GISwater

PROTOCOL DOCUMENT

How to add and configure a new *mapzone*

E-13.2

GOALS

Know which steps must be taken both in QGIS and in the database to add a new ***mapzone*** and subsequently **configure its use and impact**. This always taking into account the use of **dynamic** *mapzones*, managed by the user but created from an internal **Giswater algorithm**.

DESCRIPTION

Below are the steps to add a new mapzone. For the different types that exist, the steps to follow are exactly the same.

Background

1. In system variable ***utils_dynamicmapzones_status***, *om_dynamicmapzones_status (v3.3)* define the graph classes that are enabled:

{"SECTOR":true, "PRESSZONE":true, "DQA":true, "MINSECTOR":true, "DMA":true}

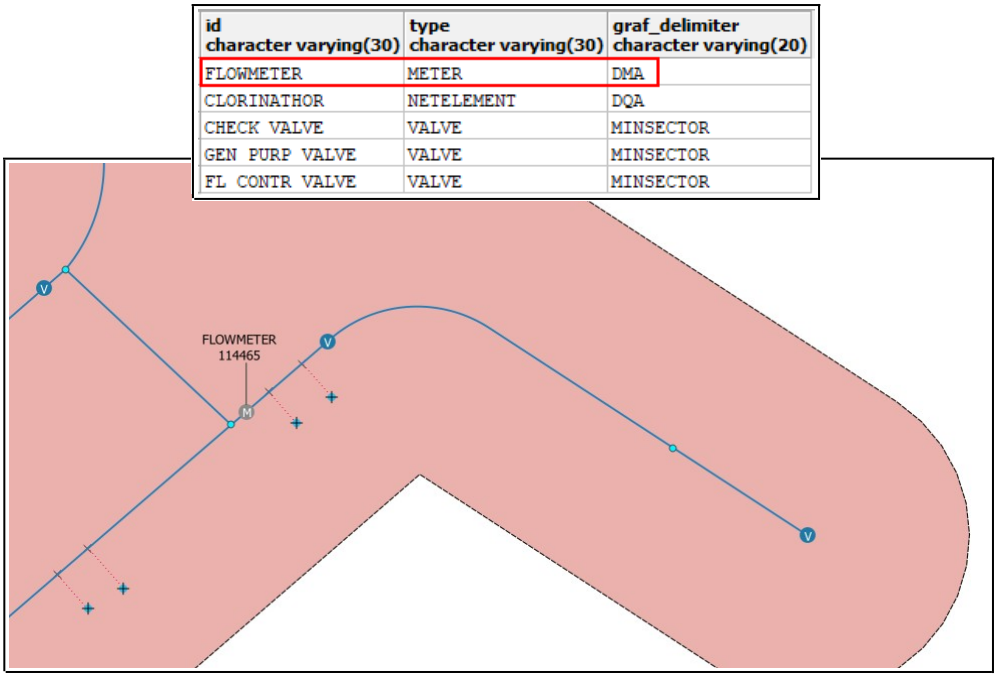
2. In the ***cat_feature_node table***, *node_type (v3.3)*, for each type of node we define graph_delimiter. The sample has:

	id [PK] character varying(30)	type character varying(30)	epa cha	nu int	ch bc	isarcdivid boolean	graph_delimiter character varying(20)	isprofilesurface boolean
1	ADAPTATION	JUNCTION	JUI	2	TF	TRUE	NONE	TRUE
2	AIR_VALVE	VALVE	UNI	0	TF	FALSE	NONE	TRUE
3	BYPASS_REGISTER	REGISTER	JUI	2	TF	TRUE	NONE	TRUE
4	CHECK_VALVE	VALVE	SH(	2	TF	TRUE	MINSECTOR	TRUE
5	CLORINATHOR	NETELEMENT	JUI	2	TF	TRUE	DQA	TRUE
6	CONTROL_REGISTER	REGISTER	JUI	2	TF	TRUE	NONE	TRUE
7	CURVE	JUNCTION	JUI	2	TF	TRUE	NONE	TRUE
8	ENDLINE	JUNCTION	JUI	1	TF	TRUE	NONE	TRUE
9	EXPANTANK	EXPANSIONTANK	JUI	2	TF	TRUE	NONE	TRUE
10	FILTER	FILTER	JUI	2	TF	TRUE	NONE	TRUE
11	FL_CONTR_VALVE	VALVE	VA	2	TF	TRUE	MINSECTOR	TRUE
12	FLEXUNION	FLEXUNION	JUI	2	TF	TRUE	NONE	TRUE
13	FLOWMETER	METER	JUI	2	TF	TRUE	DMA	TRUE
14	GEN_PURP_VALVE	VALVE	VA	2	TF	TRUE	MINSECTOR	TRUE
15	GREEN_VALVE	VALVE	JUI	2	TF	TRUE	NONE	TRUE
16	HYDRANT	HYDRANT	JUI	2	TF	TRUE	NONE	TRUE
17	JUNCTION	JUNCTION	JUI	2	TF	TRUE	NONE	TRUE
18	MANHOLE	MANHOLE	JUI	2	TF	TRUE	NONE	TRUE
19	NETELEMENT	NETELEMENT	JUI	2	TF	TRUE	NONE	TRUE
20	NETSAMPLEPOINT	NETSAMPLEPOINT	JUI	2	TF	TRUE	NONE	TRUE
21	OUTFALL_VALVE	VALVE	JUI	2	TF	TRUE	NONE	TRUE
22	PR_BREAK_VALVE	VALVE	VA	2	TF	TRUE	PRESSZONE	TRUE
23	PR_REDUC_VALVE	VALVE	VA	2	TF	TRUE	PRESSZONE	TRUE
24	PR_SUSTA_VALVE	VALVE	VA	2	TF	TRUE	PRESSZONE	TRUE
25	PRESSURE_METER	METER	JUI	2	TF	TRUE	NONE	TRUE
26	PUMP	PUMP	PUI	2	TF	TRUE	PRESSZONE	TRUE
27	REDUCTION	REDUCTION	JUI	2	TF	TRUE	NONE	TRUE
28	REGISTER	REGISTER	JUI	2	TF	TRUE	NONE	TRUE
29	SHUTOFF_VALVE	VALVE	SH(	2	TF	TRUE	MINSECTOR	TRUE
30	SOURCE	SOURCE	RE:	2	TF	TRUE	SECTOR	TRUE
31	T	JUNCTION	JUI	3	TF	TRUE	NONE	TRUE
32	TANK	TANK	TA	9	TF	TRUE	SECTOR	TRUE
33	THROTTLE_VALVE	VALVE	VA	2	TF	TRUE	NONE	TRUE
34	VALVE_REGISTER	REGISTER	JUI	2	TF	TRUE	NONE	TRUE
35	WATER_CONNECTION	NETWJOIN	JUI	2	TF	TRUE	NONE	TRUE
36	WATERWELL	WATERWELL	RE:	2	TF	TRUE	SECTOR	TRUE
37	WTP	WTP	RE:	2	TF	TRUE	SECTOR	TRUE
38	X	JUNCTION	JUI	4	TF	TRUE	NONE	TRUE

3. In the valve configuration table ***config_valve***, *anl_mincut_selector_valve (v3.3)* define the valves that act as stoppers

Example to add a new DMA type mapzone:

1) Add in QGIS a **new node** in the desired location. The type of node must be one that in the *node_type* table has as a value in the column *graph_delimiter* '**DMA**', since this column establishes which areas of the map can delimit the different types of node. If we want to make an area of the map **different** than dma, the only necessary thing to do is to insert a node that is the delimiter of the desired area. For DMA it will usually be 'METER' type nodes.



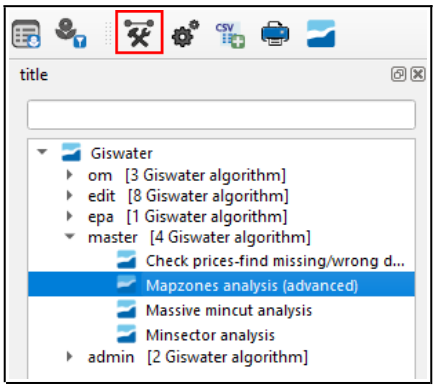
2) Add to the dma table (if we make another *mapzone* fill the corresponding table) a new value, with the name of the dma that we want to create and, above all, fill the **grafconfig** value with the following ***json***. In the value of **nodeParent** we must put the id of the node that we just inserted, which will be the header of the new dma and in **toArc** we indicate the direction to take by means of the id of the section. Example:

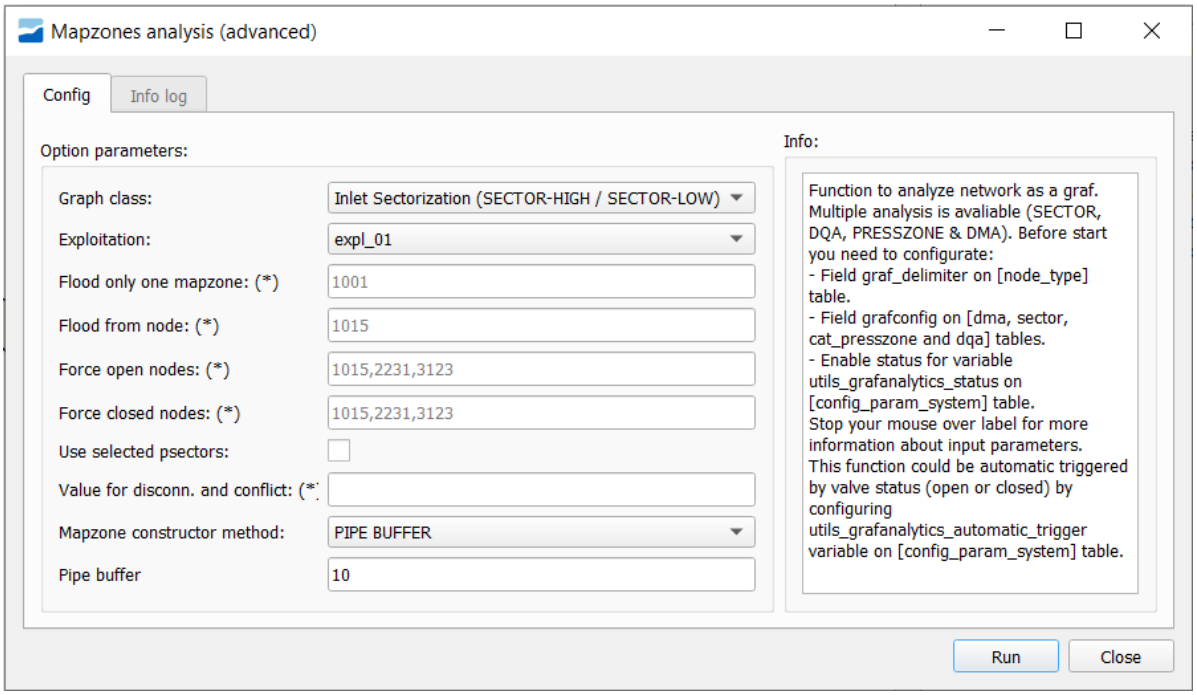
```
{"use":[{"nodeParent":"1080", "toArc":[2092]}], "ignore":[], "stopper":[1057,41,1060]}
```

In **ignore** (optional) we will put, if it is the case, those nodes that being graphdelimiter, we do not want them to participate in the algorithm (for being out of service or for not doing its job).

In **stopper** (optional) we will put, if it is the case, those nodes that we want to force stop the flood algorithm.

3) We use the **Giswater toolbox** to **recalculate** the *mapzones* with the modification that we have made for dma's, by selecting the shape of the geometry that we want.





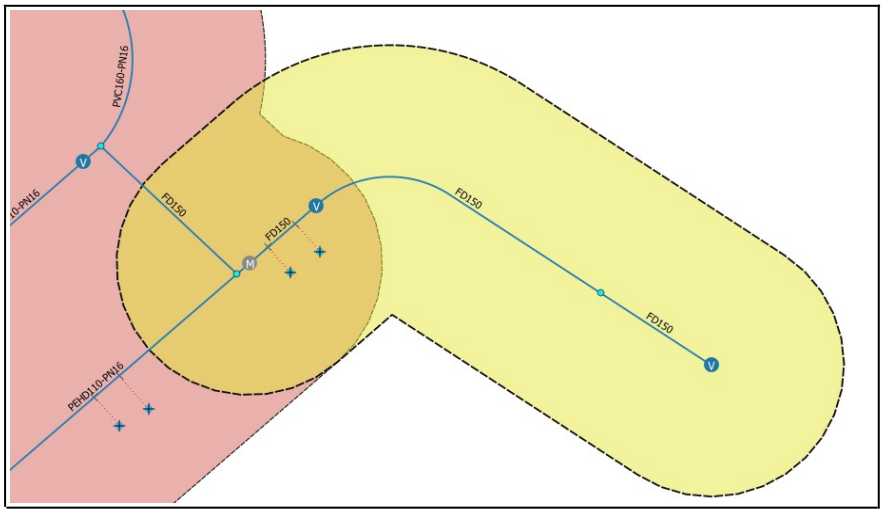
An alternative to using the Giswater toolbox, is to shoot the function directly from the database, making a call such that:

```
SELECT gw_fct_graphanalytics_mapzones('{"data":{"parameters":{"grafClass":"DMA",
"exploitation":[1], "macroExploitation":[1], "checkData":false,
"updateFeature":true, "updateMapZone":2, "geomParamUpdate":15,"debug":false,
"usePlanPsector":false, "forceOpen":[1,2,3], "forceClosed":[2,3,4]}}}');
```

where:

KEY	OBLIG.	DESCRIPTION	EXAMPLE	VALUE RANGE
graphClass	YES	Graph class to make	DMA	DMA, SECTOR, DQA, PRESSZONE
exploitation	YES	Exploitations to participate in the algorithm	[1,2]	All the available exploitations
macroExploitation	NO	Macroexplotations to participate in the algorithm	[1,2]	All the available macroexplotations
checkData	YES	If true, check if the system values are correct (topology, state_type, etc), In case there are errors, abort the process	false	false, true
updateFeature	YES	If true, update the values of dma_id, presszone_id, sector_id & dma_id of all NODES, ARCS, CONNEC that are flooded by the algorithm	true	false, true
updateMapZone	YES	0: does not update the geometry field (the_geom) of the mapzone 1: Updates by making an enveloping polygon with all the elements 2: Updates by buffering the sections with the value of geomParamUpdate 3: Updates by buffering the sections with the value of geomParamUpdate and incorporating the plot geometry (if it exists)	2	0,1,2,3
geomParamUpdate	YES	Value related to the options 2, 3 of the previous key	10	Any float between 0.1 – 100
usePlanPsector	YES	If true, use every psectors of the exploitation in the alogrithm analysis	false	false, true
forceOpen	YES	Valves that can be forced to open (e.g. for closed valves that for whatever reason we want to open)	[1,2,3]	Every closed valves
forceClosed	YES	Nodes in general which can be forced to close (e.g. in the debugging phase in case the trace gets out of control and does not converge as expected)	[1,2,3]	Every nodes (except closed valves)

4) Once the **process is finished**, we must check again the geometries of the **dma**. In our example, we see that the geometry has been generated for the new dma that has the FLOWMETER that we have added as a delimiter node.



REVIEWS

Action	User	Date
Created	Albert Bofill	09/04/2020
Updated	Xavier Torret	01/08/2020
Updated	Xavier Torret	01/12/2021