TRANSFORMING LOCATION DESCRIPTIONS TO SPATIAL DATA: DEVELOPING A CUSTOMIZED CANADA-WIDE COMPOSITE GEOCODER

Project Documentation File

PROJECT #2009, TPS2020

Ben Colgan, *Environmental Technologist & Cartographic Specialist,* Project Manager
Mara Van Meer, *Ecosystem Management Technologist & Applications Specialist,* Communications Manager

ADVISOR

Kendra Chalmers, *Professor*,
Fleming College

CLIENTS

Tyler Munn, *Senior GIS & Data Integration Specialist,*
Analytics and Innovation, Toronto Police Service

Gayathri Ganesan, *Analyst,*
Analytics and Innovation, Toronto Police Service

Submitted June 2020

School of Environmental & Natural Resource Sciences
Frost Campus | Fleming College

Fleming GIS

This final report is presented in partial fulfilment of the academic requirements for APST62, GIS Collaborative Project course, Fleming College.

Contents

List of Tables:

List of Figures:

1.0. Data Acquisition

The Canada-wide locator created as part of this project is comprised of several smaller locators. The order in which these locators were created and added to the composite prioritizes Toronto location descriptions followed by Ontario locations and finally Canada-wide locations. The locators included in the final master composite and their associated order are outlined in table 1.

Table 1: Locators included in master composite

ORDER IN COMPOSITE	LOCATOR
1	Toronto address points
2	Toronto street network
3	Toronto intersections
4	Ontario street network
5	Ontario intersections
6	Canada street network
7	Canada place names
8	National forward sortation area (FSA) centroid points

Road network data for each province and territory was sourced from the Canadian Open Data National Road Network (NRN). Ontario street network was sourced from the Ontario Road Network (ORN) with fields present for abbreviated city/ municipality and street name. Road network data are required for the creation of the street locators within the final composite, including Toronto streets, Ontario Streets, and Canada Streets. Ontario/ Toronto intersections were generated from the OBN Ontario data. Toronto address points were downloaded from the Toronto Open Data Portal. Common place names for Canada were downloaded from Open Street Map, while FSA information was sourced from statistics Canada.

Table 2: Data sources used to create composite geocoder, date last updated, and original projection information

FILE NAME	DATATYPE	SUMMARY	DATA LAYER SOURCE	LINK TO DOWNLOAD	DATE LAST MODIFED	ORIGNAL PROJECTION	FINAL PROJECTION
NRN_AB_14_0_ROADSEG	Line	This is a road network dataset for the province of Alberta. It was merged with provincial road data across Canada to produce Canada street network locator	Alberta National Road Network	https://open.canada.ca/data/en/dataset/05e0bd57-29c7-4d26-9e0f-9021212a9003	2018-06-27	GCS North American 1983 CSRS	GCS North American 1983 CSRS
NRN_YT_15_0_ROADSEG	Line	This is a road network dataset for Yukon. It was merged with provincial road data across Canada to produce Canada street network locator	National Road Network - NRN - Yukon - GeoBase Series	https://open.canada.ca/data/en/dataset/2e72388e-4815-44c0-ae92-ce252a712365	2020-04-08	GCS North American 1983 CSRS	GCS North American 1983 CSRS
NRN_NU_7_0_ROADSEG	Line	This is a road network dataset for Nunavut. It was merged with provincial road data	National Road Network - NRN -	https://open.canada.ca/data/en/dataset/78104	2020-04-08	GCS North American 1983 CSRS	GCS North American 1983 CSRS

		across Canada to produce Canada street network locator	Nunavut - GeoBase Series	94c-5fed-4a10-849d-413833a36ab1			
NRN_PE_18_0_ROADSEG	Line	This is a road network dataset for the province of Prince Edward Island. It was merged with provincial road data across Canada to produce Canada street network locator	National Road Network - NRN - Prince Edward Island - GeoBase Series	https://open.canada.ca/data/en/dataset/4c1ae636-5c7c-41ab-a943-1f82ae3faf89	2020-04-08	GCS North American 1983 CSRS	GCS North American 1983 CSRS
NRN_NL_7_0_ROADSEG	Line	This is a road network dataset for the province of Newfoundland and Labrador. It was merged with provincial road data across Canada to produce Canada street network locator	National Road Network - NRN - Newfoundland and Labrador - GeoBase Series	https://open.canada.ca/data/en/dataset/56c8b232-bf88-4e51-b650-860dfcb86e9f	2020-04-08	GCS North American 1983 CSRS	GCS North American 1983 CSRS
NRN_NT_10_0_ROADSEG	Line	This is a road network dataset for Northwest Territories. It was merged with provincial road data across Canada to produce Canada street network locator	National Road Network - NRN - Northwest Territories - GeoBase Series	https://open.canada.ca/data/en/dataset/e81802cf-9bad-47b8-8d45-591921316c66	2020-04-08	GCS North American 1983 CSRS	GCS North American 1983 CSRS
NRN_QC_9_0_ROADSEG	Line	This is a road network dataset for the province of Quebec. It was merged with provincial road data across Canada to produce Canada street network locator	National Road Network - NRN - Quebec - GeoBase Series	https://open.canada.ca/data/en/dataset/f79e8e77-093c-4cbc-ac40-47b0ea0ade59	2020-04-08	GCS North American 1983 CSRS	GCS North American 1983 CSRS
NRN_MB_6_0_ROADSEG	Line	This is a road network dataset for the province of Mantobia. It was merged with provincial road data across Canada to produce Canada street network locator	National Road Network - NRN - Manitoba - GeoBase Series	https://open.canada.ca/data/en/dataset/08364fec-3770-44e7-b7cd-b4367e911a35	2020-04-08	GCS North American 1983 CSRS	GCS North American 1983 CSRS
NRN_NS_14_0_ROADSEG	Line	This is a road network dataset for the province of Nova Scotia. It was merged with provincial road data across Canada to produce Canada street network locator	National Road Network - NRN - Nova Scotia - GeoBase Series	https://open.canada.ca/data/en/dataset/caba16f1-0dd8-4db1-813b-c7bdd67122ef	2020-04-08	GCS North American 1983 CSRS	GCS North American 1983 CSRS

NRN_NB_9_0_ROADSEG	Line	This is a road network dataset for the province of New Brunswick. It was merged with provincial road data across Canada to produce Canada street network locator	National Road Network - NRN - New Brunswick - GeoBase Series	https://open.canada.ca/data/en/dataset/1fa38552-f819-43ff-ac60-113bd10ddb49	2020-04-08	GCS North American 1983 CSRS	GCS North American 1983 CSRS
NRN_BC_14_0_ROADSEG	Line	This is a road network dataset for the province of British Columbia. It was merged with provincial road data across Canada to produce Canada street network locator	National Road Network - NRN - British Columbia - GeoBase Series	https://open.canada.ca/data/en/dataset/fd1edf8b-f06d-4074-a27d-3e2c36d905da	2020-03-04	GCS North American 1983 CSRS	GCS North American 1983 CSRS
NRN_SK_10_0_ROADSEG	Line	This is a road network dataset for the province of Saskatchewan. It was merged with provincial road data across Canada to produce Canada street network locator	National Road Network - NRN - Saskatchewan - GeoBase Series Road Network File - 2010 - Saskatchewan	https://open.canada.ca/data/en/dataset/23ba8fb5-41f9-47d8-8763-6f69ec2f55cc	2019-01-01	GCS North American 1983 CSRS	GCS North American 1983 CSRS
ADDRESS_POINT_WGS84	Point	This data set is the municipal data points for the city of Toronto for use in the Toronto address points locator	Toronto Address Points	https://open.toronto.ca/dataset/address-points-municipal-toronto-one-address-repository/	2018-12-31	WGS 84	GCS North American 1983 CSRS
Fsa_polygons	Polygon	This dataset is FSA polygons across Canada for use in the FSA locator	FSA Canada	https://www12.statcan.gc.ca/census-recensement/2011/geo/bound-limit/bound-limit-2016-eng.cfm	2019-04-15	PCS Lambert Conformal Conic	GCS North American 1983 CSRS
Ontario_Road_Network_ORN_Segment	Line	This data set is Ontario roads, for use in the Toronto street network locator, Ontario street network locator, Toronto intersections locator, and Ontario intersections locator	Ontario Road Network (ORN) Segment With Address	https://geohub.lio.gov.on.ca/datasets/mnrf::ontario-road-network-orn-segment-with-address?ge	2020-03-17	GCS WGS 1984	GCS North American 1983 CSRS

				ometry=-168.189%2C38.917%2C-1.285%2C58.786			
Alberta_Placenames, BC_PLacenames, Manitoba_Placenames, NB_Placenames, Newfoundland_Placenames, NS_Placenames, Nunavut_Placenames, NWT_Placenames, Ontario_Placenames, PEI_Placenames, Quebec_Placenames, Saskatchewan_Placenames, Yukon_Placenames	Point	This data can be downloaded from one webpage, containing point feature classes of common place names in Canada	Canada Place Names - Places point data by province	http://download.geofabrik.de/north-america/canada.html	2020-05-29 (Updated continuously by OpenStreet Map users)	WGS 84	GCS North American 1983 CSRS

2.0. Database Design Plan

Dividing the locators outlined in Table 1 between team members allocated the responsibility of data organization, automation, and documentation for each locator. Skype for business with screen share was used frequently throughout the project for the team to share the creation of locators and receive input and advice from one another. Table 3 outlines how locator responsibility was divided within the team and the names of the resulting project files/ data bases. Locators are stored outside of the geodatabase, and therefore were passed to and from team members using sharing services including Microsoft teams, OneDrive, and Dropbox.

Table 3: Allocated locator responsibility and location of associated data files

AUTHOR	LOCATOR	FEATURE CLASS LOCATIONS	PACKAGED PROJECT FILE
Mara Van Meer	Toronto address points	TPS2020_2009_Source.gdb, TPS2020_2009_Scratch.gdb, TPS2020_2009.gdb	TPS2020_2009.ppkx
	Toronto street network		
	Ontario street network		
	Canada street network		
	FSA locator		
Ben Colgan	Toronto intersections	LIO_Intersections.gdb, LIO_Intersections_scratch.gdb, Placenames.gdb, Placenames_Scratch.gdb	TPS2020_2009_TWO.ppkx
	Ontario intersections		
	Canada Place names		

3.0. Building Toronto Address Points Locator (Model 1.0. Toronto Address Points)

The final composite locator was composed of several locators, filtering from Toronto City Addresses to Canada-wide streets. Toronto is where most of the locating takes place, so the first locator in the composite was Toronto address points, ADDRESS_POINT_WGS84 (Table 2) dataset. The creation of this locator references Model 1.0. AddressLocator within the packaged ArcGIS Pro project "TPS2020_2009" as well as field calculator ". cal" files found within the "Field Calculator Scripts" folder.

3.1. Step 1: Projecting the Required Data

The first pre-processing step required was projecting the municipal Toronto Address layer into the correct projection, GCS_North_American_1983_CSRS, as requested by the client. As shown in Figure 1, the original projection of this data was WGS84.

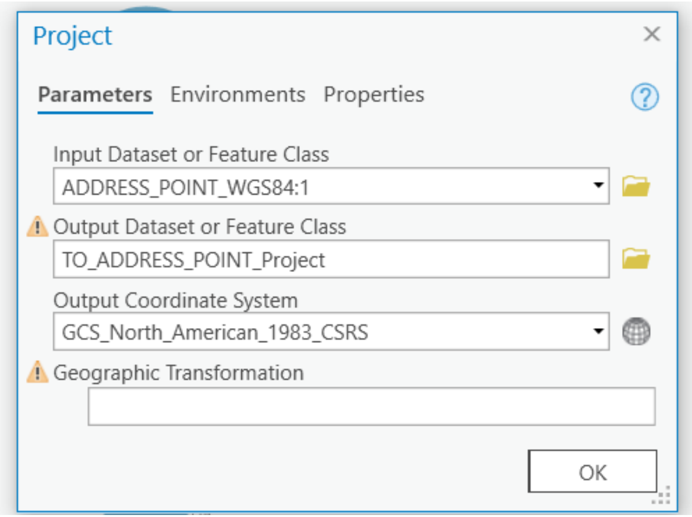
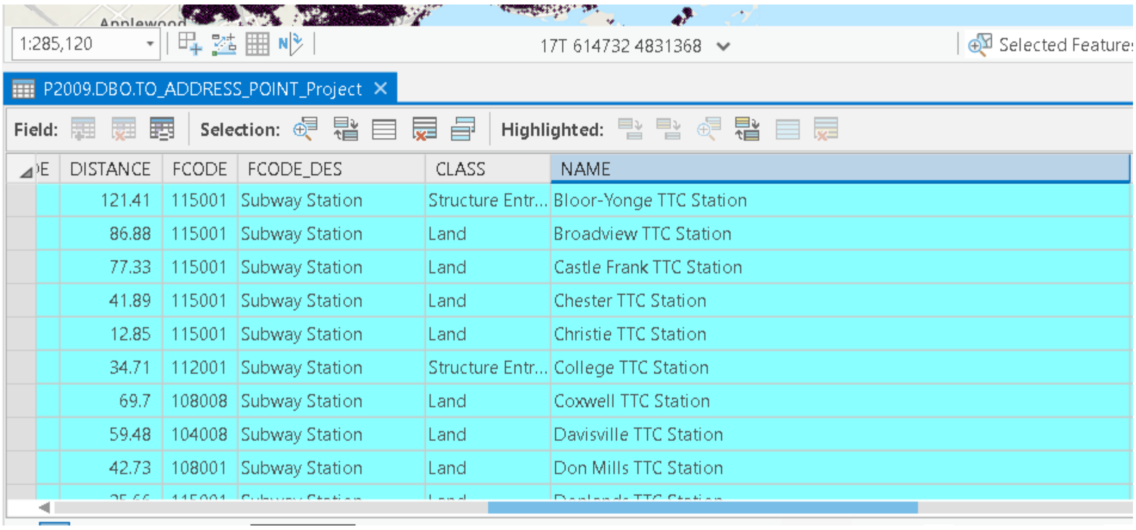


Figure 1: Projecting the Toronto Addresses Feature class into GCS_North_American_1983_CSRS

3.2. Step 2: Formatting Data to Increase Locator Accuracy

To make this locator increasingly adaptable and accurate, we started by adding the Toronto Transit Commission (TTC) stations within the City of Toronto to the 'Name' field of the Toronto Address points feature class. Other common places, such as the CN Tower, the Eaton Centre, and Ripley's Aquarium were also identified in the 'Name' field. As a result, addresses can be searched by the name field, allowing the use of common names for geolocating. TTC stations were specifically added to address business needs of TPS.



DISTANCE	FCODE	FCODE_DES	CLASS	NAME
121.41	115001	Subway Station	Structure Entr...	Bloor-Yonge TTC Station
86.88	115001	Subway Station	Land	Broadview TTC Station
77.33	115001	Subway Station	Land	Castle Frank TTC Station
41.89	115001	Subway Station	Land	Chester TTC Station
12.85	115001	Subway Station	Land	Christie TTC Station
34.71	112001	Subway Station	Structure Entr...	College TTC Station
69.7	108008	Subway Station	Land	Coxwell TTC Station
59.48	104008	Subway Station	Land	Davisville TTC Station
42.73	108001	Subway Station	Land	Don Mills TTC Station

Figure 2: Adding common names of specific addresses in the name field

To populate this field with accurate common names for TTC stations, the FormatAddress1.6.xlsm (Figure 3) macro used by TPS to format addresses for the current locator was used to identify the addresses of TTC stations within the City of Toronto. The Select by Attributes tool was used within ArcGIS pro to identify address points within the feature class designated as "Subway Stations" or "Other Transportation Facility". Once the common name was added to the address, the FCODE_DES was changed to "Subway Station" to ensure consistency and ensure simpler querying in the future. The addresses stated in the macro were compared with the location of TTC stations on Google Maps to ensure accuracy. Any adjustments made to TTC locations was documented in the FormatAddress1.6.xlsm document. Additionally, the TTC stations build as part of the 2017 expansion from Sheppard West to Pioneer Village were added to the dataset. The remaining stations, Highway 407 and Vaughan Metropolitan Centre were not included as they are not in the jurisdiction of the City of Toronto, and therefore they are no points within the Toronto Address points feature class that represent these locations.

A	B
Keele	21 Keele Street
Eglinton	2190 Yonge Street
Chester	22 Chester Avenue
Dundas West	2365 Dundas Street West
Lawrence East	2444 Lawrence Avenue East
Kennedy	2455 Eglinton Avenue East
Dupont	263 Dupont Street
Runnymede	265 Runnymede Road
Old Mill	2672 Bloor Street West
Scarborough Centr	290 Borough Drive
College	3 Carlton Street
Dundas	3 Dundas Street East
King	3 King Street East
Queen	3 Queen Street East
Donlands	30 Donlands Avenue
Lawrence	3101 Yonge Street
Main Street	315 Main Street
High Park	35 Quebec Avenue
Coxwell	355 Strathmore Boulevard
York Mills	4015 Yonge Street
St Patrick	449 University Avenue
Christie	5 Christie Street
North York Centre	5102 Yonge Street
Union	55 Front Street West
Bayview	550 Sheppard Avenue East

Figure 3: TTC stations and associated addresses within macro sent by TPS

Additionally, we found that the resulting locator was increasingly accurate when the city input of the create locator tool was populated. However, the City field within the address points layer is populated with "City of Toronto", which is not how the city is usually described when geocoding. An alternate table titled "TorontoAddress_Alias" was created using the Table to Table tool, and a "City_Shortened" field was added and populated (Figure 4). The municipality name (eg. "North York") was used as the main city/ municipality name within Toronto addresses.

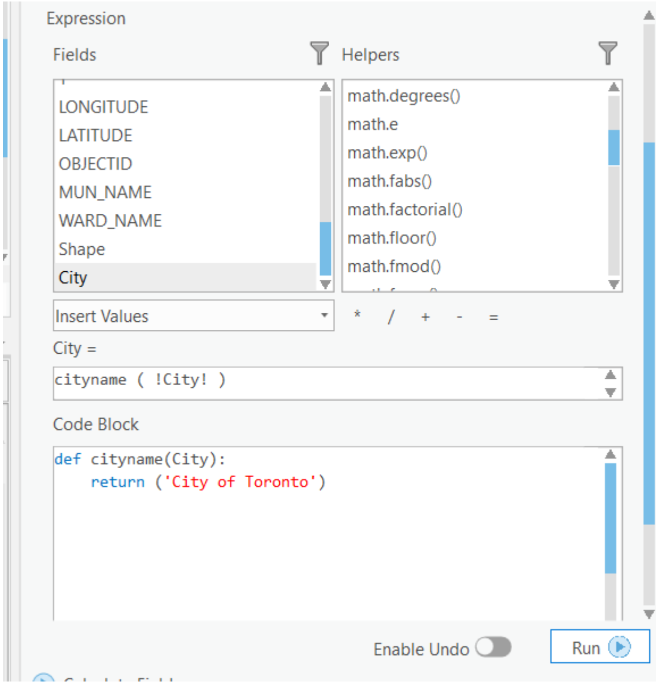


Figure 4: Code used to populate the City field within the AddressPoints feature class

3.4. Step 3: Creation of Toronto Address Points Locator

The primary table used to create this locator was the modified TorontoAddress, within the "TSP2020_2009.gdb" databases in the packaged project "TSP2020_2009". The role of this locator was as an address locator. The fields populated to create this locator are shown in Table 4. For all locators created as part of this composite, the County or Region was set to Canada.

Table 4: Create Locator tool inputs used to create Toronto Address Points locator

FIELD MAPPING NAME	CORRESPONDING FIELD
Address Join ID	OBJECTID_1
House number	ADDRESS
Building name	NAME
Street Name	LFNAME
City/ Municipality	MUN_NAME
Language Code	English
Optional Parameters	
Alternate Table Name	TorontoAddress_Alias
Alternate Table Role	Alternate City/ Municipality
City/ Municipality Join ID	OBJECTID_1
City/ Municipality	City_Shortened

4.0. Street Locators: Toronto, Ontario

The workflow for the creation of the Toronto street network locator and Ontario street network locator was similar. This section will discuss the pre-processing required for each locator, as well as the creation of alias tables. Models referenced include 2.0. TO Streets, 3.0. ON Streets within the ArcGIS Pro packaged project “TSP2020_2009”.

4.1. Step 1: Selecting Subsets from Ontario ORN Data

Initially, the process of creating the Toronto street network and Ontario street network was the same as outlined for the creation of the Canada street network locator. However, after switching to a more robust data source, the ORN data, creating the Toronto and Ontario street locators was significantly less complicated. The ORN data had to be projected to the correct coordinate system, as shown in Figure 5.

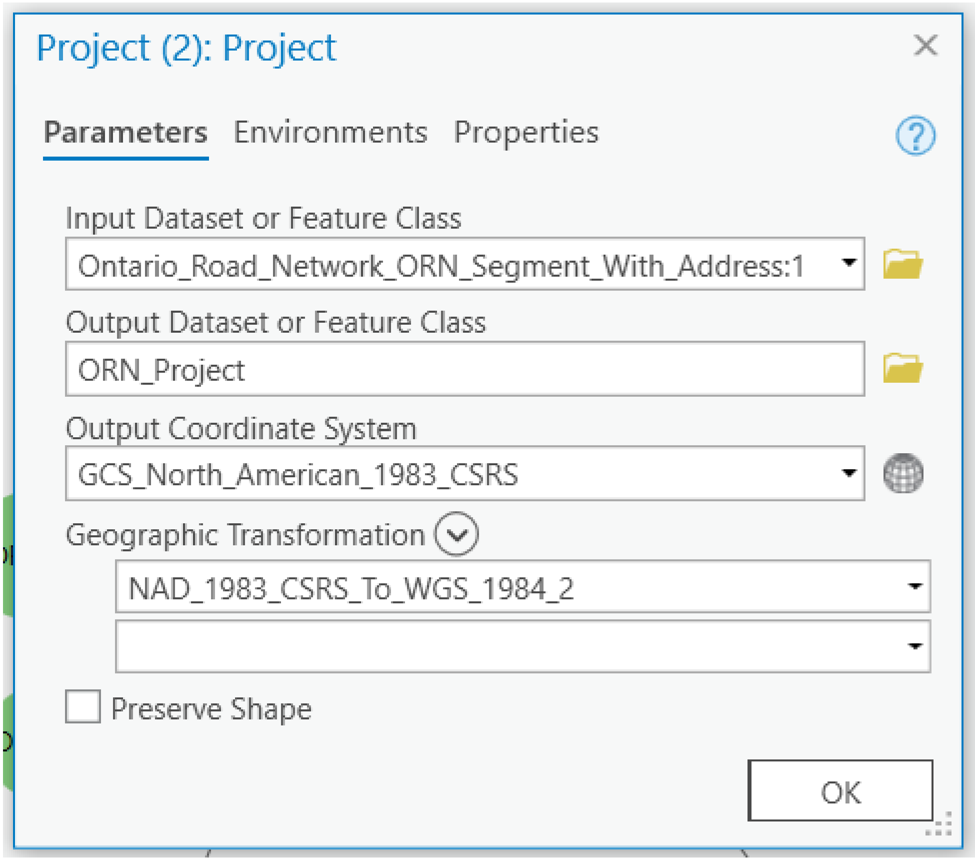


Figure 5: Projecting the ORN data for the province of Ontario to the correct geographic coordinate system

Second, the OntarioStreets and TorontoStreets subsets were created using the select by attribute tool, from the projected ORN data. as shown in Figure 6 to create the Toronto streets feature class. It was important that the exact opposite of this selection statement was used to create the Ontario dataset, to ensure that each road is only included in one locator. If both locators included the same road, it would display twice when geocoding.

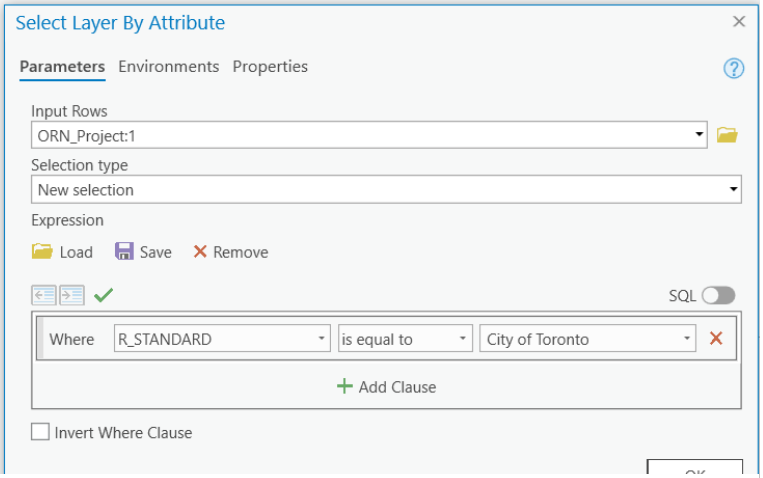


Figure 6: Extracting the TorontoStreets data

A few streets inside Toronto were not selected using the query present in Figure 6 if the road jurisdiction was the ministry of transportation (eg. Highway 401). However, the pre- processing of the Ontario street network used the opposite query, selecting all streets that are not equal City of Toronto (Figure 7) therefore any streets not included in the Ontario Streets locator were used as part of the Toronto streets locator.

After the selection was made, the feature class to feature class tool was used to create each street subset.

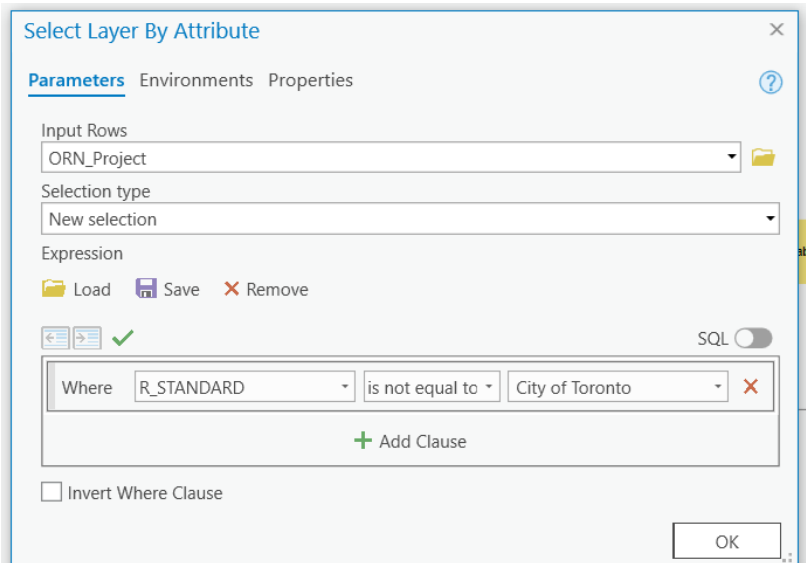


Figure 7: Extracting OntarioStreets subset

4.2. Step 2: Create Alias Tables

The table to table tool was then used to generate an alias table for each subset. The alias table was used in the production of the final locators to address abbreviated street names (Mary Street W vs. Mary St W) and shorted city names (eg. City of Quinte West vs Quinte West), allowing the final locator to geocode with both. Figure 8 displays in the table to table tool inputs to create the Ontario street network alias table.

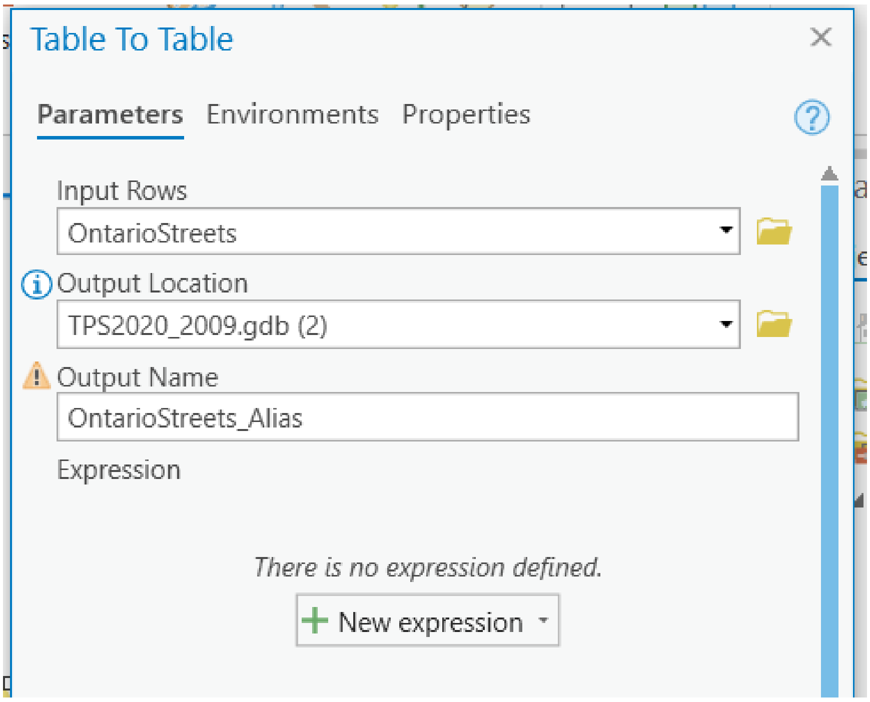


Figure 8: Table to table tool inputs used to create OntarioStreets_Alias table from the OntarioStreets subset. The same process was used for TorontoStreets

4.3. Step 3: Create Toronto Street Network & Ontario Street Network Locators

The primary table used to create the Toronto street network locator was the TorontoStreets subset, while the OntarioStreets subset was used to create the Ontario street network locator. Both were created as street locators and can be found within the "TSP2020_2009.gdb" database in the packaged project "TSP2020_2009". Table 5 below outlines specific inputs to create the final Toronto and Ontario street network locators.

Table 5: Inputs to the create locator tool to produce Ontario street network and Toronto street network locators

FIELD MAPPING NAME	CORRESPONDING FIELD FROM ORN FEATURE CLASS SUBSETS (TORONTO STREETS, ONTARIO STREETS)
Street Join ID	OBJECTID_1
Left House Number From	L_FIRST_HO
Left House Number To	L_LAST_HOU
Right House Number From	R_FIRST_HO
Right House Number To	R_LAST_HOU
Street Name	STREET_NAM
Suffix Type	STREET_T_1
Full Street Name	OFFICIAL_S
Left City/ Municipality	L_STANDARD
Right City/ Municipality	R_STANDARD
Language Code	English
OPTIONAL PARAMETERS	
Alternate Table Name	TorontoSteets_Alias, OntarioStreets_Alias, (respectively for each locator)

Alternate Table Role	Alternate Street Name Alternate City/Municipality
Street Join ID	OBJECTID_1
Street Name	ABBREVIATE
City/ Municipality Join ID	OBJECTID_1
City/ Municipality	L_ORGINAL

5.0. Canada Street Locator

The Canada street network locator was created using NRN data as opposed to ORN data, and therefore required more pre- processing and scripting to allow aliases for abbreviated street names and city/ municipality name. This section discusses how the Canada street network locator was created, referencing model 4.0 CAN Streets within the "TSP2020_2009" packaged ArcGIS Pro project.

5.1. Step 1: Pre-Processing for Canada Streets

After downloading all the NRN data for the provinces and territories, the pre-processing for the Canada streets layer was to merge all the road datasets outside of Ontario. This will allow the generation of a Canada-Wide streets locator excluding the Ontario Roads, which are part of the Ontario streets locator. Figure 9 displays the inputs to the merge tool that were used to create the merged dataset. Projecting the NRN data was not required because it was downloaded in the correct geographic coordinate system GCS North American 1983 CSRS.

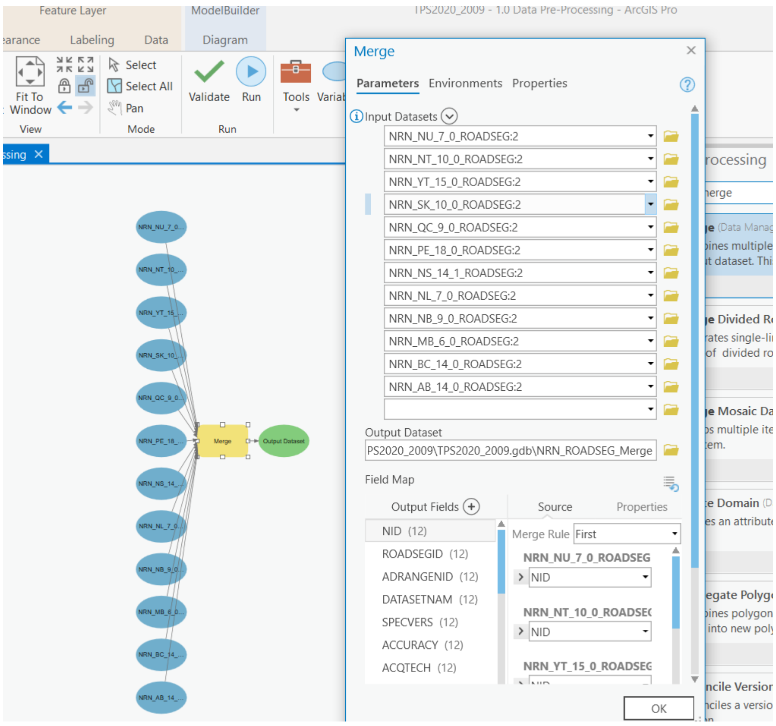


Figure 9: Merging Most Recent NRN Provincial Data to create a feature class of all roads outside of Ontario

5.2. Step 2: Creating and Populating Alias Table

An alias table was created for this dataset to ensure the final composite locator could process short form of the street names. For example, if the original street name within the street field was “Mary Street West”, the locator should geolocate to the same location given “Mary St W”. The ORN data had an “ABBREVIATE” field for this purpose, already populated. The use of python 3 scripts made it possible to create an abbreviated street name field for the NRN data, in a field called Alais.

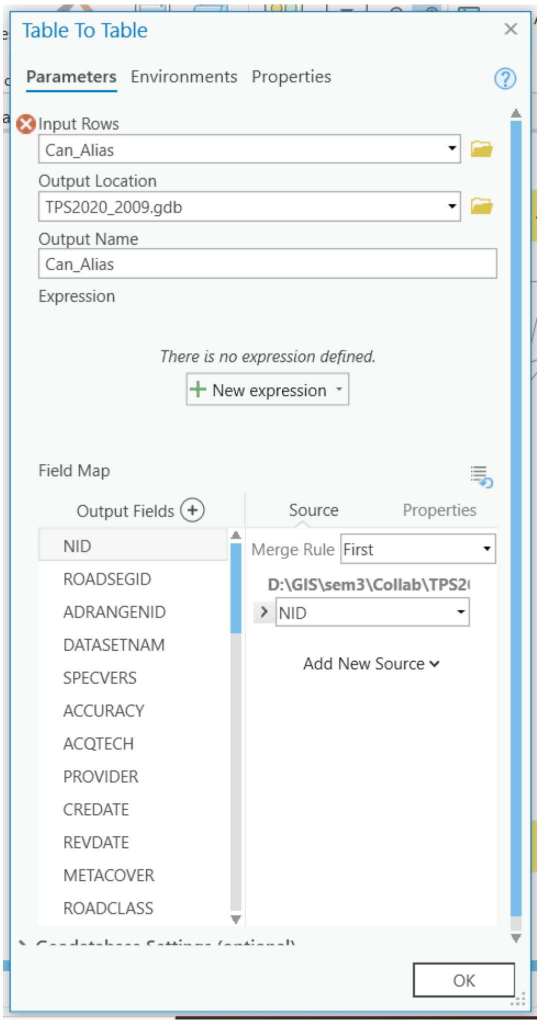


Figure 10: Creation of Can_Alias table

To create an alias table that could be used as an alternate table, the table to table tool was used on the merged Canada road network feature class (Figure 10). This tool is useful for creating alternate tables because it includes all records present in the feature class, connected via NID and OBJECTID.

Next, two fields were added to each alias table. The count field (Figure 11) was helpful when writing the script to convert the street name to its short form. The intention of this field was to count the number of words in a street name. Once populated, it was possible to sort the column descending to see the longest street name. This was important for generating the short from code to ensure that all street names would be shorted and moved to the alias field.

The second field added to the alias table was the alias field (Figure 12). This is the field that was used to house the short from of the street names to be used as an alternate street name within each street locator.

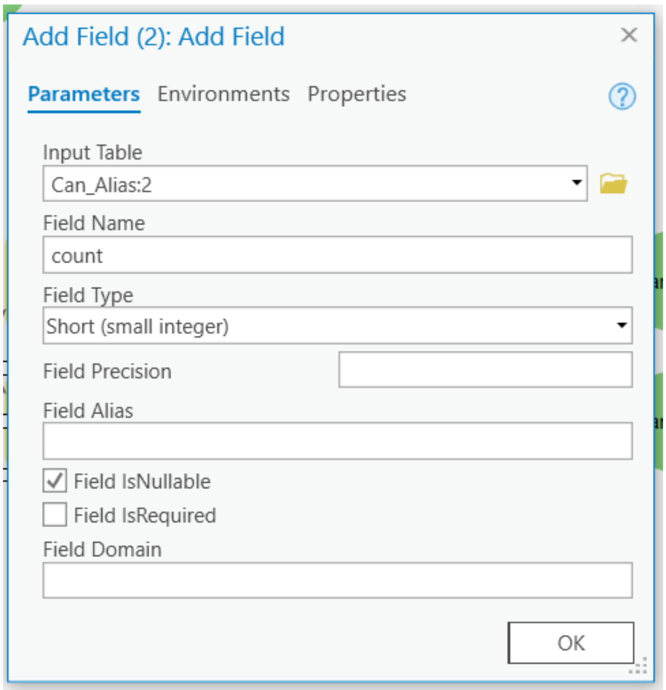


Figure 11: Adding the count field to the Can_Alias table

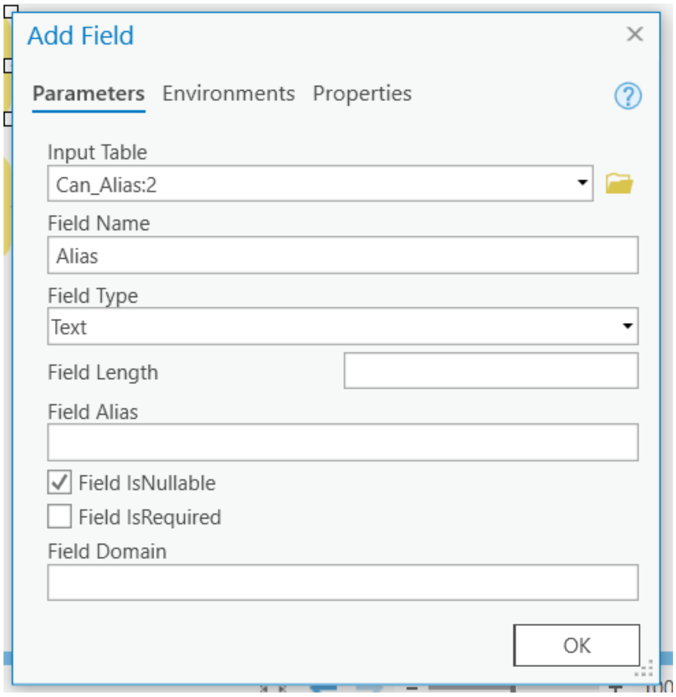


Figure 12: Adding the alias field to the Can_Alias table

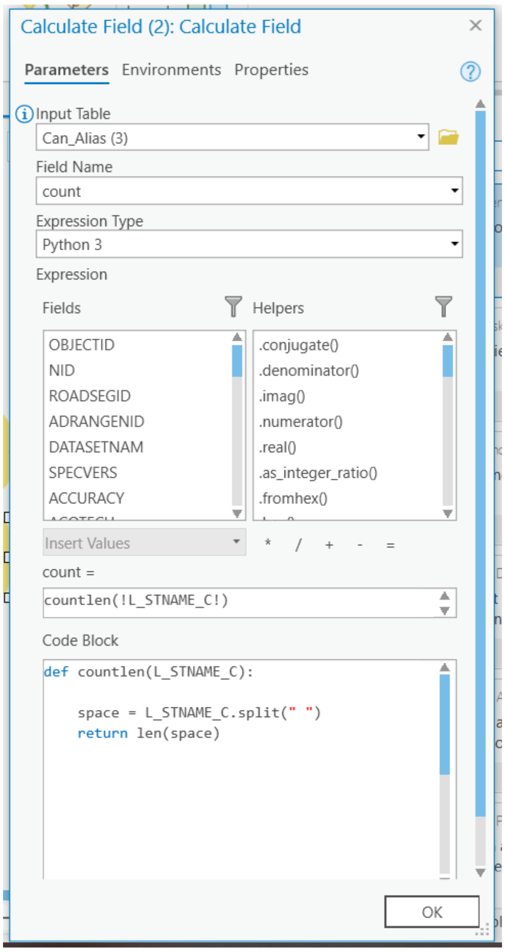
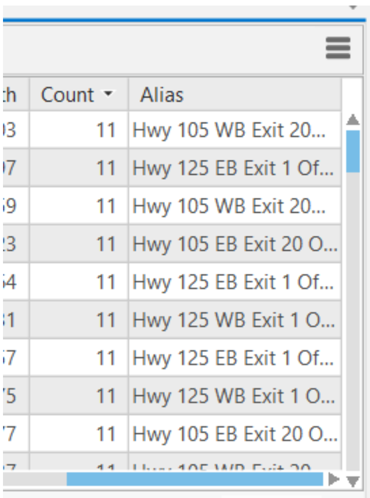


Figure 13: Script used to populate the count field within the Can_Alias table

The script present in the calculate field tool in Figure 13 was used to calculate the number of words in the street name field of each dataset. Once the count field was populated, it was sorted by in descending order, to see the maximum number of words used in a street name (Figure 14). This was helpful in creating the end of the short form code, pictured in Figure 15. The count field calculated the number of words for each field. Once calculated and sorted as descending, the maximum number of words for a street name within the dataset was found. The maximum number of words in a street name for the Canada Streets was 11, and therefore the if statement went to “if length==11”. Code was created with the aid of (Python Tips, 2017).



Count	Alias
11	Hwy 105 WB Exit 20...
11	Hwy 125 EB Exit 1 Of...
11	Hwy 105 WB Exit 20...
11	Hwy 105 EB Exit 20 O...
11	Hwy 125 EB Exit 1 Of...
11	Hwy 125 WB Exit 1 O...
11	Hwy 125 EB Exit 1 Of...
11	Hwy 125 WB Exit 1 O...
11	Hwy 105 EB Exit 20 O...

Figure 14: Sorting the count field descending to view longest street names in Canada

```
def alias(L_STNAME_C):
    space = L_STNAME_C.split(" ")


    for n, i in enumerate(space):
        if i == 'West':
            space[n] = 'W'
        if i == 'East':
            space[n] = 'E'
        if i == 'South':
            space[n] = 'S'
        if i == 'North':
            space[n] = 'N'
        if i == 'Alley':
            space[n] = 'Aly'
        if i == 'Avenue':
            space[n]= 'Ave'
        if i == 'Boulevard':
            space[n] = 'Blvd'
        if i == 'Crescent':
            space[n] = 'Cres'
        if i == 'Drive':
            space[n] = 'Dr'
        if i == 'Street':
            space[n] = 'St'
        if i == 'Road':
            space[n] = 'Rd'
        if i == 'Lane':
            space[n] = 'Ln'
        if i == 'Court':
            space[n] = 'Ct'
        if i == 'Trail':
            space[n] = 'Trl'
        if i == 'Place':
            space[n] = 'Pl'
        if i == 'Gateway':
            space[n] = 'Gtwy'
        if i == 'Gate':
            space[n] = 'Gt'
        if i == 'Hill':
            space[n] = 'Hl'
        if i == 'Grove':
            space[n] = 'Grv'
        if i == 'Gate':
            space[n] = 'Gt'
        if i == 'Circle':
            space[n] = 'Cir'
        if i == 'Highway':
            space[n] = 'Hwy'
        if i == 'Motorway':
            space[n] = 'Mtwy'
        if i == 'Parkway':
            space[n] = 'Pkwy'
        if i == 'Freeway':
            space[n] = 'Fwy'

    length = len(space)
    if length ==1:
        return space[0]
    elif length ==2:
        return space[0] + " " + space[1]
    elif length == 3:
        return space[0] + " " + space[1] + " " + space[2]
    elif length ==4:
        return space[0] + " " + space[1] + " " + space[2] + " " + space[3]
    elif length ==5:
        return space[0] + " " + space[1] + " " + space[2] + " " + space[3] + " " + space[4]
    elif length ==6:
        return space[0] + " " + space[1] + " " + space[2] + " " + space[3] + " " + space[4] + " " + space[5]
    elif length ==7:
        return space[0] + " " + space[1] + " " + space[2] + " " + space[3] + " " + space[4] + " " + space[5] + " " + space[6]
    elif length ==8:
        return space[0] + " " + space[1] + " " + space[2] + " " + space[3] + " " + space[4] + " " + space[5] + " " + space[6]+ " " + space[7]
    elif length ==9:
        return space[0] + " " + space[1] + " " + space[2] + " " + space[3] + " " + space[4] + " " + space[5] + " " + space[6]+ " " + space[7]+ " " + space[8]
    elif length ==10:
        return space[0] + " " + space[1] + " " + space[2] + " " + space[3] + " " + space[4] + " " + space[5] + " " + space[6]+ " " + space[7]+ " " + space[8]+ " " + space[9]
    elif length ==11:
        return space[0] + " " + space[1] + " " + space[2] + " " + space[3] + " " + space[4] + " " + space[5] + " " + space[6]+ " " + space[7]+ " " + space[8]+ " " + space[9]+ " " + space[10]

esri_field_calculator_splitter__

alias( !L_STNAME_C! )
```

Figure 15: The script used to populate the Alias field of the Can_Alias table. As the longest street name is 11 words, the if statement must include "elif length == 11:"

5.3. Step 3: Creating Canada Street Locator

The primary table used to create the Canada street network locator was the NRN merged dataset of all provinces and territories except for Ontario, titled " NRN_ROADSEG_MEGE". This locator was created as a street locator and the required datasets and table can be found within the "TSP2020_2009.gdb" database in the packaged project "TSP2020_2009". Table 6 below outlines specific inputs to create the Canada street network locator, while Figure 16 provides a snapshot of create locator inputs.

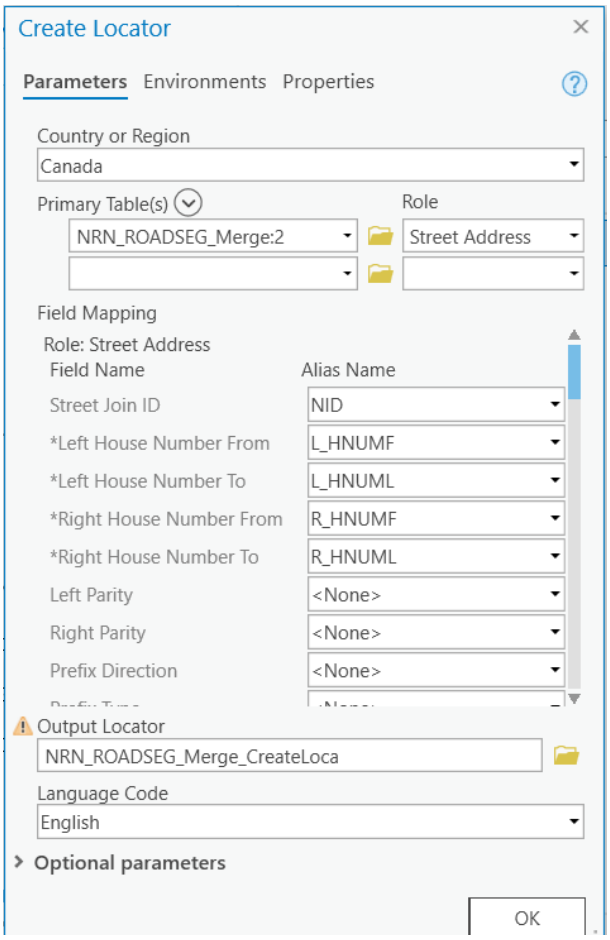


Figure 16: Creation of Canada Streets Locator

Table 6: Create Locator tool inputs used to create Toronto Streets, Ontario Streets, and Canada Streets locators

Field Mapping Name	Corresponding Field from NRN Dataset/ Subdataset
Street Join ID	NID
Left House Number From	L_HNUMF
Left House Number To	L_HNUML
Right House Number From	R_HNUMF
Right House Number To	R_HNUML
Street Name	L_STNAME_C
Left City/ Municipality	L_PLACENAM
Right City/ Municipality	R_PLACENAM
Left Province	DATASETNAM
Right Province	DATASETNAM
Language Code	English
Optional Parameters	
Alternate Table Name	Can_Alias
Alternate Table Role	Alternate Street Name
Street Join ID	NID
Street Name	Alias

6.0. Creating the Forward Sortation Area (FSA) Locator

6.1. Step 1: Pre-Processing for the FSA Locator

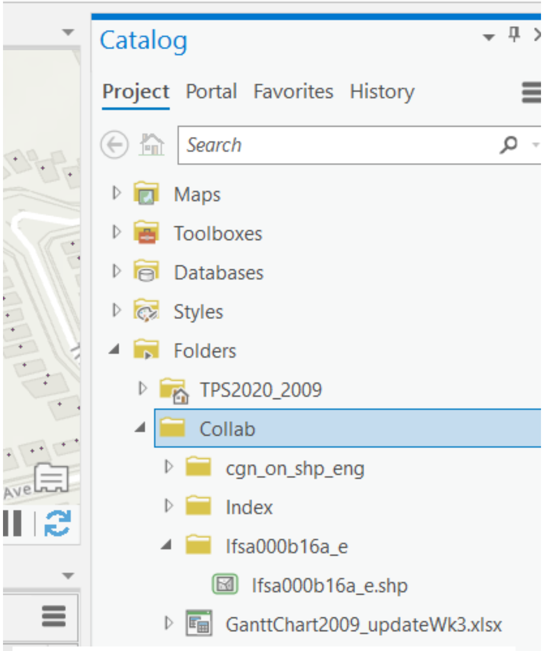


Figure 17: Adding FSA polygon data as a folder connection

The creation of the FSA locator references model 5.0 FSA locator within the ArcGIS Pro package "TSP2020_2009". To create the FSA locator, first the FSA polygon boundaries dataset was downloaded from statistics Canada (Table 1). This dataset was then added to the project as a folder connection to allow easy access to the polygon data (Figure 17). The original projection of the data did not match the rest of the data, GCS North American 1983 CSRS. As evident by Figure 18, the data was projected before the feature to point tool was used to find the centroid point of each polygon. The resulting shapefile is displayed in Figure 19.

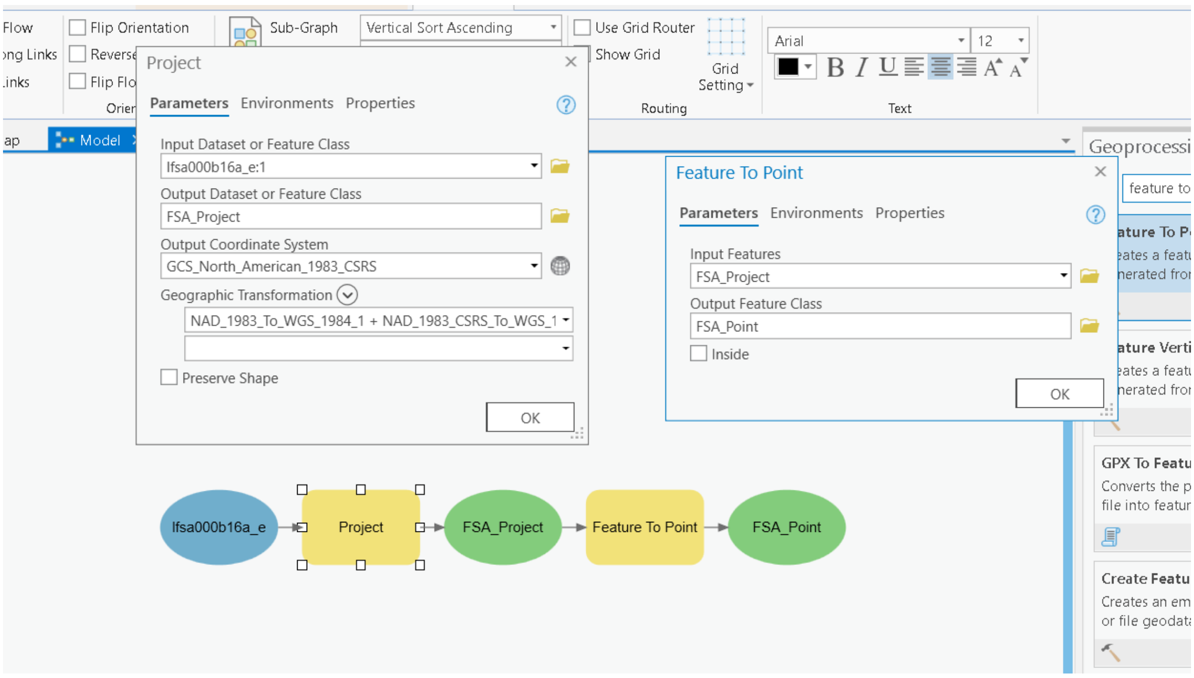


Figure 18: Resulting FSA point feature class after running feature to point tool

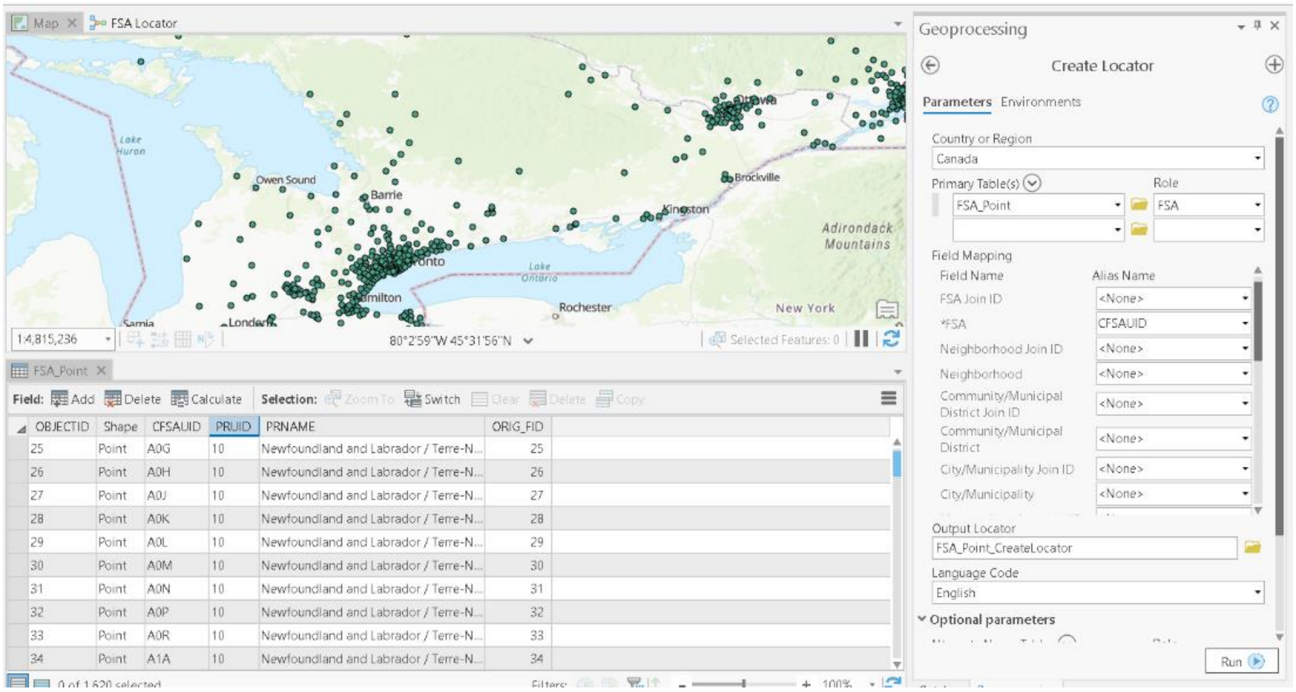


Figure 19: Projecting the FSA data and creating a point at the centroid of each polygon via the feature to point tool

6.2. Step 2: Creating the FSA Locator

To create the FSA locator, no alternate tables were required. The point feature class created in data-pre-processing was used as the primary table, the role being FSA. Refer to Figure 20 and Table 7 for specific tool inputs used to create this locator.

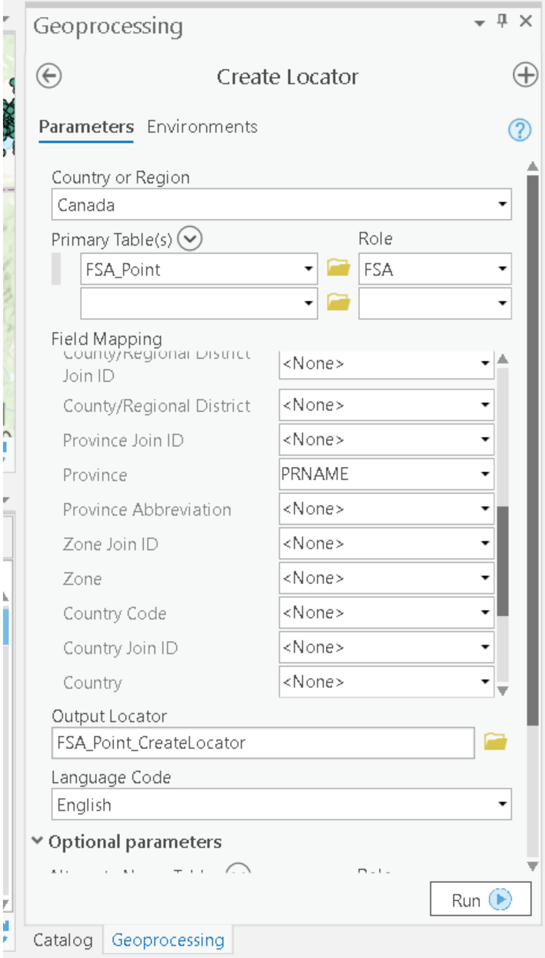


Figure 20: Create Locator tool used to create FSA locator

Table 7: Create Locator tool inputs used to create FSA Locator

Field Mapping Name	Corresponding Field from FSA Point Dataset
Primary Table	FSA_Point
Role	FSA
FSA	CFSAUID
Province	PRNAME
Language Code	English

7.0. Creating Intersection Geocoders (City of Toronto & Ontario)

This guide will reference model builder steps as found with the packaged ArcGIS Pro Project "*TPS2020_2009_TWO*" and field calculator ".cal" files found within the "Field Calculator Scripts" folder.

Both Ontario and Toronto Intersections were generated using the "Ontario Road Network (ORN) Segment With Address" dataset from the Land Information Ontario Geohub found here: https://geohub.lio.gov.on.ca/datasets/mnrf::ontario-road-network-orn-segment-with-address.

7.1. Step 1: Project to File Geodatabase (Model Builder Step 1.0)

The" Ontario Road Network (ORN) Segment With Address" shapefile is loaded into the "Project" tool. The output coordinate system is specified as NAD 1983 (CSRS) Geographic Coordinate System. The output is titled "LIO_Roads_Project".

7.2. Step 2: Intersection Creation (Model Builder Step 2.0)

***Note:** *This process is described in the article "More adventures in overlay: creating a street intersection list" by Dale Honeycutt, February 11, 2013 found here:* https://www.esri.com/arcgis-blog/products/arcgis-desktop/analytics/more-adventures-in-overlay-creating-a-street-intersection-list/

"LIO_Roads_Project".is loaded into the model where it hits the first tool "Unsplit Line". This tool removes extraneous breaks in the street line data. The dissolve field is set to "OFFICIAL_S", which is the street name field and the output feature class is titled "LIO_Roads_Unsplit".

"LIO_Roads_Unsplit" is then sent to the intersect tool. The incoming feature class is intersected with itself (only one input feature) and output type is set to point. See Figure 21 for tool parameters. The output feature class is titled "LIO_Roads_Intersect".

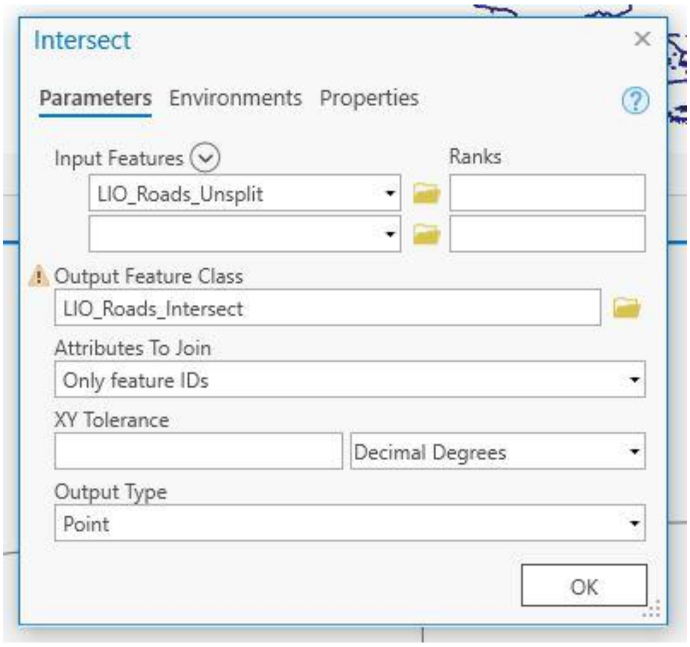


Figure 21: Intersect Tool Parameters in Model Builder Step 2.0

The last step is to use the “Spatial Join” tool to join street name data with the Intersection point data to create an attribute field with Intersections defined by street names. The output feature class is titled “LIO_Intersections_Named”. See Figure 22 below for tool parameters.

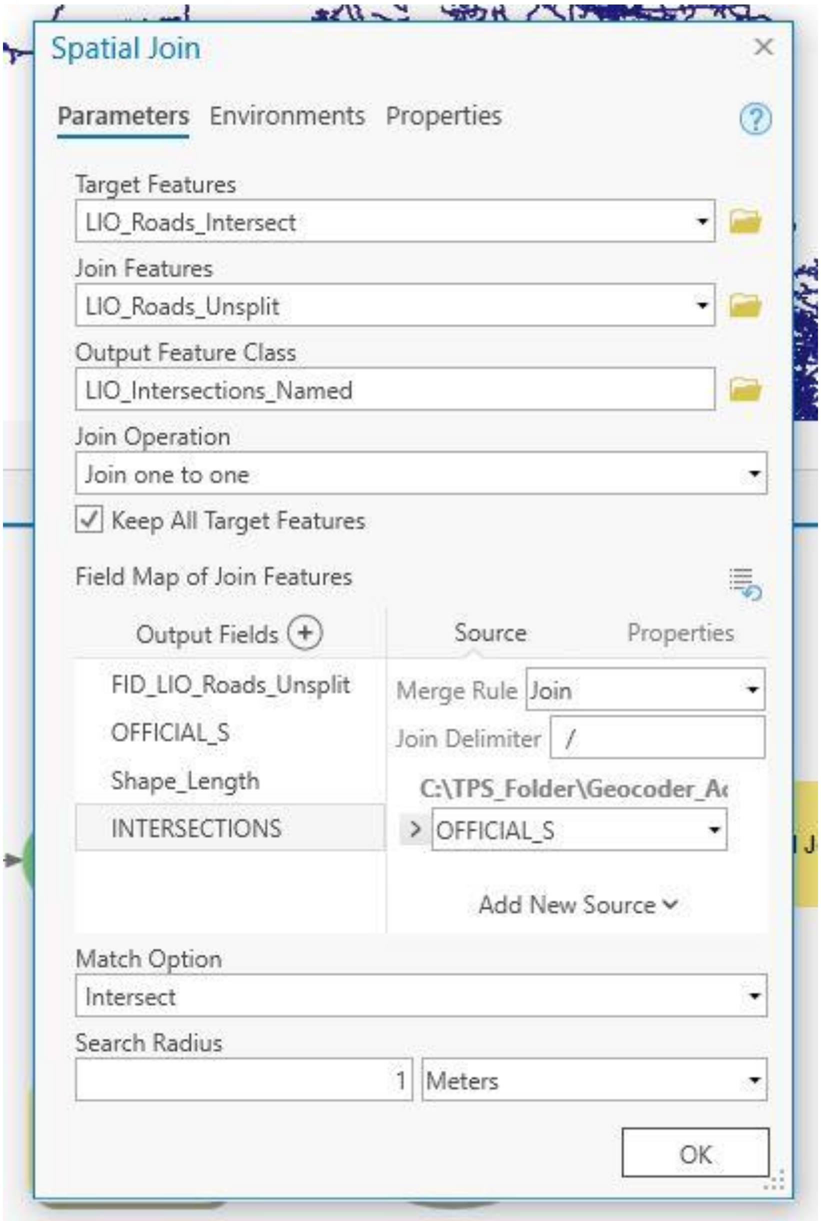


Figure 22:Spatial Join tool parameters in Model Builder Step 2.0.

The key to this step is to add a new field that will become the Intersection field and populate it. Click the + sign beside output fields and add a new field titled “INTERSECTIONS”. Under the “Source” tab, select “Join” as the merge rule and set the Join Delimiter to “ / “. Click “Add New Source” and from the “LIO_Roads_Unsplit” table select “OFFICIAL_S” as the source for populating the field.

7.3. Step 3: Intersection QAQC – (Model Builder Step 3.0)

The resulting "LIO_Intersections_Named" feature class is brought forward to the next step. A "Unique_Street_Count" field (Short Integer) is added and calculated using the "Unique_Street_Count.cal" script. This script returns the number of unique street names present in the intersection field giving a true count of the number of intersecting streets. Then the field "INTERSECTIONS_DUP_REMOVE" is added and calculated using the "Duplicate_Removal.cal" script. This removes any duplicate street names that were a carry over from the previous model and results in an accurate description of each intersection and contributing street.

***Note:** Code for the "Unique_Street_Count.cal" script was sourced from *"More adventures in overlay: creating a street intersection list" by Dale Honeycutt, February 11, 2013 found here:* https://www.esri.com/arcgis-blog/products/arcgis-desktop/analytics/more-adventures-in-overlay-creating-a-street-intersection-list/

The last step is to remove records that are not actually intersections. This is done by using the "Select" tool and selecting "WHERE "Unique_Street_Count" IS NOT EQUAL TO "1" and creating a new feature class. This removes records where streets intersect themselves, such as Cul-de-sacs. The resulting feature class is titled "LIO_Intersections_Named_USC1_Remove".

7.4. Step 4: Adding Index field – (Model Builder Step 4.0)

As a result of the process in Model Builder Step 1.0, multiple records are created for each intersection based on their street join count. This will be helpful in later populating the various ways that multi-street intersections can be described and these extra records will be used for that purpose.

First each duplicate needs to be indexed as a way to refer to which record will correspond with each unique description of the intersection.

This involves adding a new short integer field titled "Index" and using the "Calculate Field" tool and Python script "Main_Table_Index.cal". This field will add sets of sequential numbers (i.e., 1, 2 , 3 etc.) for each set of intersection locations based on their unique street count.

7.5. Step 5: Adding extra records – (Not documented in Model Builder)

In order to describe each multi-street intersection in each of the unique ways there must be **10** for every 5-street intersection, **6** records for every 4-street intersection and **3** records for every 3-street intersection.

For each intersection, the Intersection creation model in Step 1.0 creates a number of records equal the number of unique streets joining it (Join_Count). Therefore 3-street intersections already have an adequate amount of records for populating the unique descriptions while 4 & 5 street intersections do not.

7.5.1. Step 5.1 Adding records for 4-street intersections

- Select from "LIO_Intersections_Named_USC1_Remove" all records where Unique_Street_Count is equal to "4" and Index is equal to any number (select one)
- Make a subset of this data and call the resulting feature class "Extra_4" (using the Feature Class to Feature Class tool).
- Populate the index field of "Extra_4" with "5" using the "Calculate Field" tool.
- Merge the "Extra_4" feature class data set with the "LIO_Intersections_Named_USC1_Remove" data set using the "Merge" tool and name the resulting feature class "LIO_Intersections_Named_USC1_Remove_Merge1" (This adds a new set of 4-street intersection records with the index value of "5")
- Populate the index field of "Extra_4" feature class with "6" using the "Calculate Field" tool.
- Merge the newly edited "Extra_4" feature class data set with the "LIO_Intersections_Named_USC1_Remove_Merge1" data set and name the resulting feature class "LIO_Intersections_Named_USC1_Remove_Merge2" (This adds a new set of 4-street intersection records with the index value of "6")
- There are now six records for each 4-way street intersection.

7.5.2. Step 5.2 Adding records for 5-street intersections

There are only three 5-street intersections in the LIO dataset. Therefore, these index numbers were coded manually following this process:

- Manually code the Index field for all 5 records per 5-street intersections with "1 ,2, 3, 4, 5" respectively.
- Select all records where Unique_Street_Count equals "5".
- Make a subset of this data and call the resulting feature class "Extra_5" (using the Feature Class to Feature Class tool).
- Manually change the index numbers for each unique intersection to "6, 7, 8, 9, 10" respectively.
- Merge the "Extra_5" feature class data set with the "LIO_Intersections_Named_USC1_Remove_Merge2" data set using the merge tool and

name the resulting feature class "LIO_Intersections_Named_USC1_Remove_Full_Merge" (This adds a new set of 5-street intersection records with the index values

- There are now 10 records per 5-street intersection (10 is the required amount to cover all unique descriptions).

OFFICIAL_S	INTERSECTIONS	INTERSECTIONS_DUP_F	Index	Unique_Street_Count
COLONEL TALBOT RO...	COLONEL TALBOT RO...	COLONEL TALBOT RO...	1	5
COLONEL TALBOT RO...	COLONEL TALBOT RO...	COLONEL TALBOT RO...	2	5
COLONEL TALBOT RO...	COLONEL TALBOT RO...	COLONEL TALBOT RO...	3	5
COLONEL TALBOT RO...	COLONEL TALBOT RO...	COLONEL TALBOT RO...	4	5
COLONEL TALBOT RO...	COLONEL TALBOT RO...	COLONEL TALBOT RO...	5	5
HOPE STREET EAST	HOPE STREET EAST / H...	HOPE STREET EAST / H...	1	5
HOPE STREET EAST	HOPE STREET EAST / H...	HOPE STREET EAST / H...	2	5
HOPE STREET EAST	HOPE STREET EAST / H...	HOPE STREET EAST / H...	3	5
HOPE STREET EAST	HOPE STREET EAST / H...	HOPE STREET EAST / H...	4	5
HOPE STREET EAST	HOPE STREET EAST / H...	HOPE STREET EAST / H...	5	5
BRIDGE STREET EAST	BRIDGE STREET EAST /...	BRIDGE STREET EAST /...	1	5
BRIDGE STREET EAST	BRIDGE STREET EAST /...	BRIDGE STREET EAST /...	2	5
BRIDGE STREET EAST	BRIDGE STREET EAST /...	BRIDGE STREET EAST /...	3	5
BRIDGE STREET EAST	BRIDGE STREET EAST /...	BRIDGE STREET EAST /...	4	5
BRIDGE STREET EAST	BRIDGE STREET EAST /...	BRIDGE STREET EAST /...	5	5

Figure 23: Manually editing the Index field of the 5-street intersections records prior to making the "Extra_5" subset.

7.6. Step 6: Adding and populating Intersection_Options field – (Model Builder Step 6.0)

Now that each type of multi-street intersection has an adequate number of Indexed records it is possible populate a new field with all possible unique intersection descriptions.

Load "LIO_Intersections_Named_USC1_Remove_Full_Merge" into the "Add Field" tool create a text field titled "Intersection_Options". Then feed the data set into the "Calculate Field" tool and calculate using the "Populate_Options_Full.cal" script. The field is now populated with all possible unique descriptions of the multi-street intersections. 2-Street intersections are simply brought forward as is into the new field.

7.7. Step 7: Split Two-Way Intersections from Others (Model Builder Step 7.0)

At this point there are duplicate records for each two-way intersection, which is a result of the process in Step 1. This step is performed to prepare the data for deleting identical two-way intersection records using the Shape field. If the "Delete Identical" tool is used on the entire data set, multi-street intersections that have multiple records with the same X,Y location will also be removed necessitating splitting the data into subsets.

The "Select" tool is used to create subsets of the data with the following SQL expressions:

- Where Unique_Street_Count **is equal to 2** this generates a subset of all Two-way intersections. Name the resulting Feature Class “LIO_Intersections_Merged_2_Way”.
- Where Unique_Street_Count **is NOT equal to 2** this generates a subset of all other intersections. Name the resulting Feature Class “LIO_Intersections_Merged_345_Way”.

7.7.1. Step 7.1: Delete Identical from 2-Way intersections (Model Builder Step 7.1)

Load the “LIO_Intersections_Merged_2_Way” data set into the “Delete Identical” tool and select the “Shape” field as the delete parameter.

7.7.2. Step 7.2: Delete Identical from 2-Way intersections (Model Builder Step 7.2)

Load the “LIO_Intersections_Merged_2_Way” and the “LIO_Intersections_Merged_345_Way” data sets into the “Merge Tool” to re-join all intersection records. Name the resulting Feature Class “LIO_Intersections_Re_Merged”.

7.8. Step 8: Multi-point to Point (Model Builder Step 8.0)

Up to this point the data have been Multi-Point features. Multi-point features are not compatible with the “Create Locator” tool and as such the data need to be changed to Point data.

The “LIO_Intersections_Re_Merged” data set is loaded into the “Feature to Point” tool. Be sure to check the “Inside” box to ensure correct locations of the point data are retained. Name the resulting Feature Class “LIO_Intersections_FTP”.

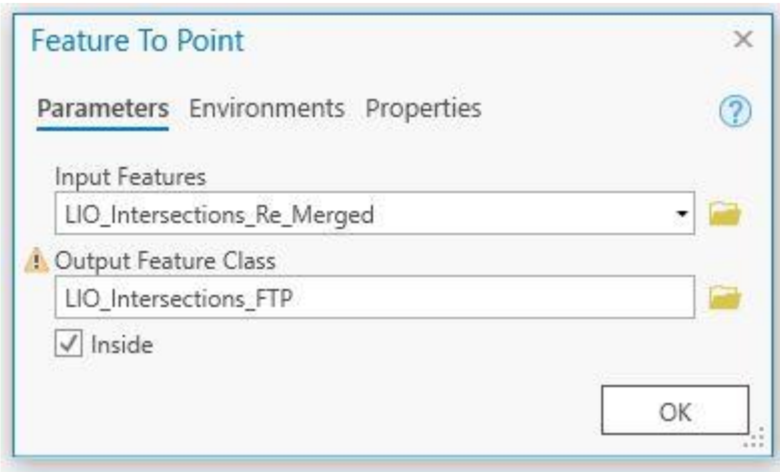


Figure 24: Feature to Point tool parameters.

7.9. Step 9: Spatial Join for City attribute (Model Builder Step 9.0)

The next step involves creating a spatial join between “LIO_Intersect_FTP” and the base data (“LIO_Roads_Project) to add and populate a field with City names for each intersection. Ensure that the field “L_STANDARD” is joined as this field contains City names. See Figure 25 below for “Spatial Join” tool parameters.

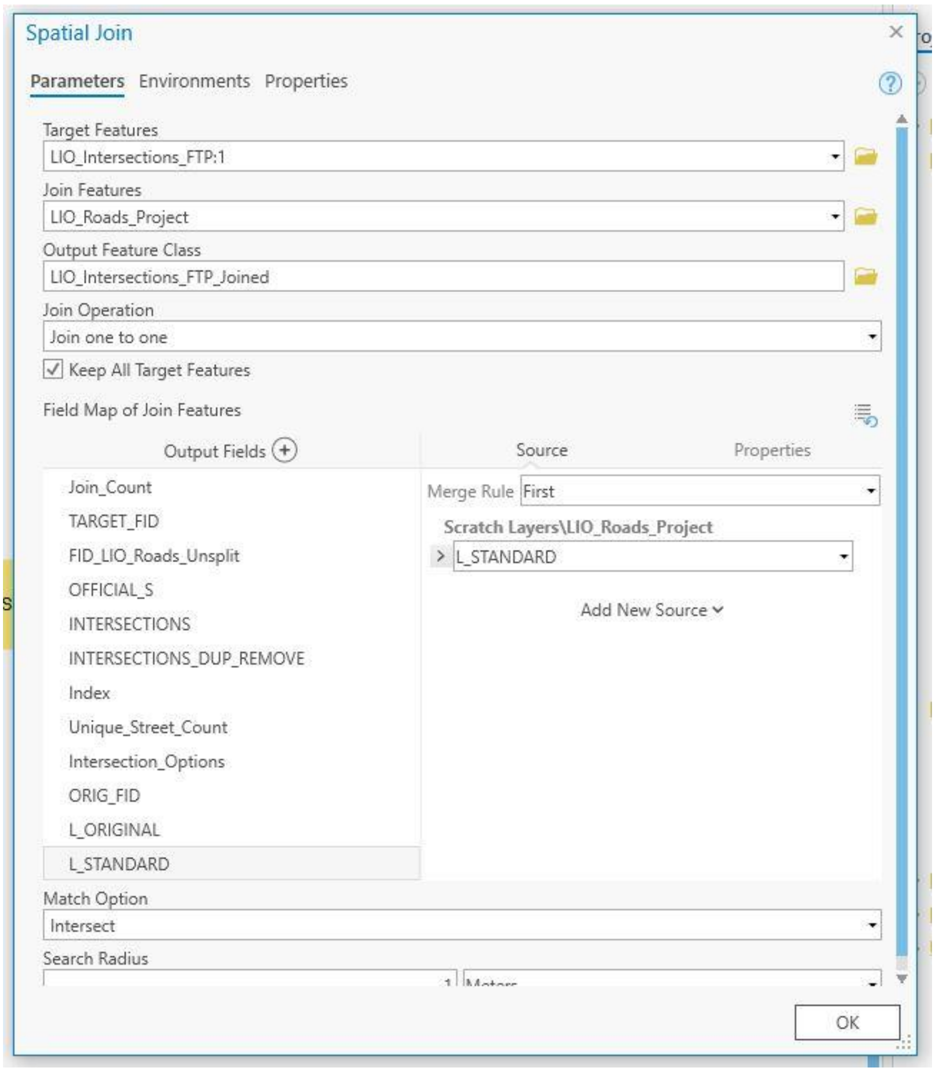


Figure 25: Tool parameters when joining LIO_Intersect_FTP to the base data to populate a city attribute.

7.10. Step 10: Clean up City Names (Model Builder Step 10.0)

The step involves removing City prefixes (such as “Town of”, “City of” etc.) to make the Geocoder more responsive to simplified entries. A text field “City” is added and is calculated using the “Place_Name_String_Adjust.cal” script. The new field has simplified City names for use in the locator.

Also included in this step is adding and populating a “Province” field with “Ontario”.

7.11. Step 11: Subset the data into Ontario and Toronto Sets (Model Builder Step 11.0)

This step subsets the data to Ontario and Toronto data sets. To create the Ontario set the "Select" tool is used with the following SQL express: "City" is **NOT** equal to "Toronto". The resulting feature class is titled "LIO_Ontario_Intersections"

To create the Toronto set the following SQL express: "City" is equal to "Toronto". The resulting feature class is titled "LIO_Toronto_Intersections

These feature classes become the basis for building the locators.

7.12. Step 12: Create Ontario and Toronto Intersection Locators (Model Builder Step 12.0)

The final step involves creating the locators with the "Create Locator" tool.

For each locator, the following parameters are set:

Role: POI

Place Name: "Intersection_Options"

City/Municipality: "City"

Province: "Province"

Save the locators where desired.

8.0. Creating the Canada Place Names Locator

8.1. Step 1: Project Place Name data (Model Builder Step 1.0)

The base data as downloaded from Geofabrik/OSM contains a number of feature class describing point, polygon and line features for each province. For this locator we need to find the "gis_osm_places_free_1" point feature data. This data contains place names for hundreds of City and Towns in each province.

Model 1.0 takes this Place Name point data and re-projects it into a File Geodatabase using the "Project" tool. The resulting feature classes are projected into the GCS_North_American_1983_CSRS coordinate system. There are 13 instances of the "Project" tool to re-project all Provinces and Territories.

8.2. Step 2: Create and Populate Province field (Model Builder Step 2.0)

Step 2 takes the resultant feature classes from Step 1 and adds and populates the Province field as this attribute is missing from the base data. The “Calculate Field” tool is to create a new field and the script “Populate_Province_Placename.cal” is used to populate the field. In each iteration the script is modified to populate the correct Province name based on the data set being altered.

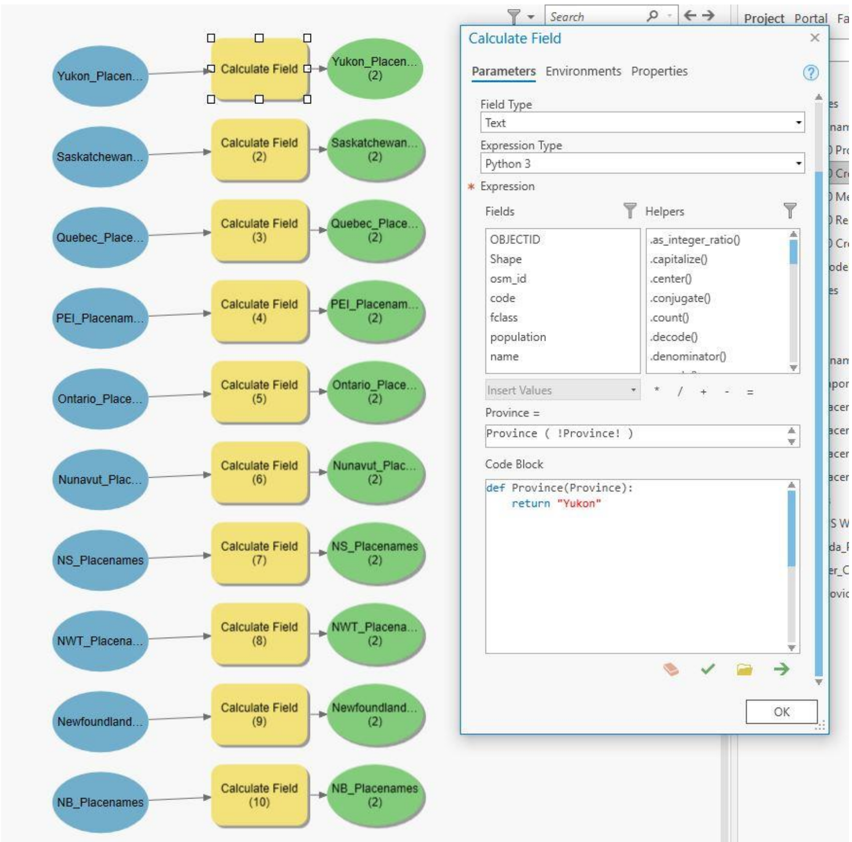


Figure 26: Creating and populating the Province field.

8.3. Step 3: Merge Canada-Wide datasets (Model Builder Step 3.0)

Step 3 uses the “Merge” tool to combine all 13 datasets into one master data set. In Model 3.0 we named the master data set “Canada_Placenames”.

8.4. Step 4: Remove blank entries (Model Builder Step 4.0)

For the POI locator to function correctly all fields in the “name” attribute must either be populated or “<Null>”. In this data set there a number of entries that have blank space. This will cause an error when creating the locator and as such they must be removed.

To do this a selecting query is performed on the “Canada_Placenames” data using the “Select” tool. The query selects “WHERE “name” IS NOT EQUAL TO “*blank*”. A blank entry is the first option when selecting the second part of the query and must be selected. This remove all records with blank entries for the “name” attribute in the resultant feature class.

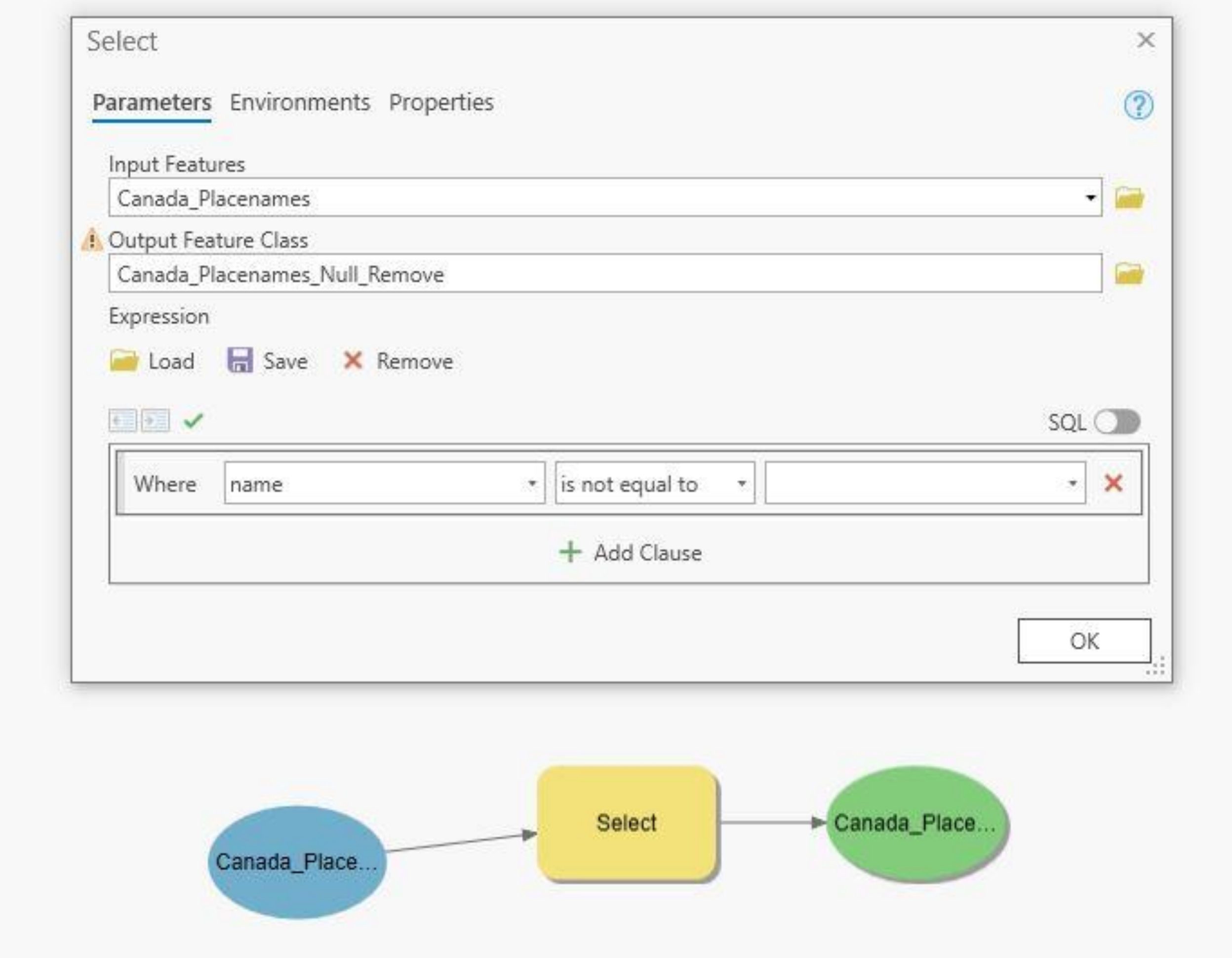


Figure 27: Removing blank entries from Canada_Placenames.

8.5. Step 5: Create Place Names Locator (Model Builder Step 5.0)

The last step is to create the locator. This involves selecting a “POI” locator and selecting the “Canada_Placenames_Null_Remove” data set as the Primary Table. The “Place Name” field is mapped to “name” and the “Province” field is mapped to “Province”. The output locator should be stored in the same folder that the Master_Composite will be stored in.