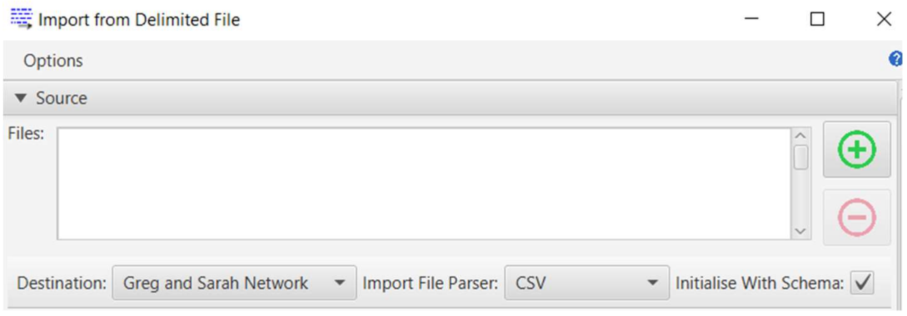
Learning Module - Importing CSVs

There are two primary methods for getting data into Constellation to analyse. The first is to use the Data Access View, which requires setting up direct feeds to pre-modelled data mapped to Constellation's schema. The second is importing from CSVs.

Importing from CSVs lets us define which elements of our data will be nodes, attributes, and relationships on an ad-hoc basis. We can easily obtain new data, manipulate it, and import it into Constellation to federate it with existing data. This module will teach you how to do this.

Using the CSV importer

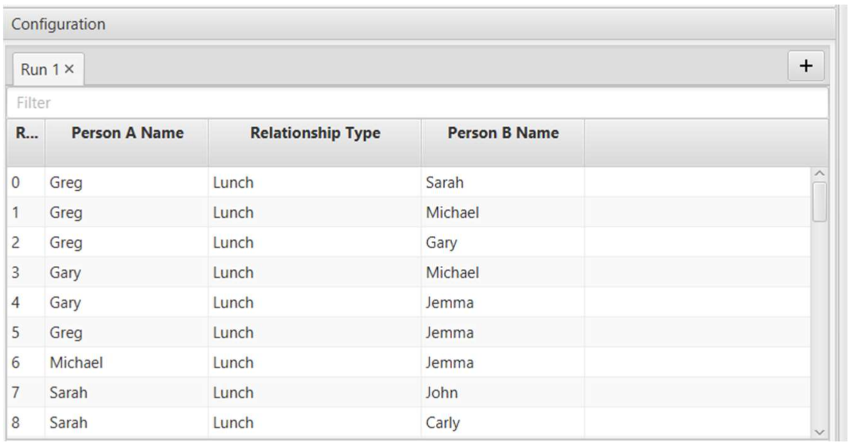
The CSV Importer can be accessed in Constellation through File > Import > From Delimited File. The Importer will open, and includes the following sections:



Destination: Greg and Sarah Network
Import File Parser: CSV
Initialise With Schema: ✓

The first section lets you choose the location of the CSV you are importing with the ⊕ button. You also need to choose a 'Destination', which will either be an existing open graph (if you want to import your data on top of your existing graph), or a new graph. There are other options for different file type imports, but we'll leave that for now.

Once you have chosen a CSV, the second section will populate with data from the CSV. Whatever column headings you have in your CSV data will be visible, along with a sample of what is in the data. I've imported the Greg and Sarah Lunch Network data as a CSV to visualise this.

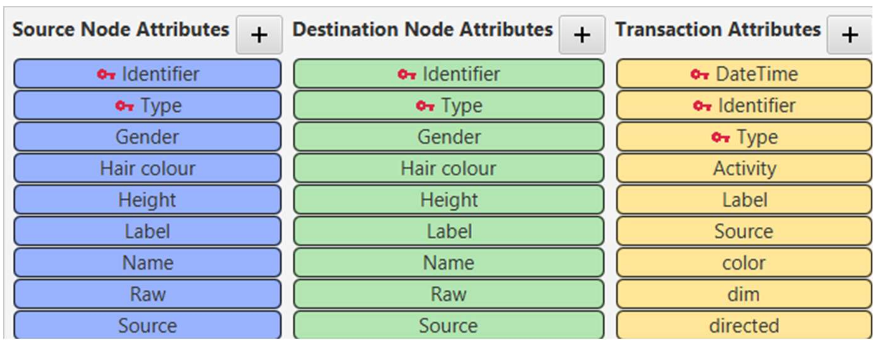

Configuration

Run 1 ×

Filter

R...	Person A Name	Relationship Type	Person B Name
0	Greg	Lunch	Sarah
1	Greg	Lunch	Michael
2	Greg	Lunch	Gary
3	Gary	Lunch	Michael
4	Gary	Lunch	Jemma
5	Greg	Lunch	Jemma
6	Michael	Lunch	Jemma
7	Sarah	Lunch	John
8	Sarah	Lunch	Carly

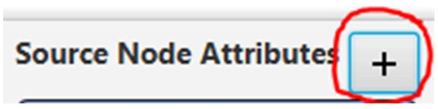
We can see that our data consists of rows showing 'lunch' between Person A, and Person B. At the moment, Constellation doesn't know anything about this data: it doesn't know which columns represent Nodes, Relationships, or Attributes. The third section of the Importer allows us to instruct Constellation *how* to graph the data.



Source Node Attributes	Destination Node Attributes	Transaction Attributes
Identifier	Identifier	DateTime
Type	Type	Identifier
Gender	Gender	Type
Hair colour	Hair colour	Activity
Height	Height	Label
Label	Label	Source
Name	Name	color
Raw	Raw	dim
Source	Source	directed

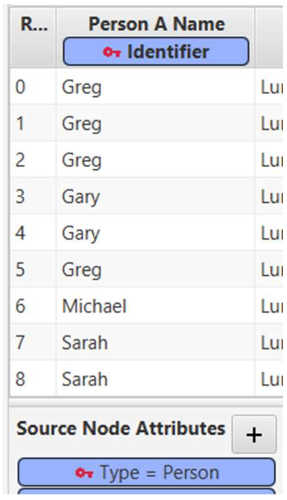
We'll focus on the first column, Source Node Attributes, for now. We have a number of Node Attributes available to set for the initiator of the relationship, including Identifier, Type, Gender, Hair colour, and etc. There are also Key Attributes, indicated by the symbol. Key Attributes are how Constellation decides which Nodes and Relationships should be merged on the graph: if the Key Attributes for any two Nodes/Relationships are the same, they will be merged together, because Constellation will assume they are the same entity/element.

We have a few options at this point, depending on what our data looks like:

- If the columns already align with the attributes we need to asign, we can drag the attributes onto the relevant columns
- If we need a new attribute, we can click the  and create one, then drag that onto a column

- If we want to set a default attribute for all Source Nodes, we can right-click on that attribute on the attribute and Set Default Value. You might do this if all of your Source Nodes were type:person, for example.

For our data, we'll set the Source Node Identifier Key Attribute to the column of 'Person A Name'. This will make Constellation merge all Nodes with the same name together; otherwise, you would get a new Node for every 'Greg' in the data. We'll also set the default value of 'Type' to 'Person'.

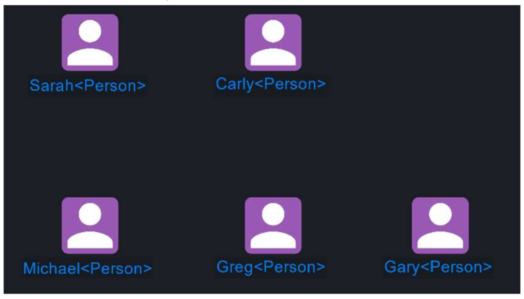


R...	Person A Name Identifier	
0	Greg	Lu
1	Greg	Lu
2	Greg	Lu
3	Gary	Lu
4	Gary	Lu
5	Greg	Lu
6	Michael	Lu
7	Sarah	Lu
8	Sarah	Lu

Source Node Attributes +

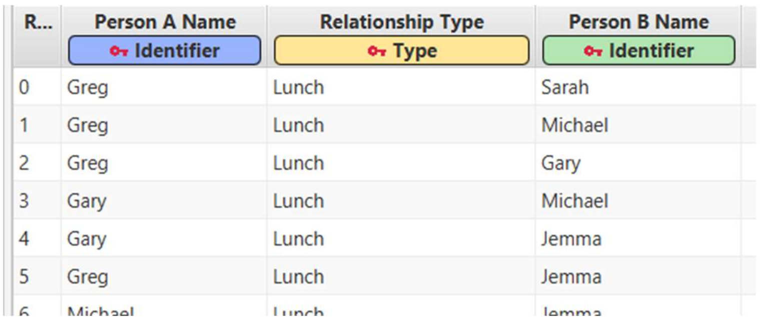
Type = Person

If we imported the data at this point, we would get a graph with only Nodes, where the Identifiers would be the values from the 'Person A Name' column, and they would all be of Type 'Person'. This isn't very helpful.



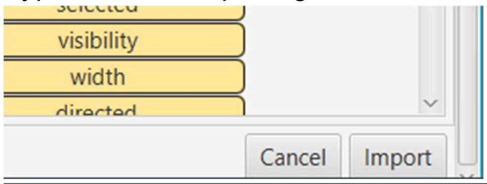
We can repeat this process for the Destination Node attributes, but this will still only give us Nodes on the graph - Constellation doesn't have any information about what the Relationship between those Nodes are. We need to fill in the 'Transaction Attributes' to provide this instruction.

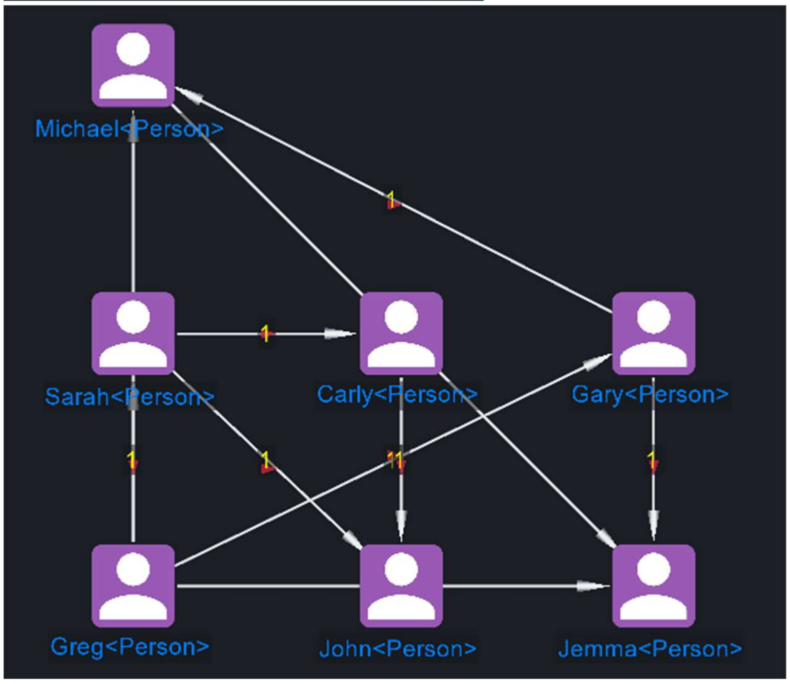
In our data, we only have information that the Type of Relationship between the Source and Destination Nodes is 'Lunch', so we'll drag the Type Transaction Attribute onto the 'Relationship Type' Column. This should give us the following:



R...	Person A Name Identifier	Relationship Type Type	Person B Name Identifier
0	Greg	Lunch	Sarah
1	Greg	Lunch	Michael
2	Greg	Lunch	Gary
3	Gary	Lunch	Michael
4	Gary	Lunch	Jemma
5	Greg	Lunch	Jemma
6	Michael	Lunch	Jemma

If we import the CSV at this point, Constellation now knows that the Identifier for the Source Nodes is the ‘Person A Name’ column, the Identifier for the Destination Nodes is the ‘Person B Name’ column, and the Type of the Relationships between the Nodes is in the ‘Relationship Type’ column. Importing this information will give us a graph.



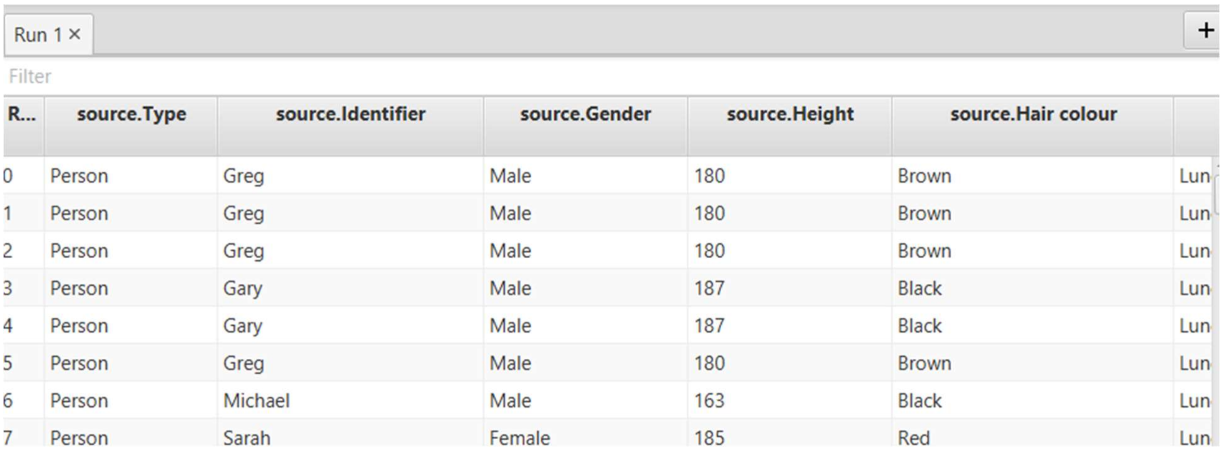


Using the CSV Importer, with more Attributes

Of course, we have more information available about both our Source and Destination Nodes, including their gender, height, and hair colour. Lets try importing a more complex CSV that includes this information.

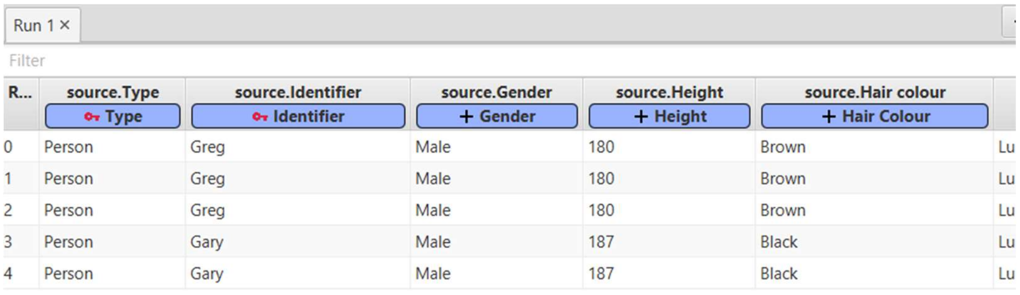
In the CSV Importer, open ‘Import - Greg and Sarah Network expanded.csv’

You’ll see there are now many more columns in the data. The column headings have the ‘source.’, ‘transaction.’ and ‘destination.’ prefixes on them to make it easier to ascertain which attributes map to which Nodes/Relationships.



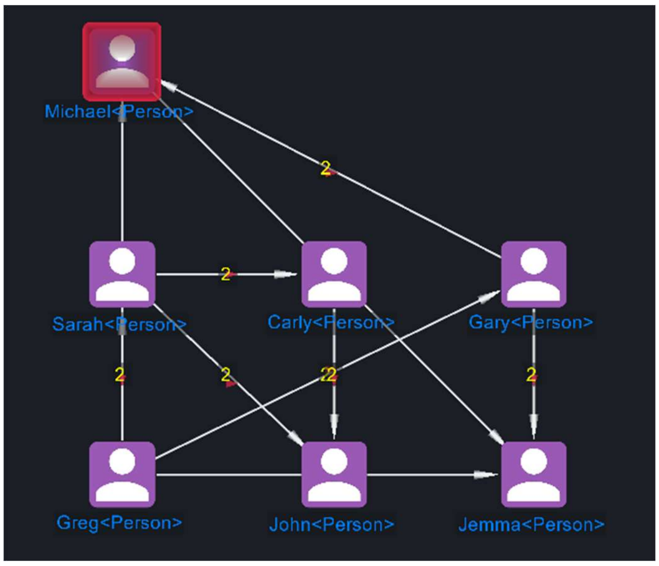
R...	source.Type	source.Identifier	source.Gender	source.Height	source.Hair colour	
0	Person	Greg	Male	180	Brown	Lun
1	Person	Greg	Male	180	Brown	Lun
2	Person	Greg	Male	180	Brown	Lun
3	Person	Gary	Male	187	Black	Lun
4	Person	Gary	Male	187	Black	Lun
5	Person	Greg	Male	180	Brown	Lun
6	Person	Michael	Male	163	Black	Lun
7	Person	Sarah	Female	185	Red	Lun

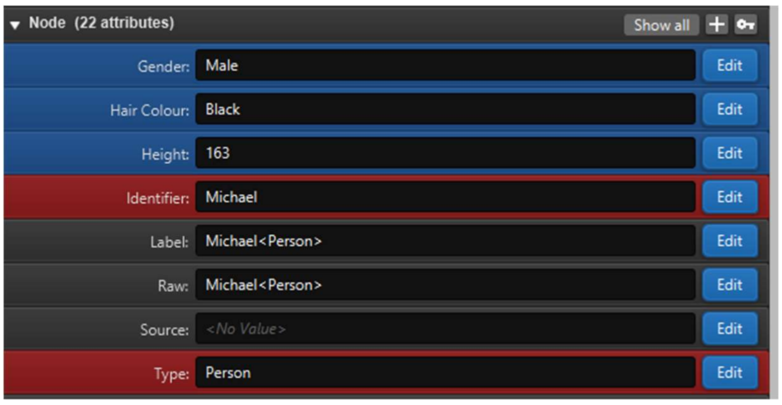
We can now start dragging our attributes from the coloured Source/Destination/Transaction attributes on to the columns. If the attribute doesn’t exist (eg. for Gender), you’ll need to use the + button to create those attributes. I’ll add in the relevant attributes for the Source Nodes.



R...	source.Type	source.Identifier	source.Gender	source.Height	source.Hair colour	
	Type	Identifier	+ Gender	+ Height	+ Hair Colour	
0	Person	Greg	Male	180	Brown	Lu
1	Person	Greg	Male	180	Brown	Lu
2	Person	Greg	Male	180	Brown	Lu
3	Person	Gary	Male	187	Black	Lu
4	Person	Gary	Male	187	Black	Lu

You’ll then need to do the same for the Destination Node Attributes, and for the Transaction Attributes. When you have done this, importing should give you a graph with all of the relevant data. If you get an error, it may be due to missing values. It’s easiest to deal with this by cleaning/deleting the data from the CSV before importing.




Node (22 attributes) Show all

Gender:	Male
Hair Colour:	Black
Height:	163
Identifier:	Michael
Label:	Michael<Person>
Raw:	Michael<Person>
Source:	<No Value>
Type:	Person

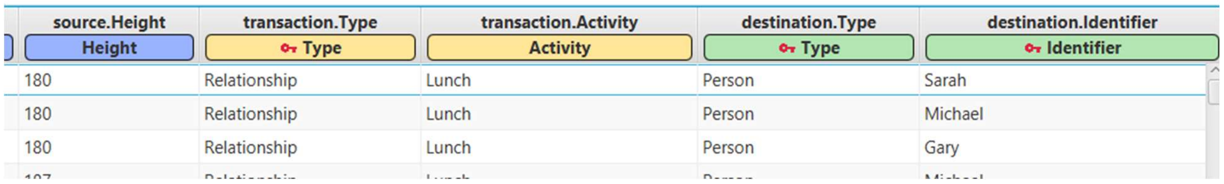
As you can see, we have now imported our data as a graph, with all of the attributes we assigned in the Importing phase assigned to the relevant Nodes and Relationships. We can now do analysis on our data.

Multiple Nodes, multiple Runs

Sometimes we have data in a table where there are multiple Relationships that we want to graph. An example of this is our Greg and Sarah Lunch Network with Locations, where there is not only a relationship between the people having lunch with each other, but also a relationship between the individuals and the locations that they are eating. In each Import, we only get to specify Attributes for one set of Source and Destination Nodes, so we need to use multiple 'runs' of the Import over the same set of data to map all of the Attributes that we require.

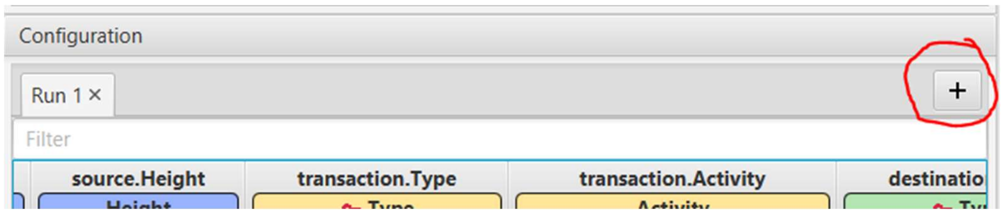
In Constellation, open the file ‘Import - Greg and Sarah Network with Locations.csv’. This is a simplified version of the network, showing only those who are meeting together at The Blue Duck.

Again, the column headings have been left with prefixes to make matching attributes to columns an easier process. You’ll see ‘source.’, ‘destination.’, ‘transaction.’, and ‘LocationDestination.’ prefixes to help you. We can start by Mapping the Source, Destination, and Transaction attributes for the relationships between the people, as we did previously.

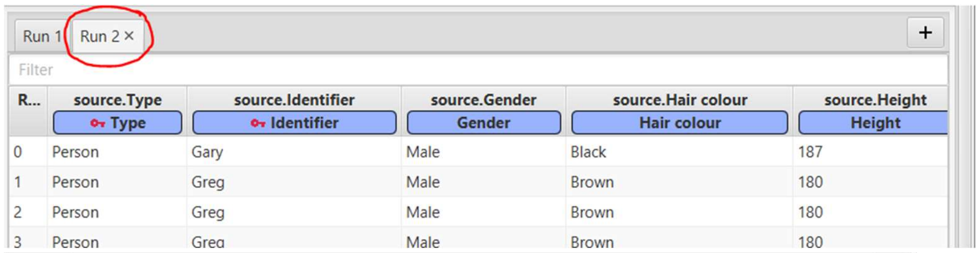


source.Height	transaction.Type	transaction.Activity	destination.Type	destination.Identifier
Height	Type	Activity	Type	Identifier
180	Relationship	Lunch	Person	Sarah
180	Relationship	Lunch	Person	Michael
180	Relationship	Lunch	Person	Gary

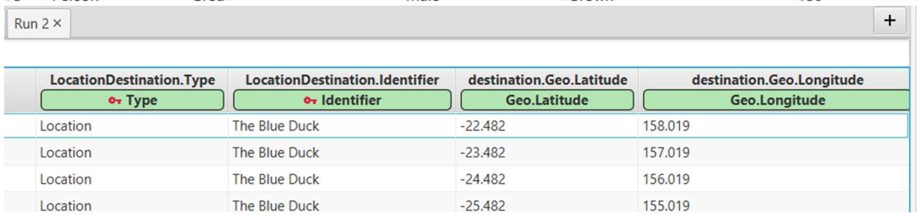
This will give us a graph with all of the connections between the people, but not between the people and the locations. There are separate columns on the far-right of the data called LocationDestination.Type, LocationDestination.Identifier, and Geo.Latitude/Geo.Longitude, but because we have been mapping attributes for the people, we haven’t been able to map attributes for the locations. To account for this, we need to do an additional ‘run’ of the importer over the data. To do this, we need to click this button:



In this second ‘run’, we’ll be mapping the attributes of the Source Nodes as the people in the first column (the original Source Nodes), and the Destination Nodes as the Locations. This will create relationships between the first people and the locations they visited for lunch. We will cover Location data in more detail in a later Exercise Module; for now, you will need to create new Destination Node attributes called ‘Geo.Latitude’ and ‘Geo.Longitude’.

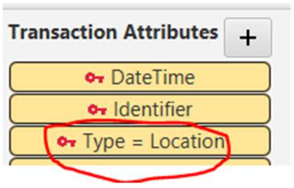


R...	source.Type	source.Identifier	source.Gender	source.Hair colour	source.Height
	Type	Identifier	Gender	Hair colour	Height
0	Person	Gary	Male	Black	187
1	Person	Greg	Male	Brown	180
2	Person	Greg	Male	Brown	180
3	Person	Greg	Male	Brown	180



LocationDestination.Type	LocationDestination.Identifier	destination.Geo.Latitude	destination.Geo.Longitude
Type	Identifier	Geo.Latitude	Geo.Longitude
Location	The Blue Duck	-22.482	158.019
Location	The Blue Duck	-23.482	157.019
Location	The Blue Duck	-24.482	156.019
Location	The Blue Duck	-25.482	155.019

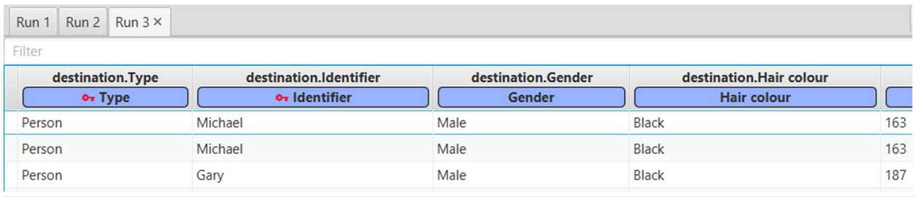
You'll notice we don't have a Type for the transaction/relationship, as there is no column for it in the data. To account for this, we need to set the default value for transaction.Type to 'Location' (right click on Transaction Type, click Set Default Value, enter 'Location').



Transaction Attributes

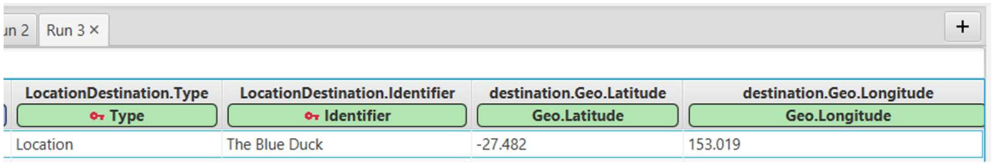
DateTime
Identifier
Type = Location

This run gives us the link between the first column of people and the locations, but we also need to create a link between the second column of people and the locations - it takes two for a lunch meeting, after all. To do this, we need to create a third run, then this time use the second column of people (the original Destination Nodes) as the Source Nodes, and the Locations as the Destination Nodes once again.



Run 1 | Run 2 | Run 3

destination.Type	destination.Identifier	destination.Gender	destination.Hair colour	
Type	Identifier	Gender	Hair colour	
Person	Michael	Male	Black	163
Person	Michael	Male	Black	163
Person	Gary	Male	Black	187

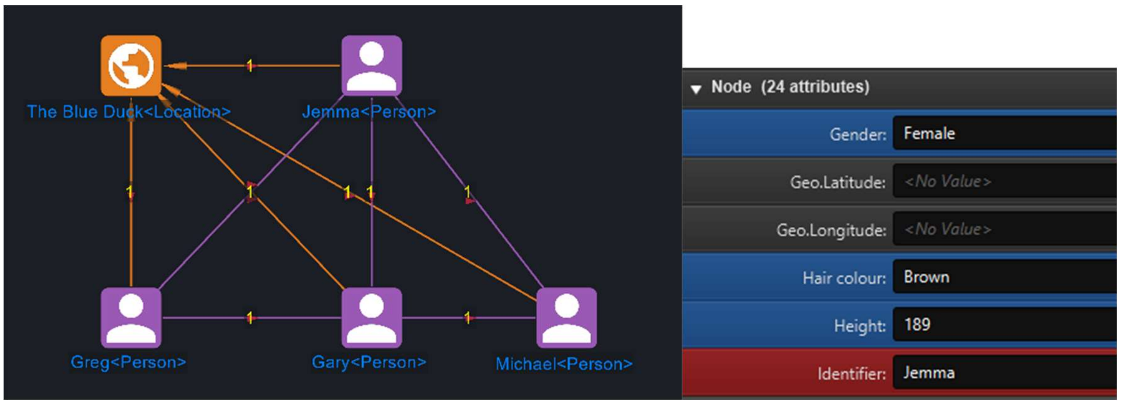


Run 3

LocationDestination.Type	LocationDestination.Identifier	destination.Geo.Latitude	destination.Geo.Longitude
Type	Identifier	Geo.Latitude	Geo.Longitude
Location	The Blue Duck	-27.482	153.019

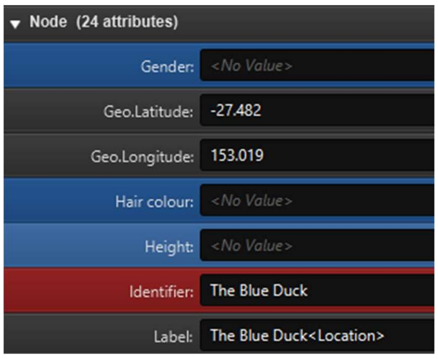
Again, set the Default Value of Transaction Type to 'Location'.

Importing the data should give you the graph with people to people, and people to location links, with all relevant attributes included.



Node (24 attributes)

Gender:	Female
Geo.Latitude:	<No Value>
Geo.Longitude:	<No Value>
Hair colour:	Brown
Height:	189
Identifier:	Jemma



▼ Node (24 attributes)	
Gender:	<No Value>
Geo.Latitude:	-27.482
Geo.Longitude:	153.019
Hair colour:	<No Value>
Height:	<No Value>
Identifier:	The Blue Duck
Label:	The Blue Duck<Location>

Depending on what your data looks like, you may need to do many more ‘runs’ to map all of the Node to Node relationships.