Analysis Workspace Framework

Version: 1.1

Inhoud

[1 System requirements 4](#_Toc421988388)

[2 Installation 4](#_Toc421988389)

[3 Concept 4](#_Toc421988390)

[4 GUI 6](#_Toc421988391)

[5 Open, Close and Save 7](#_Toc421988392)

[6 Workspaces 7](#_Toc421988393)

[7 Base collections 7](#_Toc421988394)

[7.1  Add Smallworld collection 7](#_Toc421988395)

[7.2  Add trail 8](#_Toc421988396)

[7.3 Add selection 8](#_Toc421988397)



[7.4 Add selection collections 8](#_Toc421988398)



[7.5 Add from explorer 9](#_Toc421988399)



[7.6 Add from explorer selection 9](#_Toc421988400)



[7.7 Add from scrapbook 9](#_Toc421988401)



[8 Filter collections 9](#_Toc421988402)

[8.1 Filter by geometry 9](#_Toc421988403)



[8.2 Filter by distance 10](#_Toc421988404)

[8.3 Filter by predicate 11](#_Toc421988405)



[9 Follow collections 11](#_Toc421988406)

[9.1  Topology collection 12](#_Toc421988407)

[9.2  Join collection 12](#_Toc421988408)

[9.3  Method collection 13](#_Toc421988409)

[9.4  Root collection 13](#_Toc421988410)

[10 Geometry operators 15](#_Toc421988411)

[10.1 Geometry collection 15](#_Toc421988412)



[10.2 Buffer collection 16](#_Toc421988413)



[11 Set operators 17](#_Toc421988414)

[11.1 Area collections 17](#_Toc421988415)

[11.2 Record collections 18](#_Toc421988416)

[11.3 Spatial collections 18](#_Toc421988417)

[12 Views 18](#_Toc421988418)

[12.1  View on join 18](#_Toc421988419)

[12.2 View based on common attribute 20](#_Toc421988420)



[13 Materialize 21](#_Toc421988421)

[13.1  Materialize record collection 21](#_Toc421988422)

[13.2 Materialize geometry collection 22](#_Toc421988423)



[14 Utilities 23](#_Toc421988424)

[14.1 rename 23](#_Toc421988425)

[15 Execution 23](#_Toc421988426)

[15.1 Explorer 24](#_Toc421988427)

[15.2 Goto 24](#_Toc421988428)

[15.3 Map 24](#_Toc421988429)

[15.4 Excel/Access 24](#_Toc421988430)

[16 Examples 25](#_Toc421988431)

[17 Unit Test 27](#_Toc421988432)

# System requirements

The software is build upon Smallworld 4.3 core. No TSBs are required.

# Installation

The Analysis Framework is delivered as a smallworld product, consisting of 3 modules:

* rw\_analysis\_workspace: the core analysis collections.
* rw\_analysis\_workspace\_framework: the framework, plugins and dialogs to perform analysis.
* rw\_analysis\_workspace\_test: test module to run on munit in a Cambridge environment.

1. First load the rw\_analysis\_workspace\_framework module using either sw\_module\_dialog or the sw\_module\_manager.
2. Next add the snippet:

<plugin name="analysis\_plugin" class\_name="rwan:framework\_plugin"/>

To the config.xml of your application.

1. Next add the snippet:

<action name="analysis\_plugin.activate\_framework"/>

To the gui.xml of your application.

# Concept

The main concept in the AF is usage of analysis collections.

There are 3 basic analysis collections:

1. Record collections. Collections that contains, well, records.
2. Geometry collection. Collections that contains geometries. These can be actual geometries like the trail or some selection in the map but it can also be a representation of a certain geometry in a record collection.
3. Area collection. Collections that contains an area. The area can be complex and contain many polygons and many holes. Typically they are created from a trail of a buffer operation.

Analysis collections build upon other base collections, either they filter one base collection, combine 2 base collections, travel the base collection to another collection, etc.  
For example you can take a collection of hotels. Build on that a predicated filter collection that filters for a certain type. Build upon that a topology collection that get the connected roads. This will yield the roads that are topological connected to hotels of a certain type.

Note that you have to think in reverse to solve your problem. Say for example you want to know the Restaurants of French cuisine that are near a park.

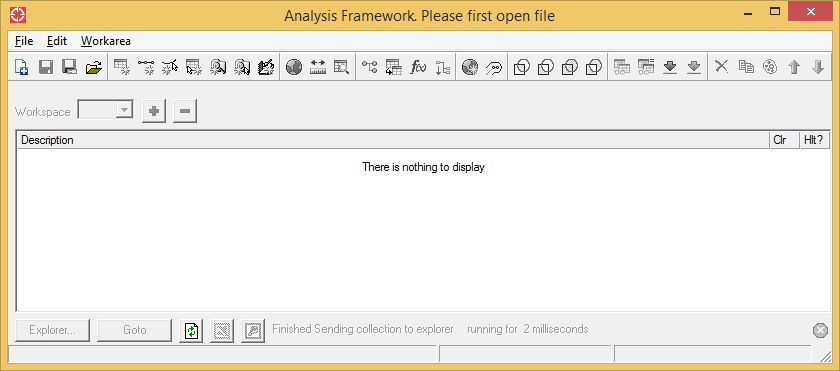
You would go about it like this:

1. Add the park collection
2. Build buffer collection on it with a certain distance (50m)
3. Add the restaurant collection
4. Build a predicate filter on it for cuisine = French
5. Spatially combine the both based on the location of the restaurant to be in the buffer.

The result is one Restaurant, “Midsummer House”.

# GUI

The main GUI of the AF is like this:



**Tools**

**Execution**

**Workspace**

It consists of 3 areas:

1. The tools area contains toolbar buttons and menu buttons
2. The workspace area shows the analysis collections in the chosen workspace. You can add more workspaces and switch between them.
3. The execution has tools to show the results of the analysis collections. Either send to the explorer, goto the area in the map or highlight it in different colors.

# Open, Close and Save

The AF uses a dataset to store the analysis collections. When the AF is started for the first time, there is no dataset yet. So you first have to create a dataset by using File|New … The dataset can be created on a local drive or a network drive. The AF needs single user access to the dataset, the dataset can not be shared with other users.  
When the AF is started for the second time the last dataset is opened automatically.

The title of the main framework will show the filename and location of the dataset.



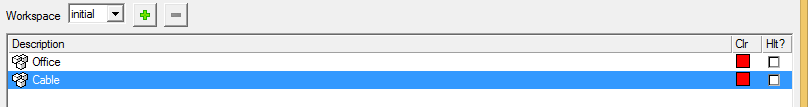
If any changes are made in the framework then title show asterisks to indicate change.



Save your work to avoid losing changes.

# Workspaces

The AF uses workspaces to organize work. By default there is one workspace named “initial”.



Click the plus  to add a new workspace. Note that names of workspace should be unique.

Click the minus  to delete the current workspace, including all the collections. Note that you cannot delete the last workspace.

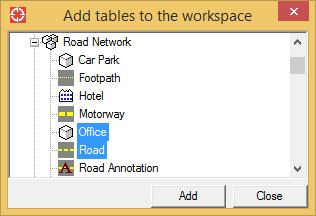
# Base collections

Each analysis will start with some base collection at its core. Either is a trail, some selection or a smallworld collection.

These collections are created using the these toolbar items: 

## Add Smallworld collection

This will open a dialog to select one or more Smallworld collections. Press Add to add them to workspace (or doubleclick).



## Add trail

This will add the trail to the workspace. If the trail is closed then an area collection is added, otherwise a geometry collection. An area collection can typically answer questions about what is inside, a geometry collection is more likely used to answer what is near.

For a single point trail



For a closed trail

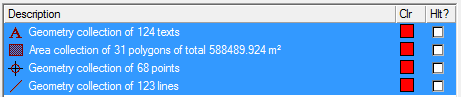


## Add selection



This will add the selection in the main map to the workspace. Each geometry type will be converted to a separate geometry collection, except for the areas who will be combined in one area collection.

This is the result of a large selection in the Cambridge environment:



## Add selection collections



This will add the collections of the selection in the map. This way you don’t have to lookup the collection in a large list but you can rather simply select the an element from the collection in the map.

## Add from explorer



This will add the elements from the explorer in the workspace. If you have a lot of records in the explore this will take a long time.

## Add from explorer selection



This will add the selected elements from the explorer in the workspace. If you have a large selection this will take a long time.

## Add from scrapbook



This will add elements from the scrapbook to the workspace.

# Filter collections

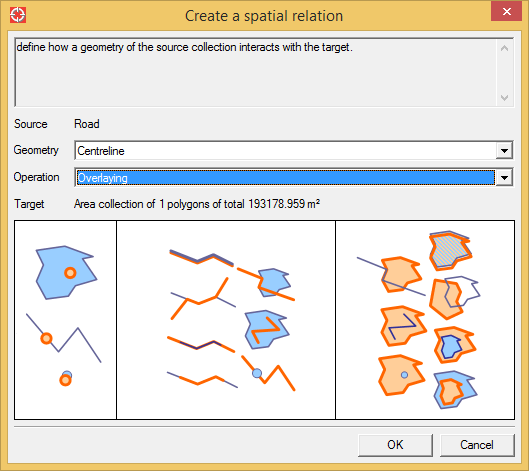
There are three basic operations to filter an analysis collection; either filter by geometry, by distance or by predicate. 

## Filter by geometry

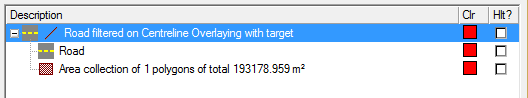


This will filter a collection by only allowing records that have a certain geometrical interaction with another collection. For this to work you will have to select a record collection (the one to be filtered) and an area or spatial collection.

The following dialog will open:



Select the geometry of the source collection that should be tested. Next select the operation that should limit the source. Note that the picture will change for each Operation as to clarify what is meant by it. Click Ok to acknowledge:

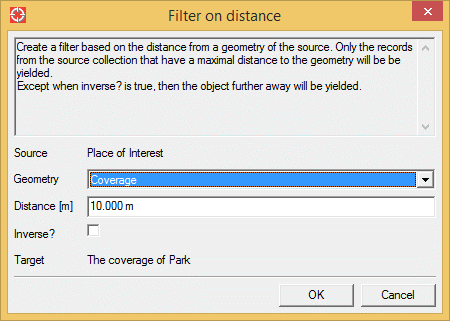


If you want to combine two record collections, then you have to convert one of them to a geometry collection using .

## Filter by distance

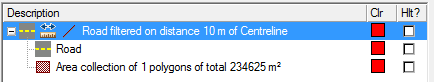
This will filter a collection by checking for the distance to another object. Only the records that are near the object will be yielded. Use this filter to find objects near another one. You have to select a record collection for this to work since they are the only one having records.

The following dialog will open:



Select the geometry field to apply to and enter the distance to use. If you want the objects outside the given distance then toggle Inverse?. By default the object within the given distance are returned.

Click Ok to acknowledge:

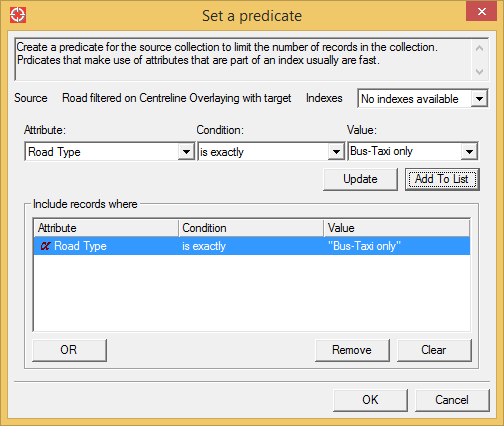


## Filter by predicate



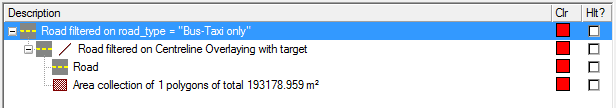
This will filter a collection by applying a predicate to it. You have to select a record collection for this to work since they are the only one having records to test predicates.

The following dialog will open:



Construct a predicate to limit the records that are found in the source collection. Note that there is list of indexes available to the collection. The list is only there for information, predicates based on indexes are much faster than others.

Click Ok to acknowledge:



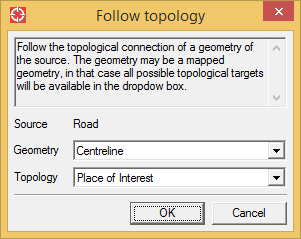
# Follow collections

Follow collections travel from one table to another using different mechanisms. Either by following topology, following a join, executing a method or by yield root records. 

## Topology collection

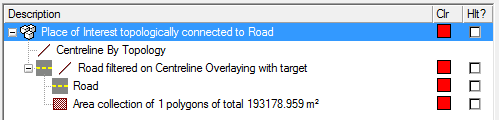
This will yield the topological connected objects from a chosen table. You have to select a record collection for this to work.

The following dialog will open:



Select the geometry to test and the collection to yield.

Click Ok to acknowledge:

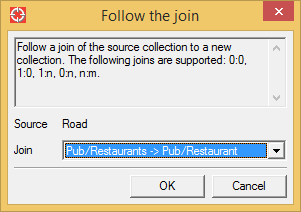


Note that the collection will not yield the same target object twice.

## Join collection

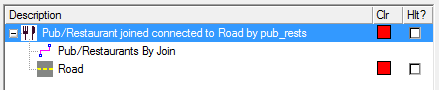
This will yield the joined objects from the chosen table. You have to select a record collection for this to work.

The following dialog will open:



Select the join to follow into the target collection. Note that heterogeneous joins are not supported.

Click Ok to acknowledge:

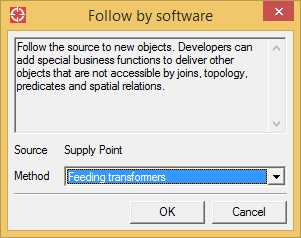


Note that the collection will not yield the same target object twice.

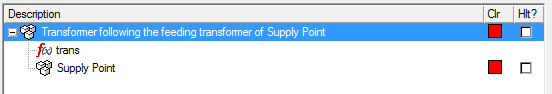
## Method collection

This will create a collection that uses a customized method to yield records from another table. The developer should provide the software for this to work. Basically anything can be provided here what can be programmed. For example on the collection of supply point could be a method that yields the feeding transformers.

The following dialog will open:



Select the method to invoke and click OK to acknowledge:



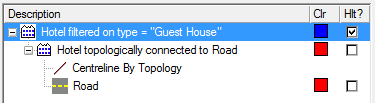
Note that the collection will not yield the same target object twice.

## Root collection

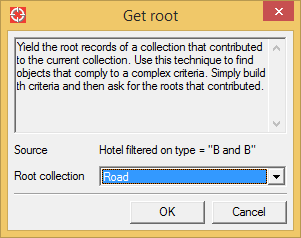
This will yield the root records of one of the collections that make up the selected collection. An example can clarify this:

Suppose you have an analysis collection that follows the topology of roads to the hotels with a predicate on type. This will give you all the hotels you need, but suppose you now want the roads that led to the hotels in question. That is where the root comes in.

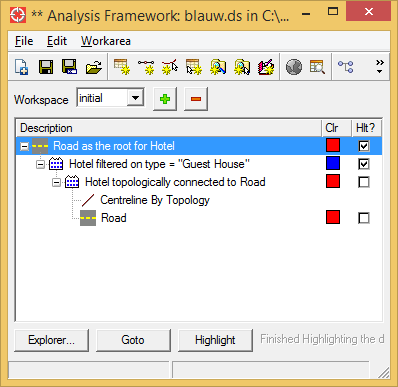
Suppose you have the following setup:



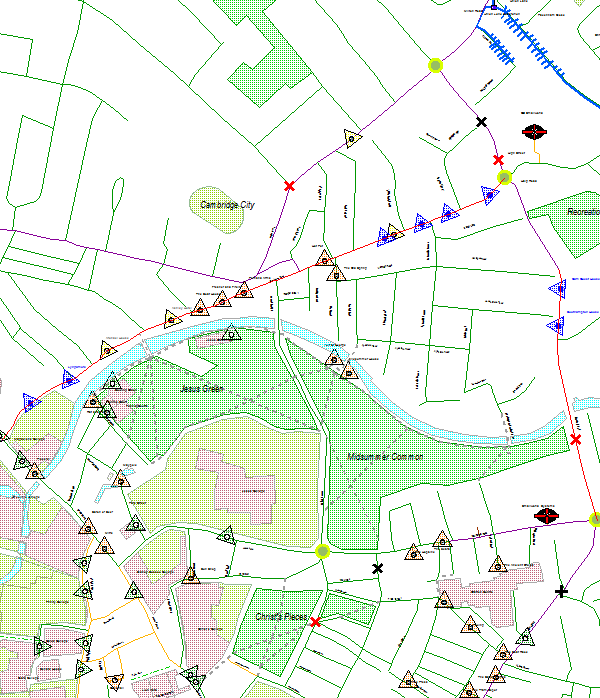
Select the top analysis collection and invoke the root dialog. The following dialog will open:



Click OK to acknowledge:



When highlighting this leads to the following map:



This looks correct, the blue hotels are the hotels with correct type and the red lines are the road connected to it. Note, again, that the root collection will not yield the same target object twice.

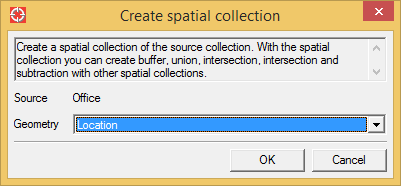
# Geometry operators

There are two functions that create geometry collections, one simply selected one geometry from a collection and the other builds a buffer around a geometry. 

## Geometry collection



Select a record collection and invoke the dialog. The following dialog will open:



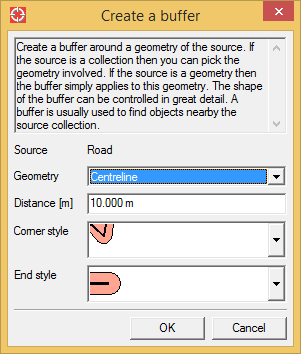
Select the geometry to require and click OK to acknowledge:



## Buffer collection



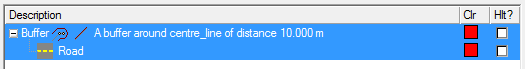
Select a collection and invoke the dialog to create a buffer around it. The following dialog will open:



Select the geometry that you want to buffer. Next enter the distance the buffer should be from the geometry.

Optionally you can select the way the corner and the ends are modeled.

Click OK to acknowledge:



Note that the buffer function is rather time consuming so you want to thinks about materializing it for further analysis. Also note that if you only are using the buffer to find objects nearby you rather should use the distance filter because that is more efficient with memory and performance.

# Set operators

The AF provides basic set operators, namely intersection, subtraction, union, xor: 

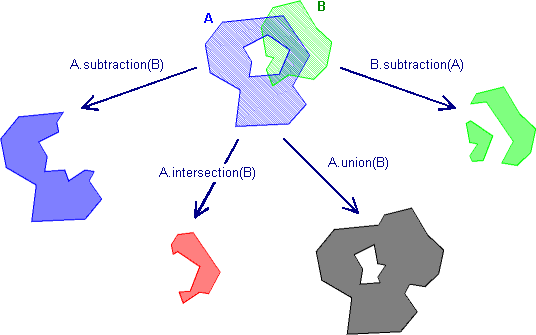
* Intersection: yield the elements or the part that is contained in both.
* Subtraction: yield the elements or the part that is in one, but not in the other. Obviously the order of the collections of important here.
* Union: yield the elements or the part than is combination of both.
* Xor: yield the elements or the part that are unique to each element.

These operators can be applies to record collections, area collections and spatial collections.

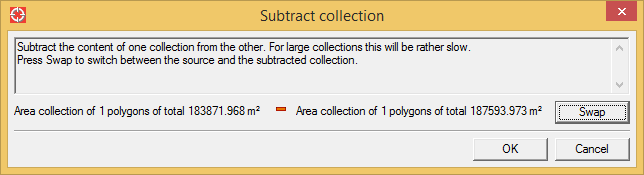
## Area collections

All area collection can be combined with these operators.

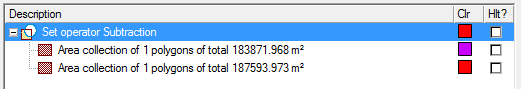
For area operators:



For subtraction the order of the operation is important. So for that function a dialog is available:



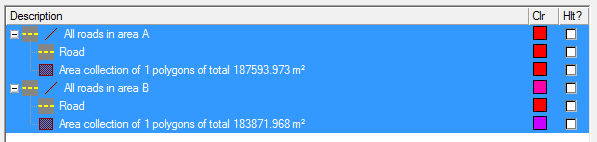
Accept the order of the operation or click Swap to swap the order. Click OK to acknowledge:



## Record collections

Set operators only make sense for records collections of the same base collection.

If the starting set of collections is like this:



Then a set operator can detect which roads belong to both sets.

## Spatial collections

Set operators only make sense for spatial collections when they are dealing with the same type of geometries. There is no sense in subtracting point from lines, because they will never match.

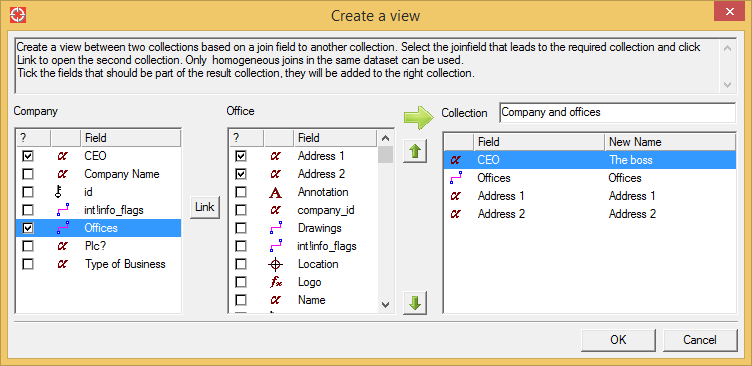
# Views

A view is a collection than combines the attributes of two collection. The AF supports two type of views, views defined by a join and views defined by a common attributes. 

## View on join

This creates a collection that is a combination of attributes from two collections that are joined by a join. This function only works on analysis collections that directly relate to a smallworld collection, it will not work on a filtered collection, set collection, etc.

The following dialog will open:



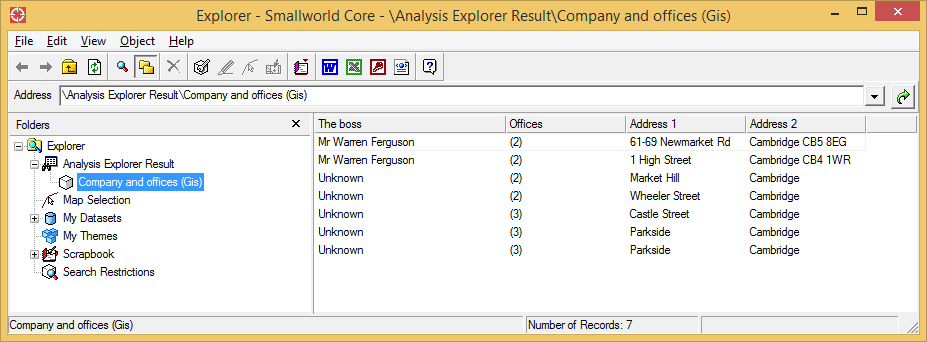
In the left list you must select a join field and click the Link button. This will fill the middle list with attribute values. Note that heterogeneous joins are not allowed, neither are cross dataset joins.

* Next check the boxes of the fields you want to appear in the target collection. You can include geometry fields.  
  You should enter the name of the new collection at the label “Collection”. You can use spaces in the name, the AF will create a correct internal name for you.
* You can also rename the target fields by clicking in the column “New Name”.

Press ok to create a new collection in the source dataset. Note that the view is created in the source dataset not in the analysis dataset. In the example above the view collection will be added to the GIS dataset.

The resulting collection can be send to the explorer or highlight, but also become the base collection for further analysis, filtering, etc.

The resulting records can be send to the explorer and they will look like this:



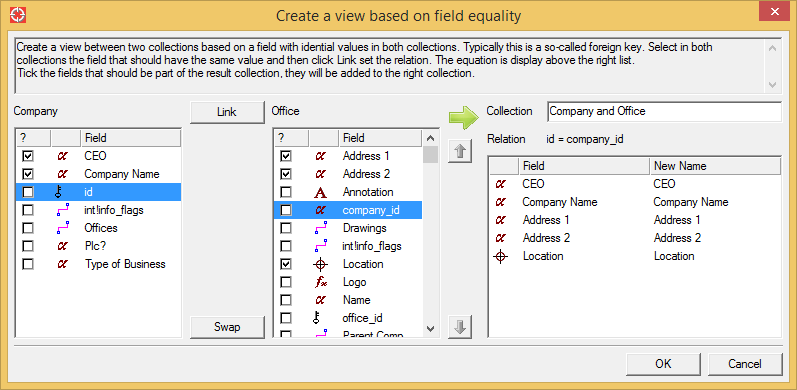
## View based on common attribute



This creates a collection that is a combination of attributes from two collections that are joined by a common attributes. This function only works on analysis collections that directly relate to a smallworld collection, it will not work on a filtered collection, set collection, etc.

You need to select two collections for this function.

The following dialog will open:

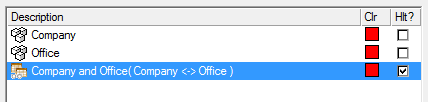


If both the left and right list you should select the attribute that is used to match the collections. In this example the id from company is selected and the company\_id from office is selected. Next click the Link button to link the collection. The resulting equation is displayed at the “Relation” label.

* Next check the boxes of the fields you want to appear in the target collection. You can include geometry fields.  
  You should enter the name of the new collection at the label “Collection”. You can use spaces in the name, the AF will create a correct internal name for you.
* You can also rename the target fields by clicking in the column “New Name”.

Press ok to create a new collection in the source dataset. Note that the view is created in the source dataset not in the analysis dataset. In the example above the view collection will be added to the GIS dataset.

Click OK to acknowledge:



The resulting collection can be send to the explorer or highlight, but also become the base collection for further analysis, filtering, etc.

# Materialize

Most analysis collections are lazy evaluated. This means that you can create them quickly and the performance only comes when you use them in the explorer or when highlighting.

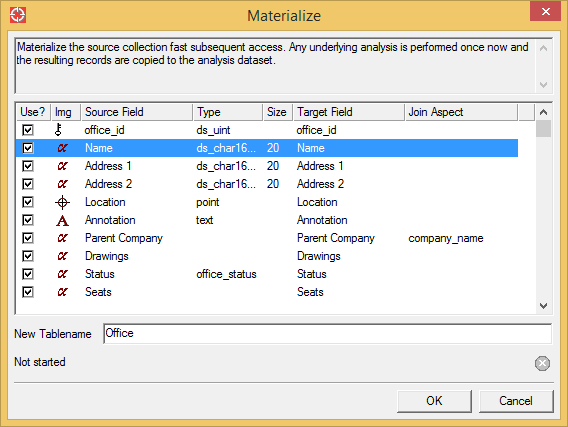
However some analysis will take a long time and you don’t want to repeat the analysis over and over to wait and wait for an answer. In that case you can materialize the analysis collection to the analysis dataset. This involves basically a copy of the collection.

You can materialize a record collection or an area/spatial collection. 

## Materialize record collection

This will copy an analysis collection to the analysis dataset.

The following dialog will open:

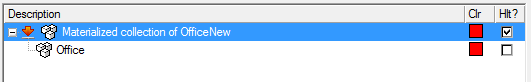


You can do the following:

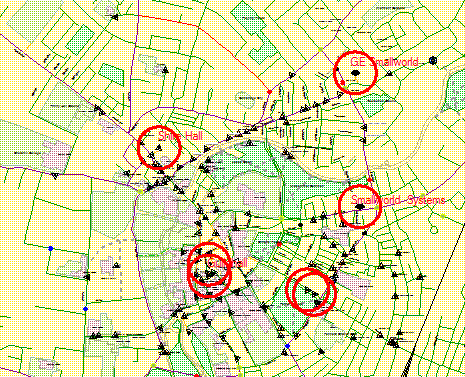
* Check the fields you can in the new collection.
* Rename the fields by clicking in the Target Field column.
* Choose a different join aspect field for the 1:1 joins. In this example the join aspect is company\_name, but if you click in the join aspect column a combobox lets you chose a different value

Note that heterogeneous joins are not support. 1:n joins will automatically be converted to integers and the field will contain the number of matching elements.

Click OK to acknowledge. The label at the bottom will inform you of the progress of the operation as this may take a long time. You can abort the operation by clicking the abort button.



The resulting collection can be send to the explorer or highlight, but also become the base collection for further analysis, filtering, etc. Note that the highlight function will not draw the objects as you might expect them to do. Because the table is a copy of the original, the styling will not match the originals. This will typically be noticeable for the point objects. The highlight points are big red circles.

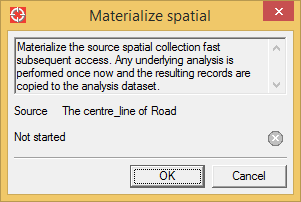


## Materialize geometry collection

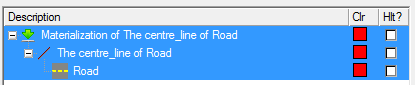


This will copy a geometry or area collection to the analysis dataset. Select the appropriate collection and invoke the dialog.

The following dialog will open:



Click OK to acknowledge. The geometry will be materialized an notepad will open to show the results of the process.



# Utilities

Will delete the current analysis collection. Note that for large materialized collections this can take a while.



Will copy the collection Note that not really the contents is copied but rather a new reference to the same content is made. This means that a copy of a large materialized collection is very fast.



Will open a dialog to pick a color to highlight the selected analysis collection.



Will move the current selected analysis collection up in the list

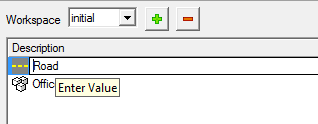


Will move the current selected analysis collection down in the list.



## rename

Each analysis collection can be renamed. Simple click on the name and enter another description.



# Execution

The execution tools are at the bottom of the AF.



There are 6 tools to evaluate your analysis results: explorer, goto, map, excel, access and report. These can all be time consuming so they are performed in a background thread. During execution a timer shows the elapsed time and you can interrupt the execution with an abort button.

## Explorer

This obviously only works for analysis collections that have records. It makes no sense for buffers or geometry collections. The explorer will use lazy evaluation when possible, this means that the first couple of records should appear pretty quick. When you scroll down the analysis collection is evaluated further to yield the follow results.

Note that the first time the explorer is used there is a significant delay to start the explorer. So be patient…

## Goto

This will zoom the current map to the extend of the results of the analysis collection. Note that for this function the entire analysis collection has to be scanned to construct the bounds.

## Map

This will bring the map to the foreground and highlight all the checked items in the workspace with the indicated color in the main map. Use the  button to select the highlight color of the selected analysis collection.  
If you want to clear the highlight in the map, un-check the analysis collections and click refresh the map again.

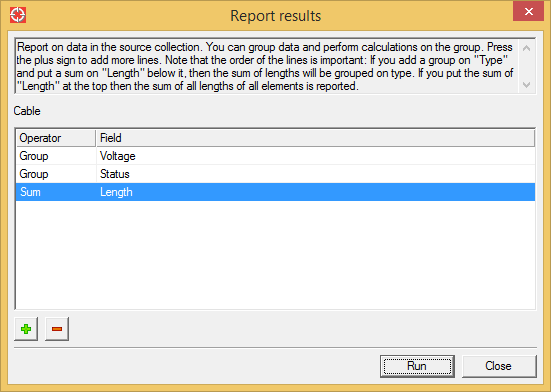
The highlight will be optimized for performance (if possible) to only evaluate the elements that are visible in the map.

## Excel/Access

This will export the collections to Excel or Access. The function is similar to exporting to the Explorer and then export it to Excel or Access.

## Report

This will open the Report dialog.



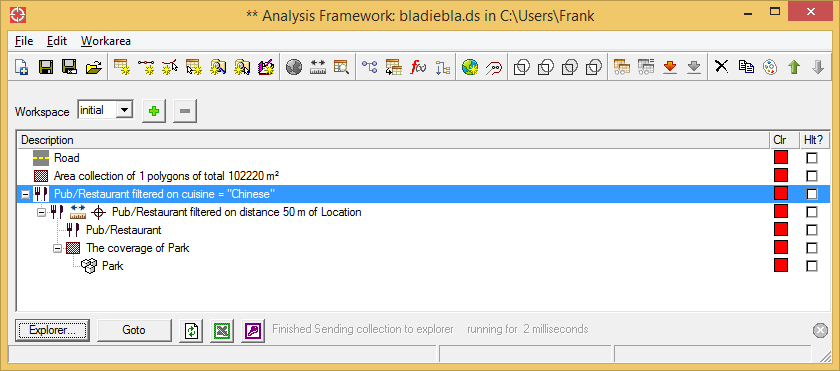
The report has two basic operators, either Group or a calculation.

The report revolves around calculations, it can sum values, count records, etc. To group the results you can add Group operators above the calculations. In the example above the sum of the cables is grouped by the Status and Voltage. So the report will show what the length is of the cables with voltage “LV” and status “In service” and any other combination of values.

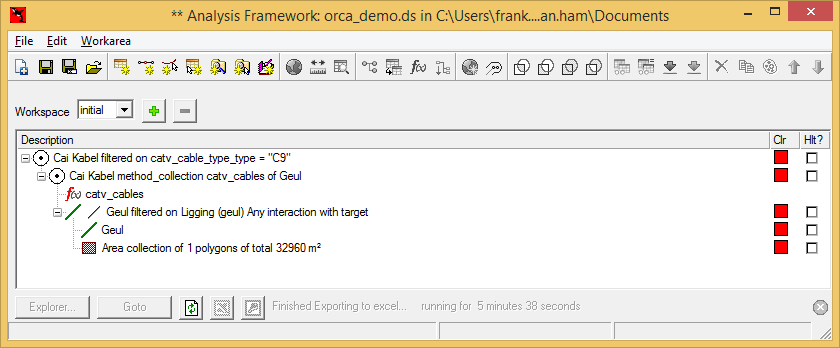
The following calculations are supported:

* Min: Reports the minimum value of the attribute.
* Max: Reports the maximum value of the attribute.
* Sum: Reports the sum of the values of the attribute.
* Count: Counts the number of records where the attribute has a value. If no attribute is given, then this will report the count of records.
* Average: Reports the average of the values of the attribute. Note that unset values are skipped in the evaluation, so the average of {2, unset, 2} is 2, not 4/3.

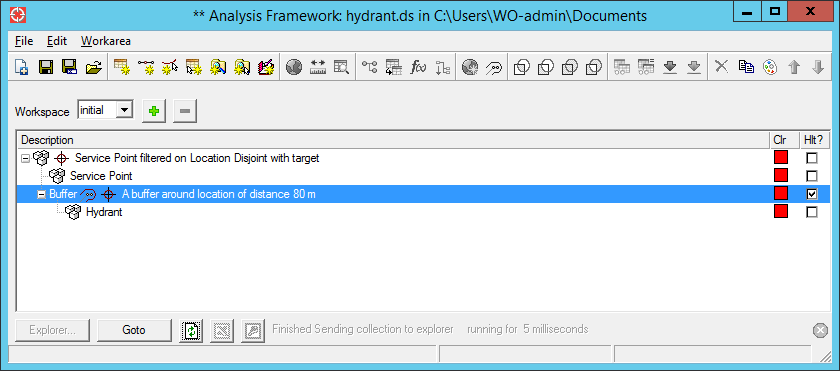
# Examples

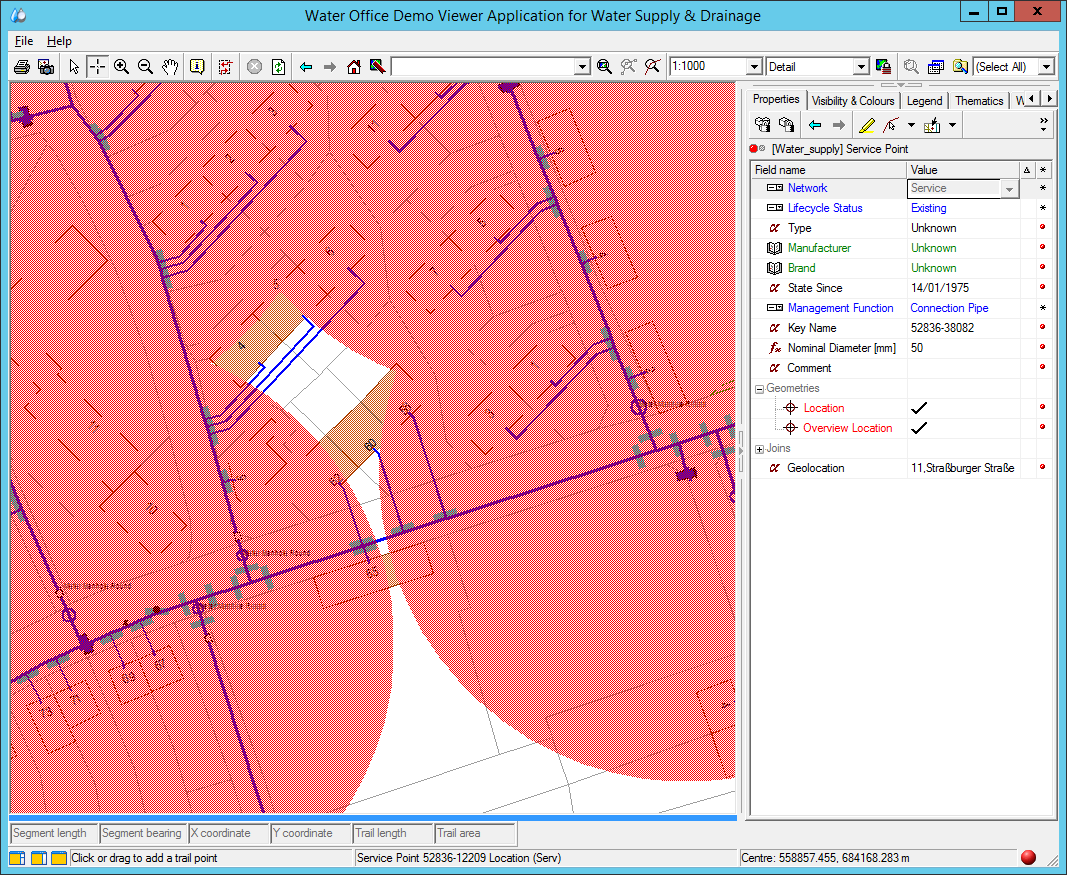


Cambridge has 3 restaurants with Chinese cuisine within 50 m of park.



ORCA has 2973 C9 cables in Waalwijk, with a total length of 185987.262 metres.

 Water office has 126 servicepoints that are further away than 80 metres from any hydrant.



# Unit Test

The delivery comes with several unit tests that run in the Cambridge environment. To run the unit tests do the following:

* Add the munit product. The product is not included in this delivery, it should be available at your company if you are serious about development.
* Load the module rw\_analysis\_workspace\_test
* Load the script drafting\_creation.magik in a writable GIS alternative. This will create areas and text labels that are used by the unit tests. Commit the dataset.
* On the prompt type: simple\_munit\_gui.open()
* Select rw\_analysis\_workspace and click on the green run button.

