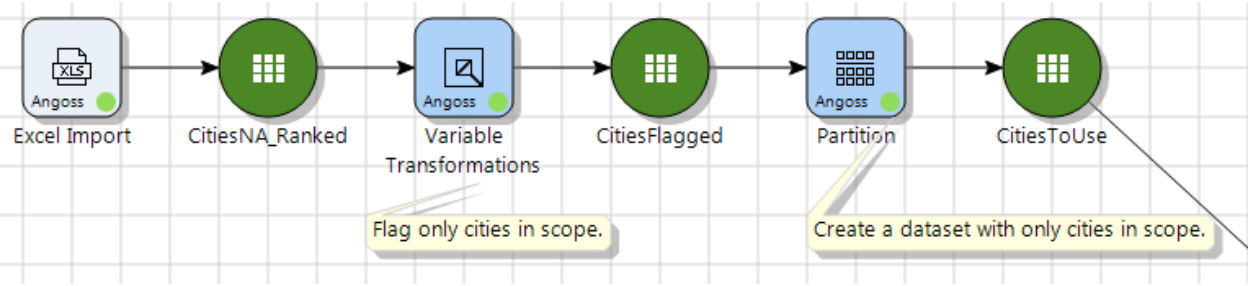
Filtering Join Challenge: Eric's Solution

I'll try to lay out what I did as concisely as possible. I have also attached an Angoss project file with my solution which can be explored.

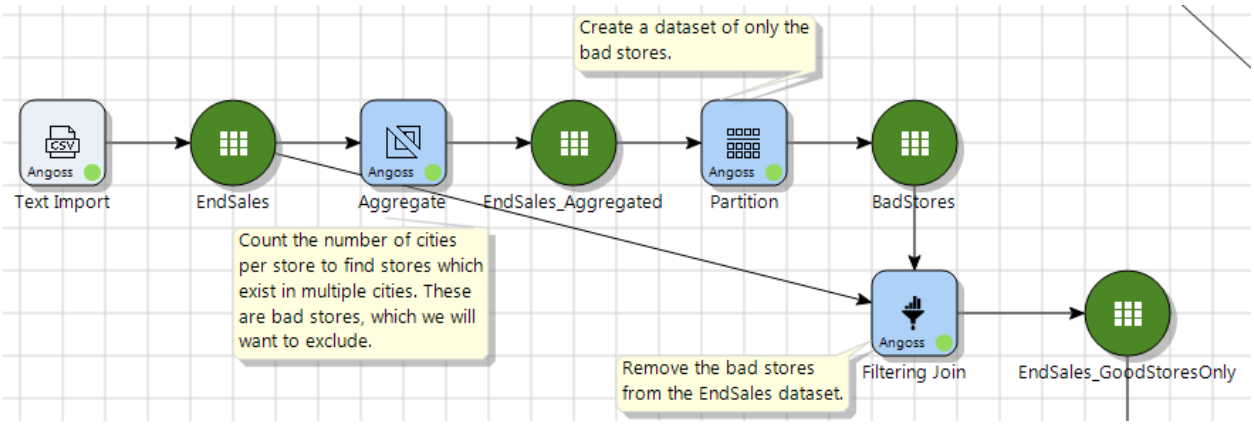
Part 1:

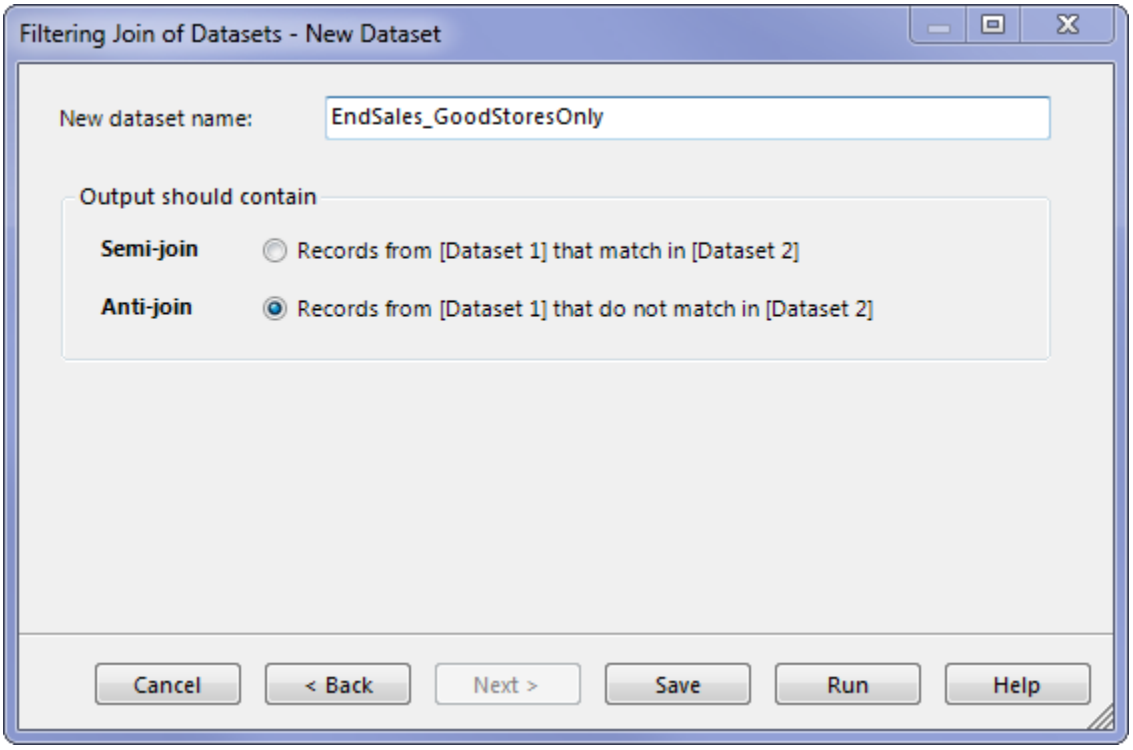
There are 3 main pathways of data prep I do in part 1.

a. First of all, I import the Ranked Cities file, and use a Variable transformations node to flag only cities in scope (those in the US, Mexico, or named Toronto or Vancouver). Then I partition the dataset to only include those cities in Scope. I call that dataset 'CitiesToUse'.

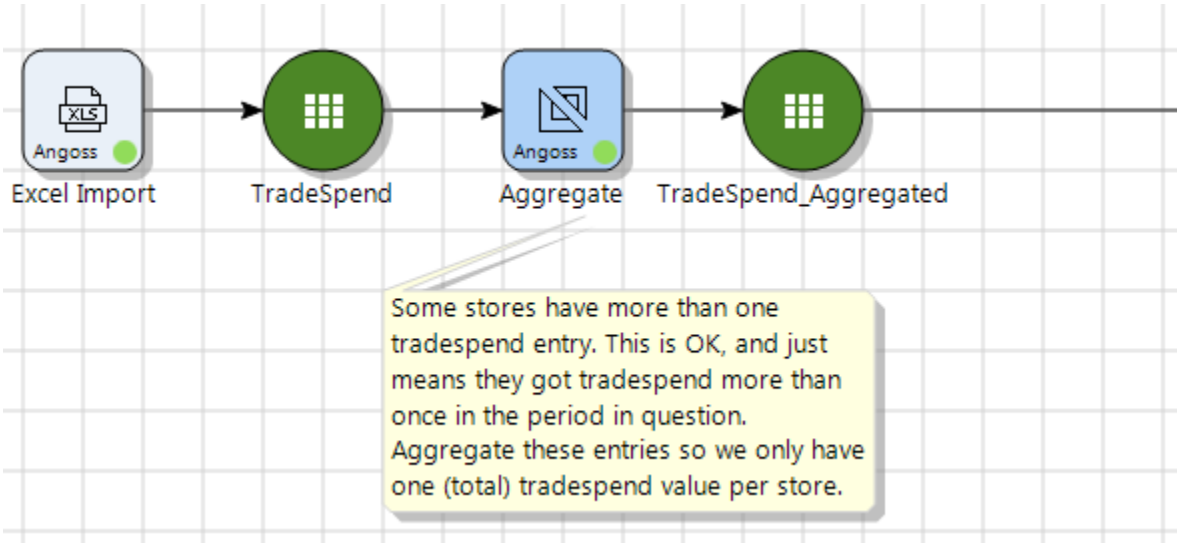


b. In a separate path, I import the EndSales dataset. I use an aggregate node to count the number of cities per store. I then use a Partition node to create a dataset of only the stores which exist in more than one city. Finally, I use a Filtering Join node with the anti-join option to exclude all stores in the "BadStores" dataset from the original EndSales dataset


Create a dataset of only the bad stores.
Angoss
Text Import
EndSales
Angoss
Aggregate
EndSales_Aggregated
Angoss
Partition
BadStores
Count the number of cities per store to find stores which exist in multiple cities. These are bad stores, which we will want to exclude.
Angoss
Filtering Join
EndSales_GoodStoresOnly
Remove the bad stores from the EndSales dataset.

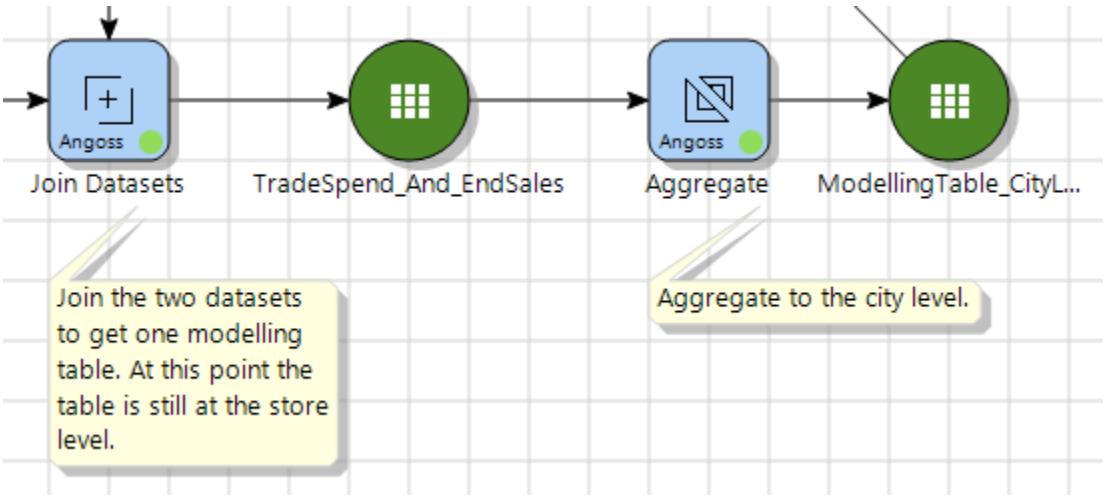

Filtering Join of Datasets - New Dataset
New dataset name:
EndSales_GoodStoresOnly
Output should contain
Semi-join
Records from [Dataset 1] that match in [Dataset 2]
Anti-join
Records from [Dataset 1] that do not match in [Dataset 2]
Cancel
< Back
Next >
Save
Run
Help

c. In another separate pathway, I import the TradeSpend dataset, and aggregate it to get the total sales per store.

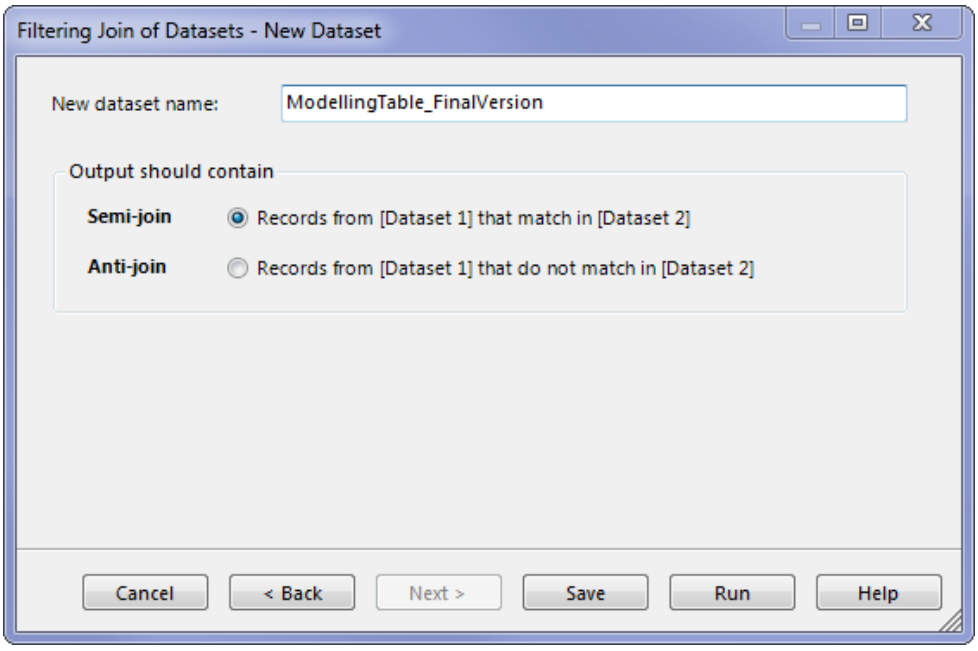


d. I then join the EndSales_GoodStoresOnly dataset to the TradeSpend_Aggregated dataset using a "left-join" to only keep records that appear in the EndSales_GoodStoresOnly dataset.

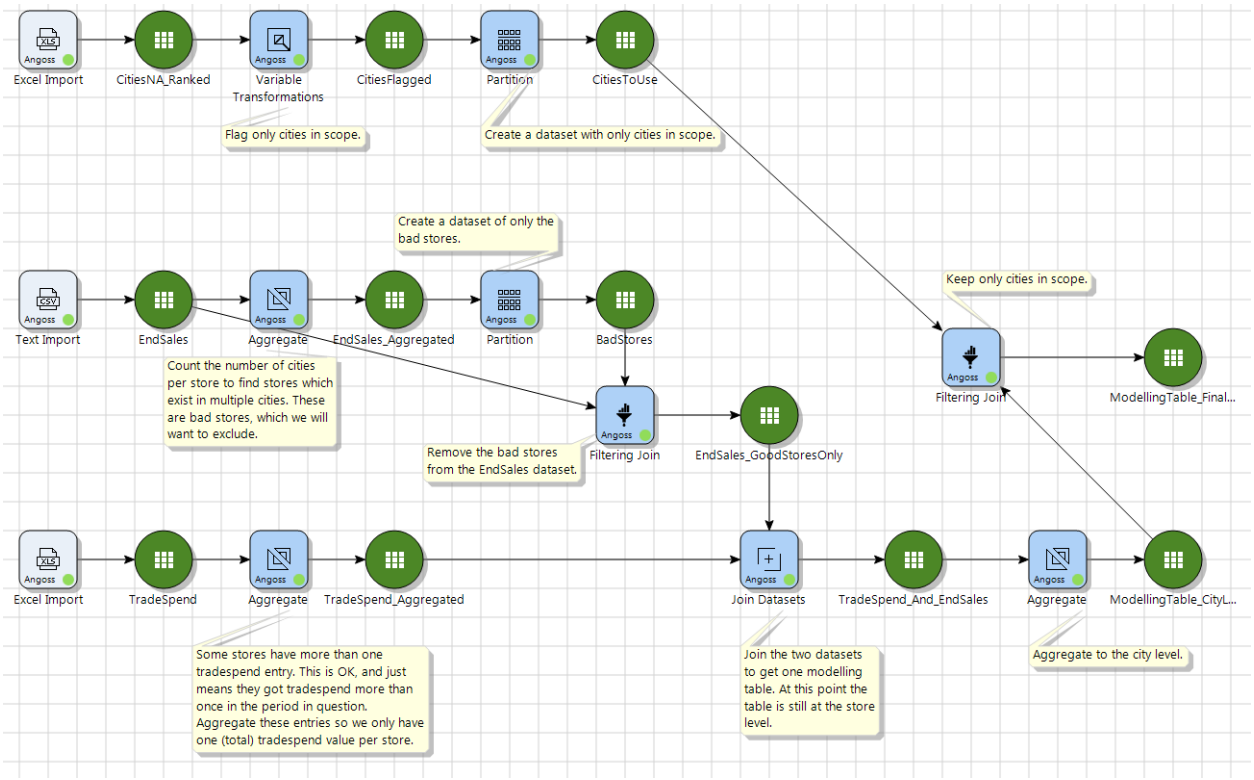
I aggregate this dataset one more time to get to the city level (taking the SUM function for both End Sales and Trade Spend).



e. Finally, I take the result of that aggregation and use a filtering join node with the “semi-join” option to produce the final modelling table.



Here’s the whole workflow:



And here's my final solution:

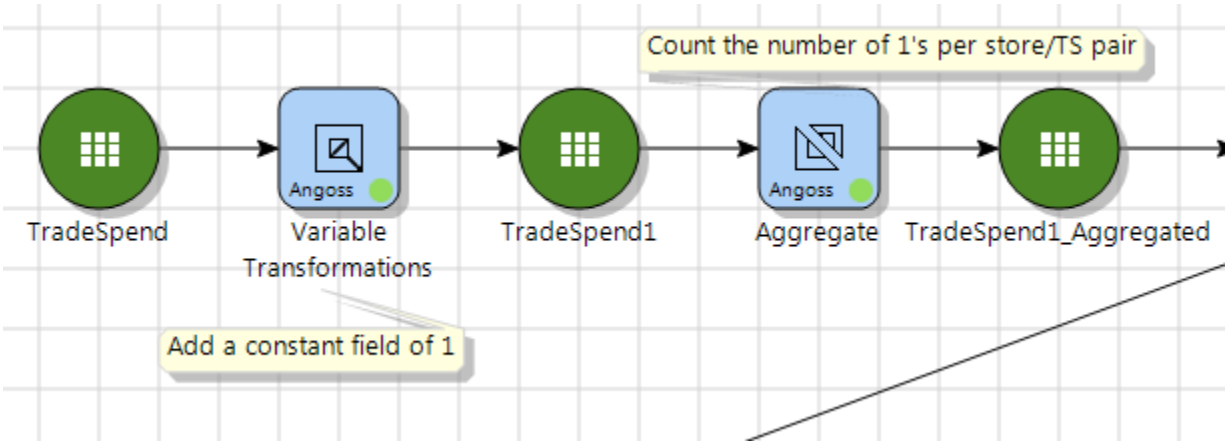
	City	Country	Total End Sales	Total Trade Spend
1	Aguascalientes	Mexico	20888475	2364305
2	Austin	United States	19337461	2257463
3	Charlotte	United States	21335774	2572733
4	Chicago	United States	21807766	2614381
5	Chihuahua	Mexico	21979176	2573078
6	Columbus	United States	21408705	2574431
7	Dallas	United States	20224661	2457337
8	Ecatepec de Morelos	Mexico	19970891	2407437
9	Fort Worth	United States	20381075	2521622
10	Guadalajara	Mexico	19678601	2420573
11	Hermosillo	Mexico	21814183	2647677
12	Houston	United States	19835481	2340546
13	Indianapolis	United States	20838435	2576535
14	Jacksonville	United States	19652549	2347824
15	Juárez	Mexico	20780420	2454352
16	León	Mexico	20011162	2558131
17	Los Angeles	United States	20635474	2532456
18	Mexico City	Mexico	20617629	2551784
19	Monterrey	Mexico	20430313	2539397
20	Montreal	Canada	21365751	2595105
21	Mérida	Mexico	20240662	2546450
22	Naucalpan	Mexico	20609112	2554242
23	New York City	United States	20174109	2440557
24	Nezahualcóyotl	Mexico	20273567	2385219
25	Philadelphia	United States	21034096	2570100
26	Phoenix	United States	20613019	2425817
27	Puebla	Mexico	19544948	2293739
28	Querétaro	Mexico	21382256	2562654
29	Saltillo	Mexico	21174597	2569044
30	San Antonio	United States	20380818	2405384
31	San Diego	United States	20820977	2524689
32	San Francisco	United States	20744055	2489231
33	San Jose	United States	21735623	2572386
34	San Luis Potosí	Mexico	20900721	2488306
35	Tijuana	Mexico	21083442	2613619
36	Toluca	Mexico	21058015	2413299
37	Toronto	Canada	20437679	2457881
38	Veracruz	Mexico	20824602	2534716
39	Zapopan	Mexico	22334102	2646394

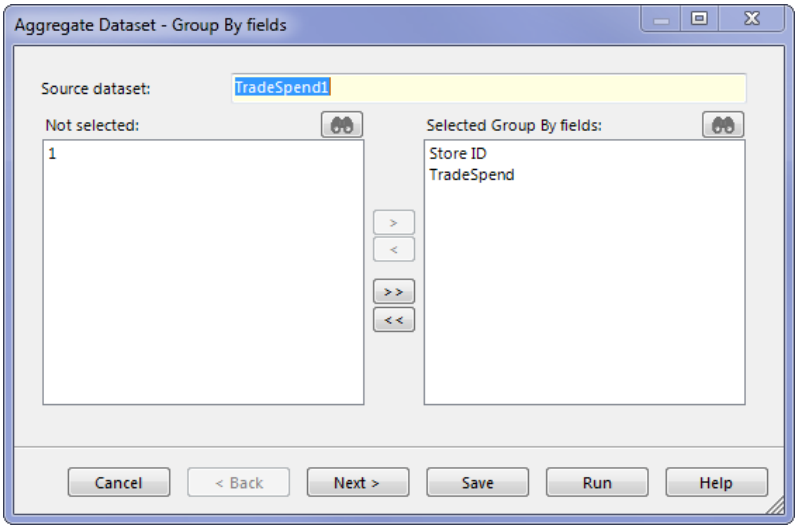
Part 2:

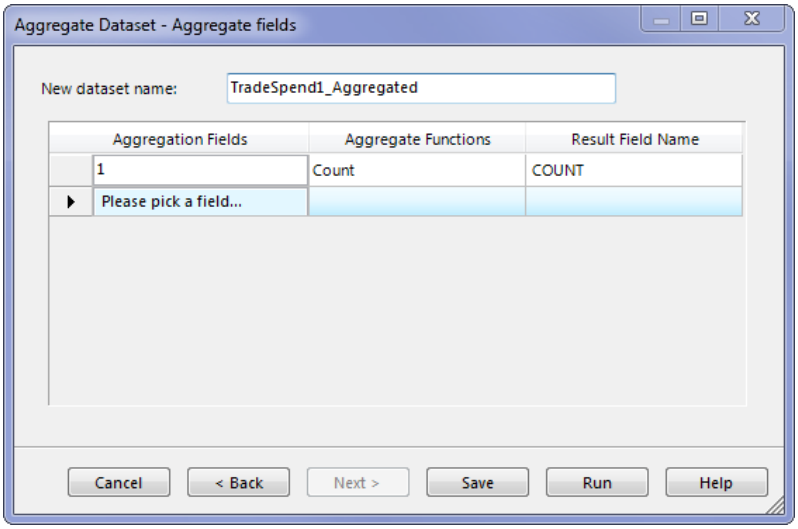
The first step for me was to find the set of identical entries in the TradeSpend file. Since that file only has 2 fields (Store ID and TradeSpend), we're looking for duplicate entries.

Note that if we believed that the duplicate entries were not fraud and just mistakes that we needed to keep one of, we could use the helpful Deduplicate node. But since we believe these are fraudulent entries and want to exclude them all in this case, we need to do something more complicated.

I started by using a Variable Transformations node to add a new field with the value of 1 for every record. Then I aggregated with both Store ID and TradeSpend as my group by fields. I count the number of 1's in the Aggregate field (and called it COUNT).

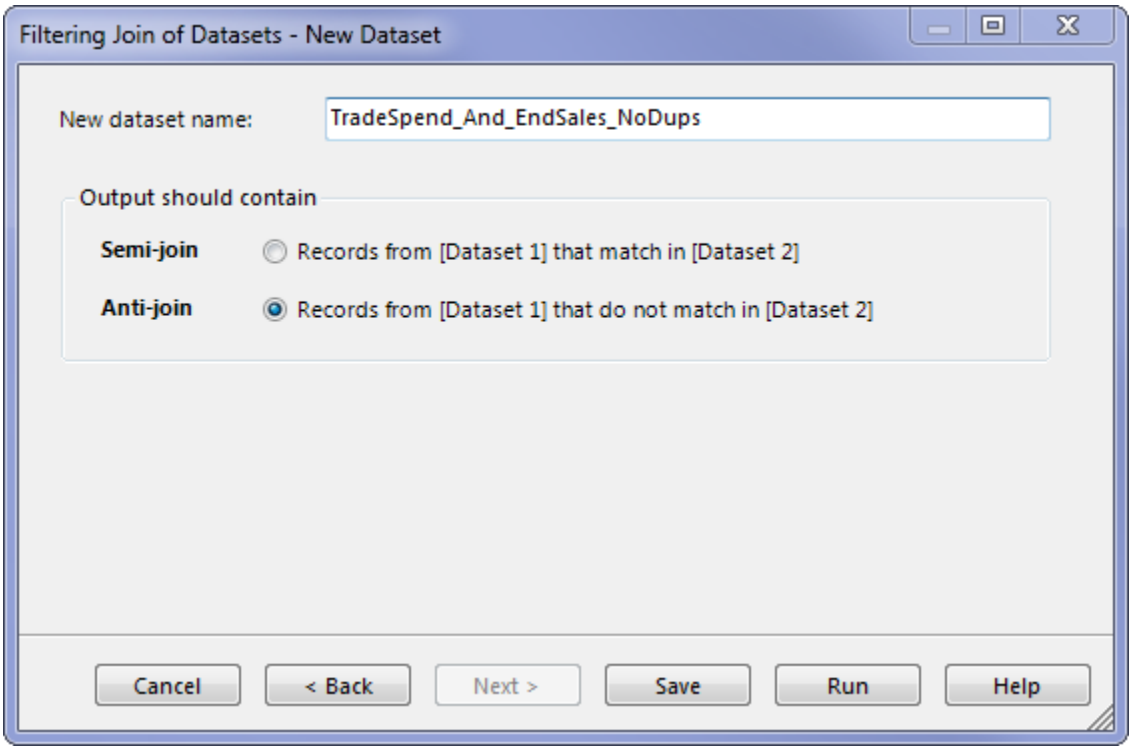




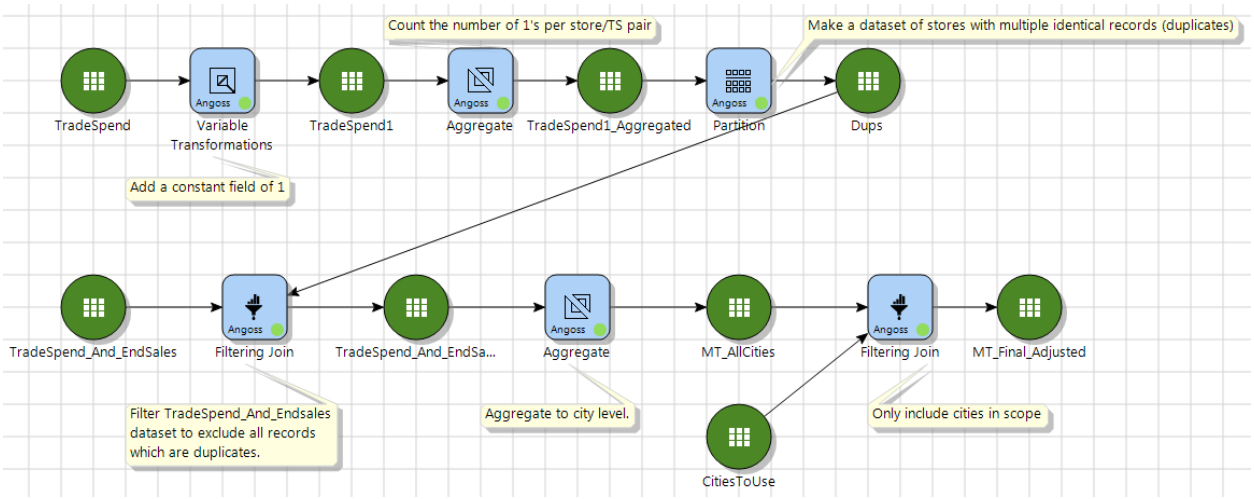


I then partition this dataset to only keep the records with a COUNT value greater than 1. (There are only 3 such records). I called that dataset DUPS.

At this point, I am in pretty good shape. I already have the dataset “TradeSpend_And_EndSales” from my first pass at the problem. I use a filtering join with the Anti-Join feature to remove the 3 stores which are bad from that dataset, and then follow the same path as before (aggregate to city level, and filtering join to only include cities in scope).



The whole workflow looks like this:



And the adjusted solution is below. (If you're wondering why only one number changed, it's because the other two stores with fraudulent entries were in cities out of scope for the project.

	City	Country	Total End Sales	Total Trade Spend
1	Aguascalientes	Mexico	20888475	2364305
2	Austin	United States	19337461	2257463
3	Charlotte	United States	21335774	2572733
4	Chicago	United States	21807766	2614381
5	Chihuahua	Mexico	21979176	2573078
6	Columbus	United States	21408705	2574431
7	Dallas	United States	20224661	2457337
8	Ecatepec de Morelos	Mexico	19970891	2407437
9	Fort Worth	United States	20381075	2521622
10	Guadalajara	Mexico	19678601	2420573
11	Hermosillo	Mexico	21814183	2647677
12	Houston	United States	19835481	2340546
13	Indianapolis	United States	20838435	2576535
14	Jacksonville	United States	19652549	2347824
15	Juárez	Mexico	20758301	2448882
16	León	Mexico	20011162	2558131
17	Los Angeles	United States	20635474	2532456
18	Mexico City	Mexico	20617629	2551784
19	Monterrey	Mexico	20430313	2539397
20	Montreal	Canada	21365751	2595105
21	Mérida	Mexico	20240662	2546450
22	Naucalpan	Mexico	20609112	2554242
23	New York City	United States	20174109	2440557
24	Nezahualcóyotl	Mexico	20273567	2385219
25	Philadelphia	United States	21034096	2570100
26	Phoenix	United States	20613019	2425817
27	Puebla	Mexico	19544948	2293739
28	Querétaro	Mexico	21382256	2562654
29	Saltillo	Mexico	21174597	2569044
30	San Antonio	United States	20380818	2405384
31	San Diego	United States	20820977	2524689
32	San Francisco	United States	20744055	2489231
33	San Jose	United States	21735623	2572386
34	San Luis Potosí	Mexico	20900721	2488306
35	Tijuana	Mexico	21083442	2613619
36	Toluca	Mexico	21058015	2413299
37	Toronto	Canada	20437679	2457881
38	Veracruz	Mexico	20824602	2534716
39	Zapopan	Mexico	22334102	2646394