrics combined with spatial proximity using the H3 tile hierarchy. Key word searching is also used.Finding spatial proximity becomes easy by finding addresses in the same h3 tile or neighbouring tiles.

Address strings are tokenised into their constituent words these words have the double metaphone calculated, so fuzzy comparisons can be made. The initial key words can be searched in the reference addresses tokened words to find addresses in an area which share common initial words for further comparison this is fruitful where addresses don’t contain numbers.

String searching refers to looking for a sub set of an incoming string in the reference data or vice versa. Eg Unit 2, gyle Shopping centre, nowhere street, Edinburgh might have the text following unit 2 searched in the reference data using an instr argument. Or the 2 from unit 2 could be using to find all addresses with a sub unit of 2 in the address in that area.

String distance metrics compare two strings and return a score to indicate similarity:

Jaro winkler: [Jaro–Winkler distance - Wikipedia](https://en.wikipedia.org/wiki/Jaro%E2%80%93Winkler_distance#:~:text=The%20Jaro%E2%80%93Winkler%20distance%20uses%20a%20prefix%20scale%20which,match%20and%200%20means%20there%20is%20no%20similarity.)

Levenstein edit distance : [Levenshtein distance - Wikipedia](https://en.wikipedia.org/wiki/Levenshtein_distance)

A set of accepted matches are accumulated in a series of tables which are assigned a stage key for future traceability and sampling for errors. Possible matches are grouped for example in an analytic query by incoming ID, score and count of UPRN the resultant best match is accepted. Because most similar and strictest matches are made first different combinations of less strict grouping on the tail can be used to find matches where maybe only one parent UPRN is possible making a building level match but not sub unit in the case of flats.

These tables then are bought together by a UNION query and this holds the result.

Unmatched records remain in the Second\_a table in the users oracle database. The original load table remains to show the total records for the seeding job in Add\_in table.

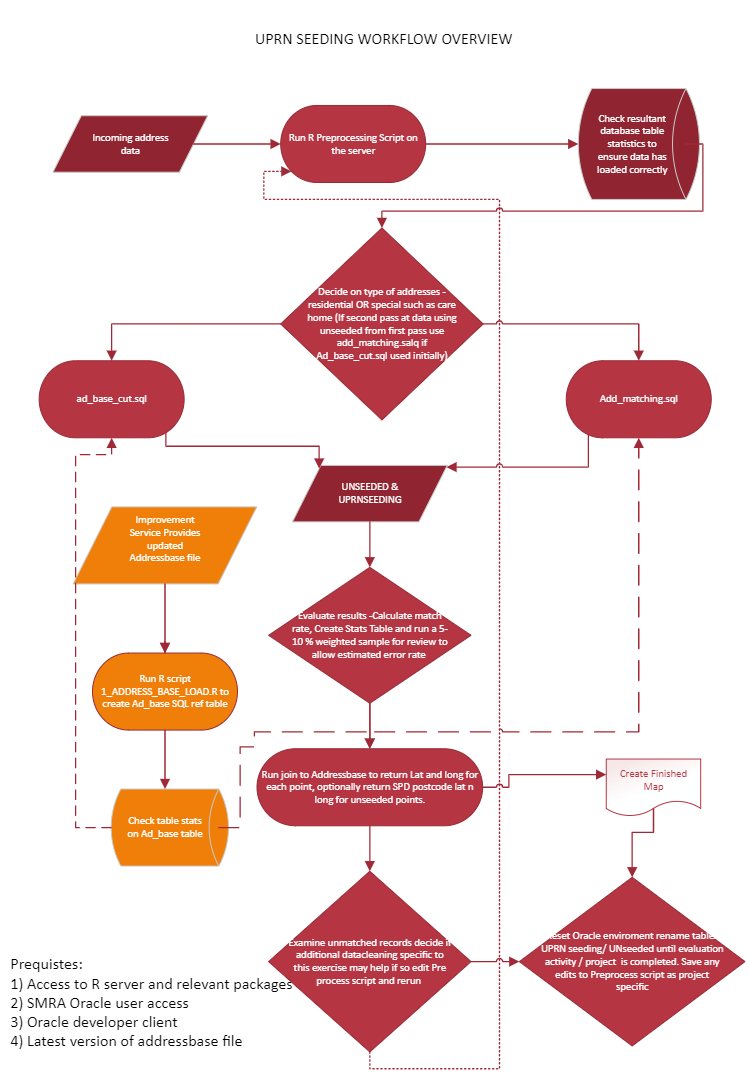
# Requirements

* Access to Martir03.ad\_base oracle table. Granted access to Data Management and GIS team so far.
* R script to clean and upload incoming data
* Oracle script to run the seeding
* An r session to collect and evaluate the results
* Time to sample and understand process for estimating error rates.

# Improvements

* Consider most optimal way to get unmatched manually reviewed
* Consider most optimal way to establish likely error rate for any job or find errors – most likely stratified review

# Appendix 1 – UPRN Workflow Overview



## Creation / updating the UPRN Reference File

Our reference file for UPRN seeding is the latest Addressbase file.

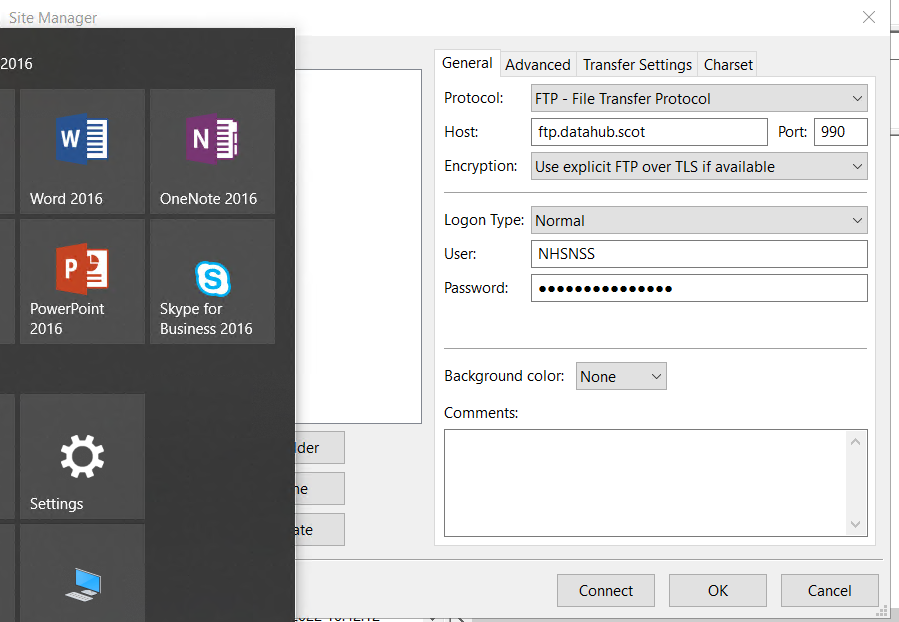
Address base is updated every 6 weeks in line with updates supplied by the Improvement Service.

Accessing the update is via an ip address from inside Meridian Court or Gyle which are white-listed.

You can access the ftp server :

Username: **NHSNSS**

Password: **RAMU968mif$PH60**



In the “out” folder setup for this user the new update will appear every 6 weeks.

It comes as a compressed file naming convention like this: Scottish\_Address\_Data\_NHSNSS\_EPOCH95\_2022-09-14 Epoch and date increments with new files.

If you are working remotely you can RDP to the PC : NSS152228 this will allow collection of files.

# Appendix 2 – AD\_BASE defination

The Addressbase file is processed using an R script : h3\_matching\_SQL\_TABLE\_BETA2.R at "cl-out\Martin\_R\UPRN Seeding\h3\_matching\_SQL\_TABLE\_BETA2.R".

The purpose of the above script is to read in addressbase from the flat file supplied and using UPRN coordinates to assign H3 tiles at several levels 4,6,7,8 and 10.

Filtering is applied to remove proposed developments, non-residential and addressable objects we won’t need such as ATMs and other infrastructure we don’t want to search for.

The addresses are standardised to remove Address base conventions you don’t see in real life addresses. Whole address line objects are created for later comparison. Numeric elements of addresses are separated as this is often the differentiating factor between many addresses as this is a key metric.

The USRN dataset is cleaned, filtered for relevant roads to remove slip roads, motorways and other things like roundabouts, footpaths and cycleways etc. Next the street abbreviation words file is read in and joined on the basis of street type. E.g., avenue or road are street types and the associated abbreviation are assigned such as road / rd and avenue ave.

Next a record is created for the unabbreviated streets and any associated abbreviations are added to help with joining later. This USRN dataset can be joined to the Addressbase dataset using the USRN. All this goes up to the SQL table and further variables are derived on the way.

|  |  |  |  |
| --- | --- | --- | --- |
| Addressbase file variables | Distinct Count | Average length | Comments |
| MIGRATION\_DATE | 1 | 11 | Date of last epoch update loaded |
| UPRN | 3019776 | 11 | Not unique due to different sources in addressbase |
| PARENT\_UPRN | 2398720 | 11 | Extra logic if not parent then make uprn parent for logic in analytic queries |
| SUB\_UNIT | 97584 | 6 | Regex to generate unit, flat or room number |
| BUILD\_NO | 19240 | 4 | Street number |
| ADD\_NUM | 235008 | 5 | All numerical elements together with any intervening words |
| ADDRESS\_LINE\_1 | 2036096 | 15 | Address text |
| ADDRESS\_LINE\_2 | 162560 | 10 | Optional address text |
| ADDRESS\_LINE\_3 | 17130 | 8 | Optional address text |
| ADDRESS\_STRING\_NOPC | 3915264 | 34 | Concatenation of line 1-3 minus nulls |
| POSTCODE | 161104 | 9 | Full postcode |
| USRN | 96792 | 9 | Unique street index |
| DESCRIPTION | 49760 | 14 | Plain English name of street |
| ABBR\_STREET1 | 48188 | 12 | Street abbreviation 1 |
| ABBR\_STREET2 | 29120 | 9 | Street abbreviation 2 |
| ABBR\_STREET3 | 1095 | 6 | Street abbreviation 3 |
| POST\_TOWN | 1750 | 10 | Name of town relating to the address |
| TOWN | 1620 | 9 | NRS town name from Settlement and localities data |
| LOCALITY | 1397 | 7 | Locality name form NRS |
| SETTLEMENT\_NAME | 509 | 13 | Alternative place name |
| ISLAND | 1 | 5 | Name of island |
| LONGITUDE | 363584 | 9 | Numeric coordinate |
| LATITUDE | 45592 | 8 | Numeric coordinate |
| H3\_4 | 104 | 16 | H3 4 tile code based on lat n long above |
| H3\_6 | 2660 | 16 | H3 6 tile code based on lat n long above |
| H3\_7 | 11370 | 16 | H3 7 tile code based on lat n long above |
| H3\_8 | 43172 | 16 | H3 8 tile code based on lat n long above |
| H3\_10 | 254384 | 16 | H3 10 tile code based on lat n long above |
| WORD\_ONE | 78208 | 9 | Initial word post numerical elements removal |
| WORD\_ONE\_M | 36088 | 6 | Metaphone of above |
| WORD\_TWO | 24706 | 7 | 2nd word post numerical elements removal |
| WORD\_TWO\_M | 13753 | 5 | Metaphone of above |
| WORD\_THREE | 21766 | 9 | 3rd word post numerical elements removal |
| WORD\_THREE\_M | 13147 | 6 | Metaphone of above |
| WORD\_FOUR | 13691 | 7 | 4th word post numerical elements removal |
| WORD\_FOUR\_M | 9222 | 6 | Metaphone of above |
| WORD\_FIVE | 8446 | 6 | 5th word post numerical elements removal |
| WORD\_FIVE\_M | 6021 | 6 | Metaphone of above |
| WORD\_SIX | 5371 | 6 | 6th word post numerical elements removal |
| WORD\_SIX\_M | 4072 | 6 | Metaphone of above |
| CLASSIFICATION\_TERTIARY | 108 | 22 | Addressbase classification for this record |

# Appendix 2 Evaluation

* Stages matches got agreed -stats table
* Errors found and error ratio – weighted review
* Addressed errors – any changes – potential to adjust parameters and re run
* Overall error rate – error rate – useful to weed records if exercise is small or establish accuracy in larger exercise

Improvement Service got 883378 of 1156948 individual records in the dataset or 63%

Oracle got 1116886 matches or 96% of 1156948.

Agreement 847654 between oracle and Improvement Service system of 96% of the matches Improvement Service got oracle got too.

Disagreement at UPRN level 23961 or 2% of total looks like worth checking at

parent UPRN level higher match rate here. They don’t match at UPRN level AT parent\_uprn match level a further 4000 or so match leaving 19863 not in agreement or 1.7% of total matches.

ORACLE process got 365464 extra hits over Improvement Service system or 31.5% extra. This is where we need to check the 2% disagreement above.

# Appendix 3 – Data Quality.

Sometimes an address string can refer to a range of addresses, e.g. 115-120 high street.

Buildings get knocked down, and then rebuilt and renumbered, and merged or split into smaller units so subunits can be confusing.

Things can be the same on the ground but renumbered. So what was flats 1,2,3,4 could become a,b,c,d. In short, what seems easy at first is in fact not as straightforward, nor it is easy to judge accuracy. I tried to code for the base case or the correct usage. Edge cases get handled as late as possible. I generally don’t aim for 100% match as this is normally false even with very good data. Change is the only constant - Addressbase is updated 6 weekly. Results in theory then aren’t 100% repeatable with updated data.

## Error rate tracking

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Script used | Job name | Overall Error rate (Estimated) | Number of records | Overall Match Rate | Found errors | Sample size | Stages & stage error rates |
| Nov -Care home script | CHR dataset Carehomes | 1.1% | 15534 | 83% | 11 | 5% | 55 -  83 -  86 -  93 - |
| Nov -Care home script | CI dataset | 0.4% | 1701 | 72.1%  1225 | 6 | 5% | 28-1  83-2  88-1  89-1  92-1 |
| Beta 1 | OPTICIANS | 0.97% | 719 | 85.9%  618 | 10 | 100% | 25-3,  78-2,  82&83&85-1,86-2 |
| Beta 1 | OVER 65S \* | 2.79% | 1156948 | 96% | 32,286 | 5% | TO FIND OUT |
| Dec -Care home script | CHR dataset Carehomes |  |  |  |  |  |  |
| Dec -Care home script | CI dataset |  |  |  |  |  |  |
| Jan-Care home script | CHR dataset Carehomes |  |  |  |  |  |  |
| Jan -Care home script | CI dataset |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |