



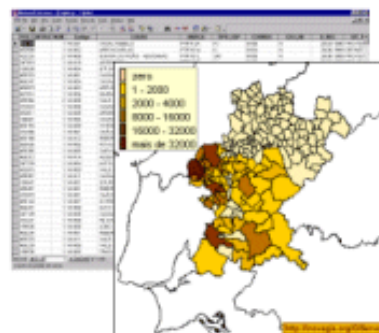
---

<b>Original File</b>	ShapeODBC.htm
<b>Abstract</b>	Help guide for the ShapeODBC service.
<b>Author</b>	Pedro Pereira Gonçalves ( <a href="mailto:pedro@inovagis.org">email</a> )
<b>Implemented Functions</b>	
<b>Last Change</b>	
<b>History</b>	2002-03-05 : File Created
<b>Index Page</b>	GIServer <a href="#">tutorial</a>
<b>Important Links:</b>	

---

The ShapeODBC service is an extension to GIServer that allows you to dynamically update your geographic data with values from ODBC database without the need to physically update the shape file.

This means that you can query an ODBC database with multiple tables using SQL and dynamically display the desired information in your WMS layer. The data can be a simple update of the resulting query values or several operation can be performed like COUNT (counts the number of ODBC records that obey your query for that geographic record) or ADD ( sums the value of the record sets found) with the values from the same or different ODBC table's fields.



This service is made in ASP (Active Server Pages) and the code is included in the professional version of GIServer so that you can adapt it to your project or convert it to the Windows versions of PHP or PERL.

Like in a normal GIServer request, it accepts the [WMS parameters](#), the [vectorial specific parameters](#) and the specific ShapeODBC parameters that are equally sent in relation to the requested spatial layer. The key field in the shape file is the one defined with the [field](#) parameter

The request parameters specific of ShapeODBC must include the prefix of the GIServer layer name like `<dataset>.<layer>` and are the following :

- **.ODBC (required)**  
Defines the name of the ODBC to be used.
- **.ODBC.TABLE (required)**  
Defines the table, tables or JOIN table of the selected ODBC.

Examples :

`<dataset>.<layer>.ODBC.TABLE = Table1`

or even using a JOIN

`<dataset>.<layer>.ODBC.TABLE = Table1+INNER+JOIN+Table2+ON+Table1.ID=Table2.ID`

- **.ODBC.FIELD (optional)**  
Defines the ODBC field that will act as the key to the shape field. If not present the default value is equal to the vectorial [field](#) parameter.

Examples :

`<dataset>.<layer>.ODBC.FIELD = ID`

or (when using JOIN)  
<dataset>.<layer>.ODBC.FIELD = TABLE1.ID

- **.ODBC.QUERY (optional)**

Its a optional parameter to extend the query the ODBC from the simple shape key / ODBC key query.

- **.ODBC.QUERYFIELDS (optional)**

Its a optional parameter to restrict the fields to query. Default value is \*

- **.ODBC.OPERATION (optional)**

This parameter allows the ShapeODBC to use a different operation to the shape value update. Possible values are :

- COUNT - Its the default value for this service and updates the shape record with the number of ODBC records found for the requested queries
- UPDATE - Updates the shape record with the value of the first ODBC records found for the requested queries
- ADD - Updates the shape record with the sum of the values of the ODBC records found for the requested queries

- **.ODBC.OPERATIONFIELD (optional)**

To be used with the UPDATE and ADD operations. It expresses the field name or number (if QUERYFIELDS parameter is equal to \*) to be used in the operation. The default value is the ODBC.FIELD and if this one is not present is the SHAPE field.

- **.INITVAL (optional)**

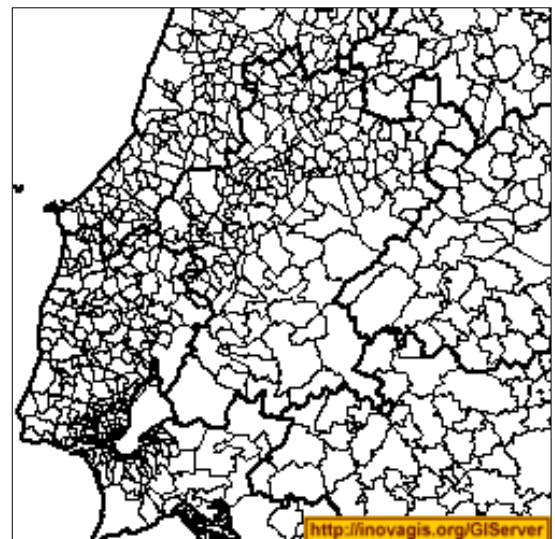
Defines the initial/default value of the updated record. This should be use for instance when using the operation ADD and the initial value is not equal to zero.

---

## Example Project

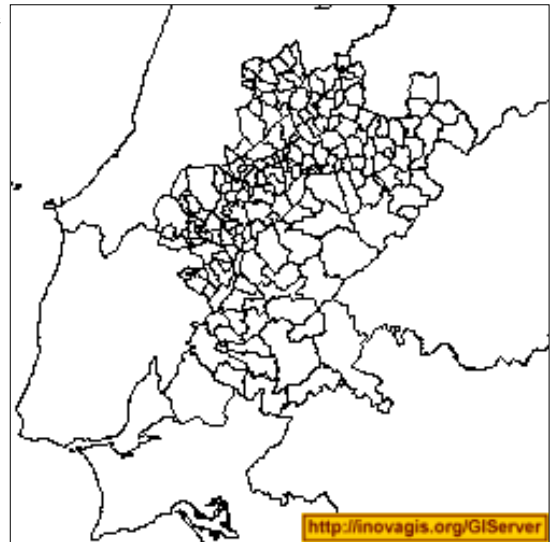
In this example we have two layers, PORTUGAL.FREG and PORTUGAL.DIST250. The first refers to the Portuguese *freguesias* (the smallest administrative unit) and the later is the *distritos* (more or less equivalent to regions)

```
REQUEST=GETMAP&  
LAYERS=PORTUGAL.FREG,PORTUGAL.DIST250&  
WIDTH=250&  
HEIGHT=250&  
BBOX=80000,160000,250000,330000&  
EPSG=SHGM&  
FORMAT=GIF&  
STYLES=,STRONG
```



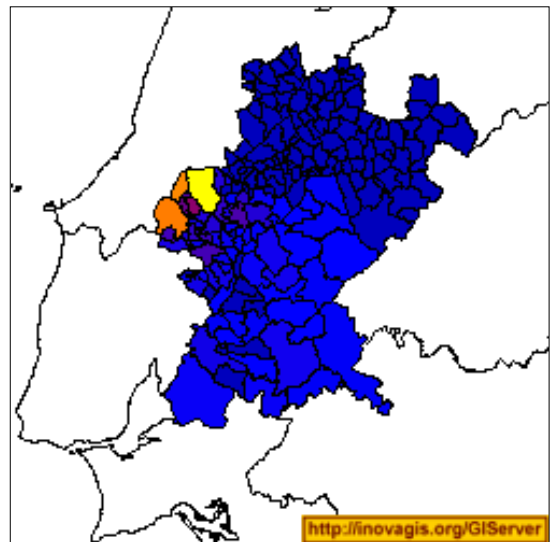
We can query the *freguesias* dataset so that only the codes inside the Santarém *distrito* are selected, i.e. where its DTCCFR code is inside [140000,150000]

```
REQUEST=GETMAP&
LAYERS=PORTUGAL.FREG,PORTUGAL.DIST250&
WIDTH=250&
HEIGHT=250&
BBOX=80000,160000,250000,330000&
EPSG=SHGM&
FORMAT=GIF&
PORTUGAL.FREG.QUERY=(DTCCFR>140000) AND
(DTCCFR<150000)
```



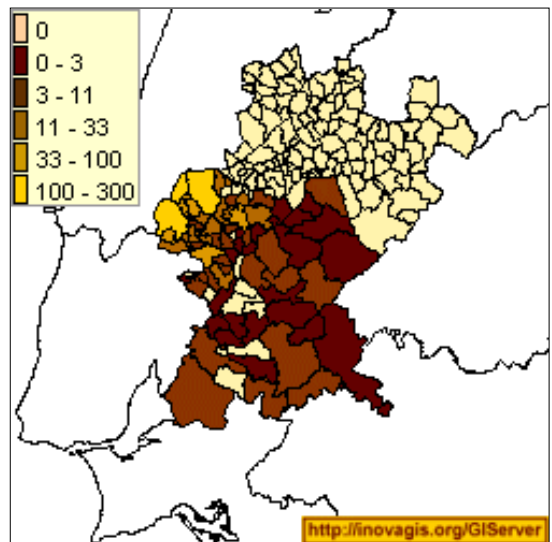
Now we want to map how many agricultural explorations with pigs exist in each *freguesia* of this *distrito*. This data is available from the ODBC called FMV\_SUINOS in the table EXPLORAC. To map the different *freguesia* codes we have to inform the server that in the shape the *freguesia* code is called **DTCCFR** and in the ODBC is called **CODIGO**.

```
REQUEST=GETMAP&
LAYERS=PORTUGAL.FREG,PORTUGAL.DIST250&
WIDTH=250&
HEIGHT=250&
BBOX=80000,160000,250000,330000&
EPSG=SHGM&
FORMAT=GIF&
PORTUGAL.FREG.QUERY=(DTCCFR>140000) AND
(DTCCFR<150000)
PORTUGAL.FREG.FIELD=DTCCFR&
PORTUGAL.FREG.ODBC=FMV_SUINOS&
PORTUGAL.FREG.ODBC.TABLE=EXPLORAC&
PORTUGAL.FREG.ODBC.FIELD=CODIGO&
```



After this we can add a max and min value for display and legend creation using the automatic colorquery constructor.

```
REQUEST=GETMAP&
LAYERS=PORTUGAL.FREG, PORTUGAL.DIST250,
LEGEND(PORTUGAL.FREG)&
WIDTH=250&
HEIGHT=250&
BBOX=80000,160000,250000,330000&
EPSG=SHGM&
FORMAT=GIF&
PORTUGAL.FREG.QUERY=(DTCCFR>140000) AND
(DTCCFR<150000)
PORTUGAL.FREG.FIELD=DTCCFR&
PORTUGAL.FREG.ODBC=FMV_SUINOS&
PORTUGAL.FREG.ODBC.TABLE=EXPLORAC&
```



PORTUGAL.FREG.ODBC.FIELD=CODIGO&  
PORTUGAL.FREG.MAXVALUE=300&  
PORTUGAL.FREG.MINVALUE=0&  
**PORTUGAL.FREG.COLORQUERY**=AUTO(MULT5,ShapeODBC,0x630000,0xFFCF00,5,0,0xFFF1AD)

If in the end we can create a style called EXAMPLE with all the layer specific tags and we can call the same image with

REQUEST=GETMAP&  
LAYERS=PORTUGAL.FREG, PORTUGAL.DIST250,  
LEGEND(PORTUGAL.FREG)&  
WIDTH=250&  
HEIGHT=250&  
BBOX=80000,160000,250000,330000&  
EPSG=SHGM&  
FORMAT=GIF&  
**STYLES=EXAMPLE,STRONG,EXAMPLE**

