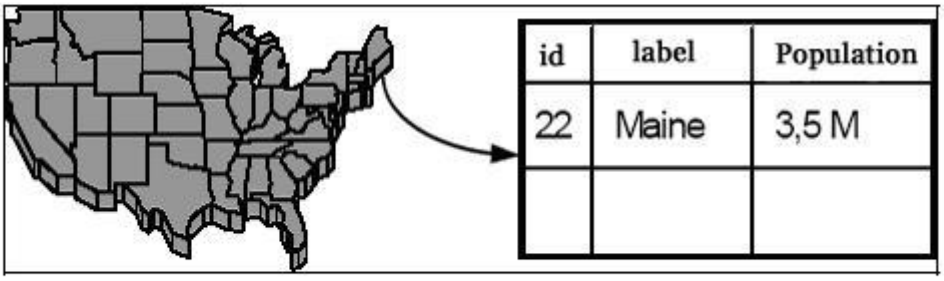
Class 9 –Spatial Analysis and Query

1 – Database Query

The RDBMS (Relational Database Management System) is the link between a geographic information system and a relational database. It is also called a *geo-relational* model where the spatial and descriptive components of the geographic object are stored separately. The conventional attributes are stored on the database (as tables) and the spatial data are handled by a dedicated system. The link is done using object identifiers (**id**).

To retrieve an object the user must search both subsystems and the answer will be a composition of results. This architecture is described on the figure below:



Initially, the user should define a Cadastral category, an Object and its attributes. The file that contains the lines will be imported but the user will be in charge of identifying the objects.

1.1 – Defining the Cadastral and Object Models

⇒ **Creating cadastral and object categories:**

Windows: #Start – Spring<version><Language><system> – Spring<version> <Language>
Linux: # Command to be typed on the Console (Shell) – # s_spring

MAC: #Dock – Launchpad – Spring <version>< language *>*

SPRING

* *Load the database **Course***

* *Load the project **Brasilia***

– [File] [Data model...] or button 

Data model

Categories Tab

– {Categories – Name: Urban_cadastre} – (Model Cadastral) – (Create)

(Execute)

(Close)

[File] [Object/Non Spatial...] or

Object/Non Spatial

Table tab

– {Categories – Name: Lot} – (Model Object) – (Create)

– (Apply) – to store the created categories.

– (Close)

⇒ *Defining attributes of the Object category*

SPRING

- [File] [Object/Non Spatial...] or
- Object/Non Spatial
- Table Tab
- (Categories Lot)
- Attributes Table

Defining Integer attributes

Category Attributes

– {Name: Populat} – *Note: Maximum of 8 characters*

– (Type Integer)

– (Insert)

– {Name: Schools}

– (Type Integer)

– (Insert)

Defining text attributes

– {Name: Type}

– (Type Text)

– (Size: 20)

– (Insert)

Defining real attributes

– {Name: Income}

– (Type Real)

– (Insert)

- (Apply), (Close)
- (Close)

1.2 - Elaboration of cadastral map and object identification

Importing cadastral data:

SPRING

- [File] [Import][Importe Vectorial and Matricial Data]

1Import

- (Directory...: C:\Tutor_10classes\Data) - Windows
  ~/Tutor_10classes/Data - Linux
  ~/Tutor_10classes/Data - MAC
- (Format | ASCII-SPRING: Quadras.L2D)
- (Entity Line with topology), (Unit m), {Scale: 1/ :
25000}

* *Projection and bounding box- not necessary*
* *Project - not necessary, the active project is taken*

Output Tab

- (Category...)

Categories List

- (Categories: Urban_cadastre) *category created above*
- (Apply)

- {PI: Lots_map} *name of Infolayer to be created*
- (Execute)
- (Close)

⇒ **Editing Object Attributes:**

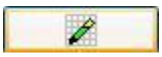
Control Panel

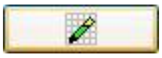
- (<u>Categories</u> | Urban_cadastre)
- (<u>Infolayers</u> | Lots_map)
- (Lines)

SPRING

- [Edit][Tables][Object...]

Object Edition

- (Object...)
- Categories List
- Categories | Lot
- Apply
- (Edit) 

- {Label: Q001}
- {Name: Q001} or (Name = Label)
- (Add) 
- (Edit) 

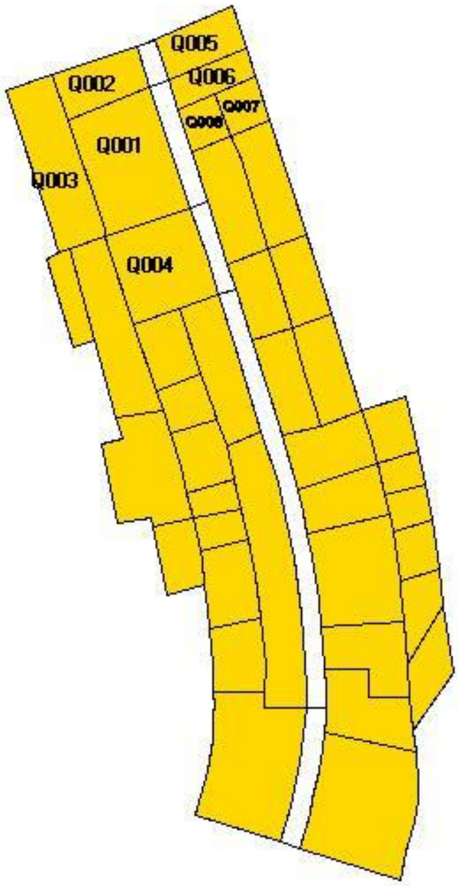
Object Selection

(Operation <=> Associate)

(Entity <=> Polygon)

** Select the object Q001 on the visual plan. If you can not find the polygon, create a topology for this plan.*

- Repeat for the other objects using the following map:



Editing the object attributes

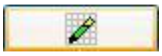
SPRING

- [Edit][Table][Object...]

Object Edition

- (Object...)
- Categories List
- (<u>Category</u> | Lot)

-(Execute)

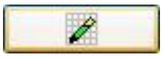
- (Edit) 

- (Associate a Graphic Representation - Select Table)

Select on the screen the object to be edited

The line that represents the object, double-click on the desired column and enter the attribute value.

- (<u>Attributes</u> | Populat)
- {Value: 500} (Enter)
- (<u>Attributes</u> | Schools)
- {Value: 2} (Enter)
- (<u>Attributes</u> | Type)
- {Value: Residential} or Industrial, or Hotels, or Leisure
- (<u>Attributes</u> | Income)
- {Value: 2476.5l} (Enter)

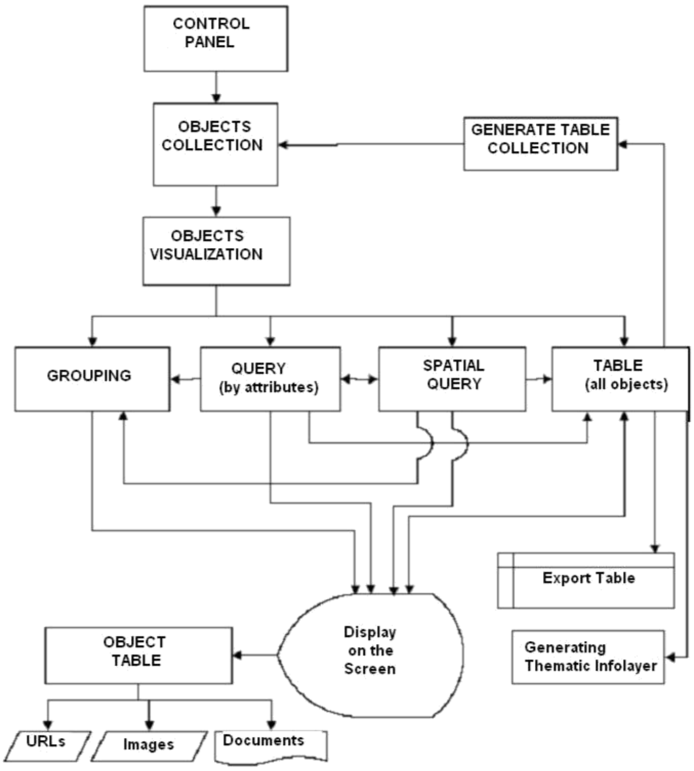
-Edit 

- (Apply)

- *Repeat the procedures above for the other objects.*

1.3 - Query on the edited objects

The query on a map of objects starts with a selection on the "**Control Panel**" followed by the creation of an **Object Collection** and **Visualization Control.** From the Visualization Control, you can access the modules **Query**, **Spatial Query**, **Grouping** and **Table.** With the objects on the **Visualization Screen**, you can process a query on the **Attributes/Images/URL** module. Using the **Table** module you will be able to save its content, generate other collections or thematic plans. The figure below shows the relationships among these modules, as well as the possible paths you can follow when processing a query on its objects.



The figure above shows the dependency of the query modules – they need to be processed in a sequence (see the query examples at the end of this class). Next, we will describe how to use each one of these modules:

1. In the "**Control Panel**", you must define which ILs will be presented on the drawing area specially activating the cadastral or network IL to be queried;

2. On the active chosen IL, you must pick an object category to be queried. It is possible to work with all objects (option **ALL**) present on the map. However, if you wish, you can create a **collection**;

3. The purpose of the **Objects Visualization** module is to command the visualization of the object categories inside a cadastral or network IL. If one collection is defined and applied, only the corresponding objects category will be available in this module. Basically, the control is over how and which objects will be visualized. Besides these controls, this module organizes the exhibition of legends and controls the sequencing of the graphical presentation determining

which object category must be queried, grouped or visualized in table format. This module also determines which object category is active to be analyzed on the screen.

4. The modules **Query**, **Spatial Query**, **Grouping** and **Table** change the graphical presentation of the queried object. Notice on the figure above that all modules reflect a display on the screen, either coming from the objects collection or from a pre-selection by other module. The sequence of operations is important for obtaining the desired result.

1· **Query**: selects objects satisfying a condition imposed by you, based on its descriptive attributes which may be combined on the same expression. The selected objects are presented on the display and table module, and might be further used on the **grouping** or **spatial query** modules

2· **Spatial Query**: select objects based on their relationship (**topological**, **metric** or **directional**) with other objects, from the same category or same IL. The selected objects are presented on the display and table module and might be used by the **grouping** or attributes **query** modules.

3· **Grouping**: create groups of geographical objects according to their descriptive attributes, starting from objects defined in the collection or pre-selected by an attribute query, a spatial query or both. The result is presented only on the display with the color code defined for each group. You can also generate pie chart and histograms for each object.

4· **Table:** displays all the attributes of an object category even from a non-spatial table that has been previously related. Each line of the table presents the set of objects defined by the collection or later selected by an attribute query or spatial query. If you click on the object on the display its corresponding line will be highlighted on the table, and vice-versa. It is possible to define graphical and statistical analysis on the numerical attributes. It is also possible to save the table contents into text files, generate an object collection or a thematic IL of the objects listed on the table.

⇒ **Processing a Query the objects map of cadastral model:**

Windows: #Start – Spring<version><Language><system> – Spring<version> <Language>

Linux: # Command to be typed on the Console (Shell) – # s_spring

MAC: #Dock – Launchpad – Spring <version> < language >

SPRING

SPRING

* Load the database **DF**

* Load the project **Brasilia**

Control Panel

– (Categories | Urban_cadastre)

– (Information Layer | Lots_map)

– (Lines), (Objects)

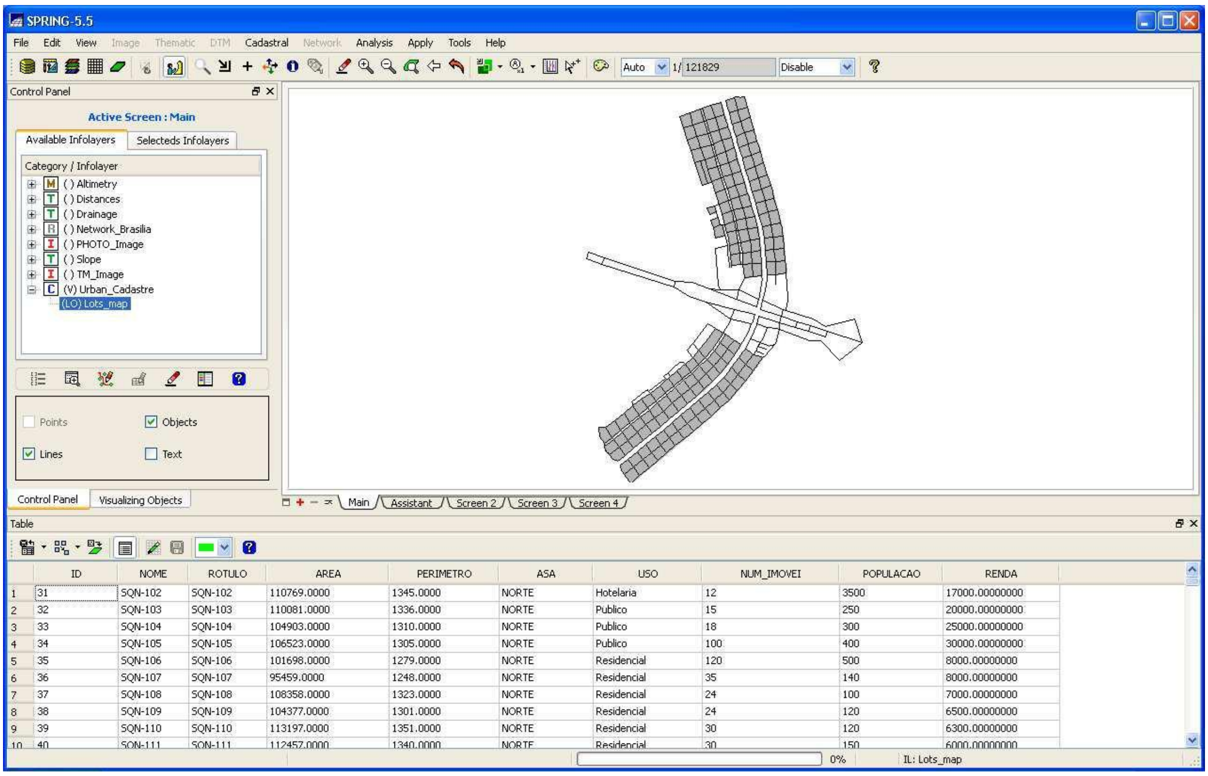
– (Query...) or button

Generating and Selecting Collections

– (Object Categories| Lots)

– (Collections | ALL)

- (Apply) – *displays both the* "**Visualizing objects**" *and* "**Table: Lots**" *windows.*



⇒ **Processing a query on the table module:**
Table

1 • Attribute Table x Visualization Display:

- Select one *object* on the Attribute Table, clicking the left mouse button (LB) over the first column of the desired line (the column that numbers each line, not the ID column). The object will be automatically highlighted on the Visualization Display with the current color (color shown on the top left of the table).

- Select several consecutive *objects* on the attributes Table by dragging with the mouse over several lines on the first column starting at the first desired line. The objects will be automatically highlighted with the current color.

1 • Visualization Display x Attribute Table:

- Select an object, in this case a polygon, on the **Visualization Display** by clicking on it with the LB. The object will be highlighted on the attribute table with the current color.

Check below some features over the columns of a table

⇒ **Changing the color when selecting lines**:

- Click the right button (RB) over the current color, on the top left of the table.

- Select the desired color thru  (button).

- Select other objects you want.

⇒ **Deselecting all selected lines on the table**:

- Click the RB over the column that identifies the crescent numbers of the lines.

- On the **Remove Menu**, select the option: [**All colors**]

1

⇒ **Deselecting the set of selected lines with the same chosen color.**

2- On the  button, choose the color to be removed.

- Press the LF(left button) over the column that identifies (crescent) the number of the lines.

- Select on the **Remove** Menu, the option [**Current Color**].

1

⇒ **Amplifying on the screen the selected object or all of them**

2 The Zoom can be used to facilitate the localization of a selected object on the screen.*

- Click the LB over the column that identifies the number of lines on crescent order.

- On the menu [**Zoom**], select the option [**All colors**] if you want to zoom all selected lines, independently from the color. Choose [**Current color**] if you want only the line where the mouse is.

⇒ **Showing basic statistics for numeric attributes**

** First, deselect all lines as shown above.*

- Click the RB over the attribute button named "**POPULATION**" and select on the sub-menu the option [**STATISTIC**]. ** Valid only for numeric attributes.*

Data Report

** Observe the values presented:*

Number of Samples, Number of Absences, Minimum, Median, Maximum, Total Sum, Average, Standard Deviation and C. of Variation.

- If you wish, click on (Save...) and type a name for the file.

⇒ **Sorting by attributes**

To make it easier to lacate any data, sort by column.

- Click on the RB on the attribute button named "**USE**", for example and press the LF to access the sub-menu.

- Select the option [**Ordering**].

- Select the option [**Ascending**] or [**Descending**].

** The whole table is presented in the selected sorting type.*

⇒ **Hiding a column during table exhibition**

** When the tables are very long (many attributes), it might be useful to hide certain columns.*

- Click the RB over one column that you wish to hide and click [**Hide Column**] on the sub-menu
  ** The selected columns can no longer be seen on the screen*

⇒ **Showing back hidden columns**

- Click on the RB on any attribute button (first line).

- Select the option [**Show Column...**]

Show Columns

- Click on the attribute name on the list to show or hide it.

** Attributes highlighted with a blue color are visible while the others are not.*

- (Close)

Check below some features for graphical analysis on the table lines and columns:

⇒ **Generating a histogram**

** Select one column (only of real or integer types) to generate the histogram (no line can be selected)*

- Click the RB over the desired attribute button (first line), for instance “SCHOOLS”

- Select the option [**Graphic...**]

** The graphic window is presented. To edit the graphic data such as title, color, etc., click on the button*  *, on the upper left corner of the window. If you want enlarge the sides.*

⇒ **Generating a Scatter Plot**

** Select two columns (no line can be selected)*

- Click the LB over the first and second attribute buttons (“SCHOOLS”) and (“POPULAT”).

** Observe that each column is painted with a different color.*

- Click the RB over one of the attribute names (first line), and select the option [**Graphic...**]

** The graphic window is presented. To edit the graphic data such as title, color, etc., click on the button*  *, on the upper left corner of the window. If you want enlarge the sides.*

⇒ **Generating a Pie Chart**

** It is necessary to select at least one line.*

- Select one or more lines with the current color.
- Change the current color and select another set of lines.
- Click the RB over the desired attribute button (first line).
- Select the option [**Graphic...**]

** * The graphic window is presented. To edit the graphic data such as title, color, etc., click on the button* *, on the upper left corner of the window. If you want enlarge the sides.*

NOTE: Before closing the Table module, deselect all lines and columns to avoid interference on the next analysis.

⇒ **Grouping objects by attributes:**

Visualizing Objects

- (Blocks)
- [Edit][Grouping...] or  button.

Grouping Objects: Blocks

- (<u>Attributes</u> | Income)
- (Mode: Equal Step)
- (Number of Parts: 5)
- (Group)
- (Apply)

- *Analyze the outcome on the active screen*

** Try other grouping options*

** **Undo** the grouping before closing the window*

- (Ungroup)
- (Apply)
- (Close)

⇒ **Attributes query:**

Visualizing Objects

- [Edit][Attributes query...]

Attributes Query (**)**

- (<u>Attributes</u> | Income)
- (<u>Operation</u>| >)
- (<u>Values</u> | 2000) - *or any other value you wish to type in.*
- (CR)

** Observe that the logical expressions is presented while you are typing it in*

- (Apply)

- the selected objects are shown on the active screen according to the defined expression.

** **Undo** the expression before closing the query interface:*

- (Cancel) - *as many times as necessary until you clear the **Logical Expression**.*
- (Close)

⇒ **Associating JPEG, GIF, HTML files and URL's to geo-objects.**

SPRING

- Select a polygon on the display by double clicking with the left button. You will see an interface showing the attributes of the associated geo-object.

Attributes: Blocks

- Click the RB anywhere over the Attributes table.
- Select Insert: JPEG/GIF/HTML...

Open File

- Select the desired file.

* Observe on the interface the inclusion of the file.

- Click the RB over the Attributes table.
- Select Insert: URL...

Inserting an URL address

- Type in the desired URL.

* Example: http://www.inpe.br or www.inpe.br
(Apply)

* Observe on the interface the inclusion of the URL associated.

⇒ **Showing JPEG, GIF, HTML files or associated URL.**

Attributes: Blocks

- Click the RB over the line that contains the desired file or URL.
- Select View...

Removing the JPEG, GIF, HTML file or associated URL.

Attributes: Blocks

- Click the RB over the line that contains the desired file or URL.
- Select Delete...

1.4 - Other Query resources

The objective of this exercise is to present other query resources on cadastre objects and networks. We will use a database and project other than **Brasilia**: a cadastral map of urban blocks of a district called "Jardim Brasil" whose descriptive attributes are fictitious. The network of roads is available in another map.

In the next exercises we will analyze the spatial relationships between these blocks and also between blocks and roads. The objects on these maps have the following descriptive attributes:

- **Lots** (Plots of land) = label, owner name (PROPRIET), process type (TIPO), address (END), built area (AREA_C), city hall code for taxes (CODPREF), type of property (TIPO_IMO), property destination (DESTINAC), tax 1 exemption (ISEN_IPTU), Tax 2 exemption (ISEN_TSU), Tax 3 exemption (ISEN_APO) and tax collecting sector (SET_TRIB).

- "**Logradouro**" (Public Areas) = name and label only

Besides these two object tables described above, there is a non-spatial table (**Atri_INSS**) that does not have a direct connection with the polygons from the blocks map. It describes some characteristics of all owners of Jardim Brasil district.

This table has some INSS (a government agency) attributes and a key to the Blocks table through a city hall code. The descriptive attributes of this table are:

- **Atri_INSS** = Owner's code for INSS (COD_P), type of taxation (TRIBUT), real tax value (VALOR_R), total tax value (VALOR_T) and type of action (ACAO).

IMPORTANT: The key connecting tables **Lots** and **Atri_INSS** are the attributes CODPREF e COD_P respectively.

Starting SPRING:

Windows: #Start - Spring<version><Language><system> - Spring<version> <Language>

Linux: # Command to be typed on the Console (Shell) – # s_spring

MAC: #Dock - Launchpad - Spring <version> < language >

SPRING

- [File] [Database...] or button

Database

- (Directory...) select the path

C:\Tutor_10classes\Springdb - Windows

~/Tutor_10classes/Springdb - Linux

- (Database | Urban)
- (Load) Answer Yes in case you have a Database/Project active.

- [File] [Project...] or button

Projects

- (Projects | Jardim_Brasil)
- (Load)

Next, you should do some query exercises:

1.4.1 Query 1

This exercise will answer the following question – ***"Calculate the average area of all urban properties of type 4, with an area greater than 800 square meters"***. Observe that this query only depends on the descriptive attributes (AREA and TYPE) of the object Lots.

1 Query by attributes - Query 1

Control Panel

- (Enable Auxiliary Display)
- (Categories | Cad-Lots)
- (Infolayer | Map_Lots)
- (Lines), (Objects), (Text)
- (Query...)

Generating and Selecting Collections

- (<u>Object Categories</u> | Lots)
- (<u>Collections</u> | ALL)
- (Apply)

** The **Visualizing Objects** and **Table: Lots** windows will show up.*

Visualizing Objects

- Press [Attributes Query...] on the Objects Visualization Window.

Attributes Query

- (<u>Attributes</u> | TIPO)
- (Operation ⇔ =)
- (Show)
- {Values: 4}
- (AND)
- (<u>Attributes</u> | AREA)
- (Operation ⇔ >)
- {Value: 800}
- (Apply)

** Observe on the display the nine objects that satisfy the query expression.*

Visualizing Objects

- Press the button [Table...] on the Objects Visualization Window.

Table: Lot

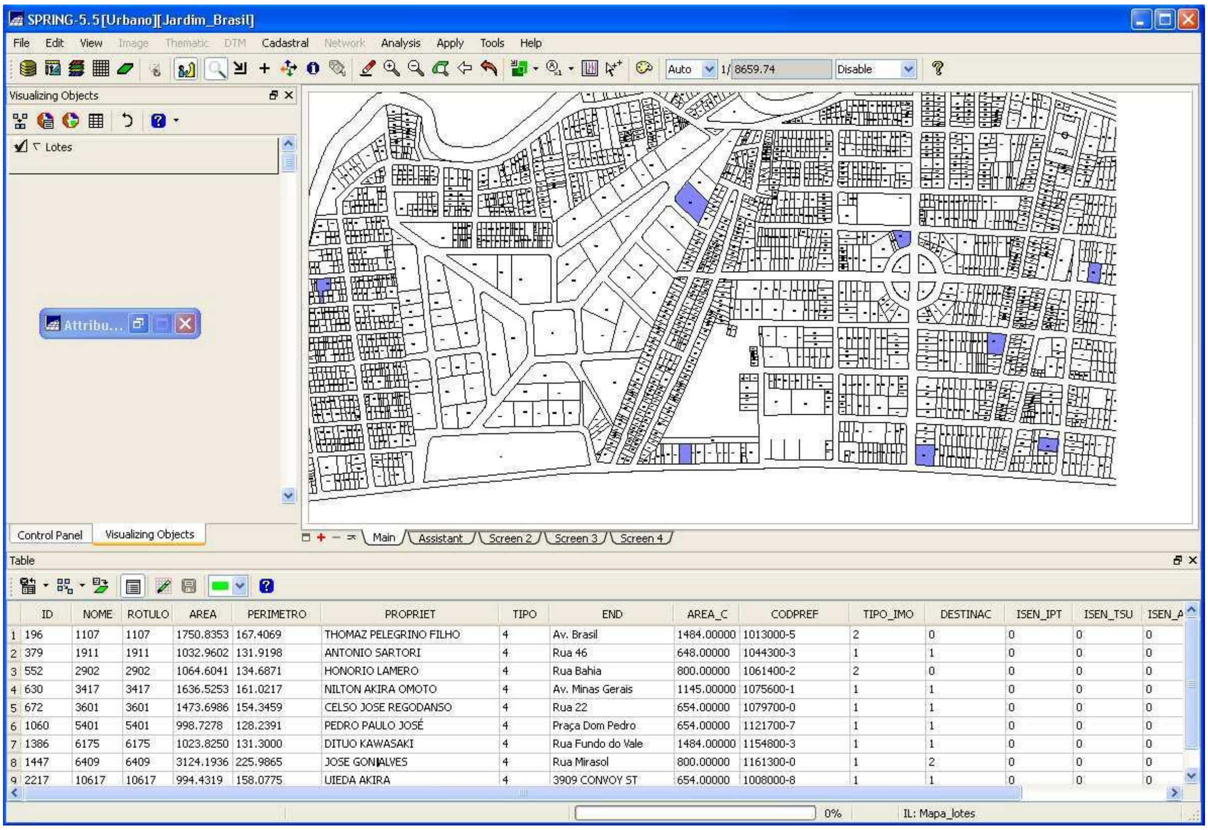
** Click the right button on the attribute name **AREA** and choose operation **Statistics**.*

Data Report

** The searched value is 1455.5335 (square meters). Following data os also available:*

N. of SAMPLES	9		
N. of ABSENCES	0		
MINIMUM	994.4319	MEDIAN	1064.6041
MAXIMUM	3124.1936	TOTAL SUM	13099.801 8
AVERAGE	1455.5335	STD. DEVIATION	653.5178
C. of VARIATION	0.4490		

The following figures present the result of Query 1.



1.4.2 Query 2

The next exercise will answer the following question – “Show a quantile grouping of the AREA attribute in three parts and a pie chart of the land area and built area, of all lots at Antunes Street”. Observe that this query only depends on the attributes END, AREA and AREA_C from the object Lots, where a selection is initially performed (attribute END = “Rua Antunes”) and then the objects are grouped to suit this selection.

IMPORTANT: Before starting this exercise, cancel all and any consult that might had been done before. Click [Execute] [Clear] [All], on the main toolbar Menu. [Reset] [Queries].

Querying by Attribute and Grouping Objects – Query 2

Control Panel

– (Enable | Auxiliary Display)

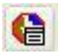
- (Categories | Cad-Lots)
- (Infolayers_| Map_Lots)
- (Lines), (Objects), (Text)
- (Query...)

Generating and Selecting Collections

- (Object Categories | Lots)
- (Collection | ALL)
- (Apply)

** The Visualizing Objects and Table: Lots will show up.*

Visualizing Objects

- Press  [Attributes Query...] on the Objects Visualization Window.

Attributes Query

- (Attributes | END)
- (Operation ⇔ =)
- (Show) **the names of the roads will show up*
- {Values: “Rua Antunes”}
- (Apply)
- (Close)

** Observe on the display **Tables: Lots** the nine objects that satisfy the query expression.*

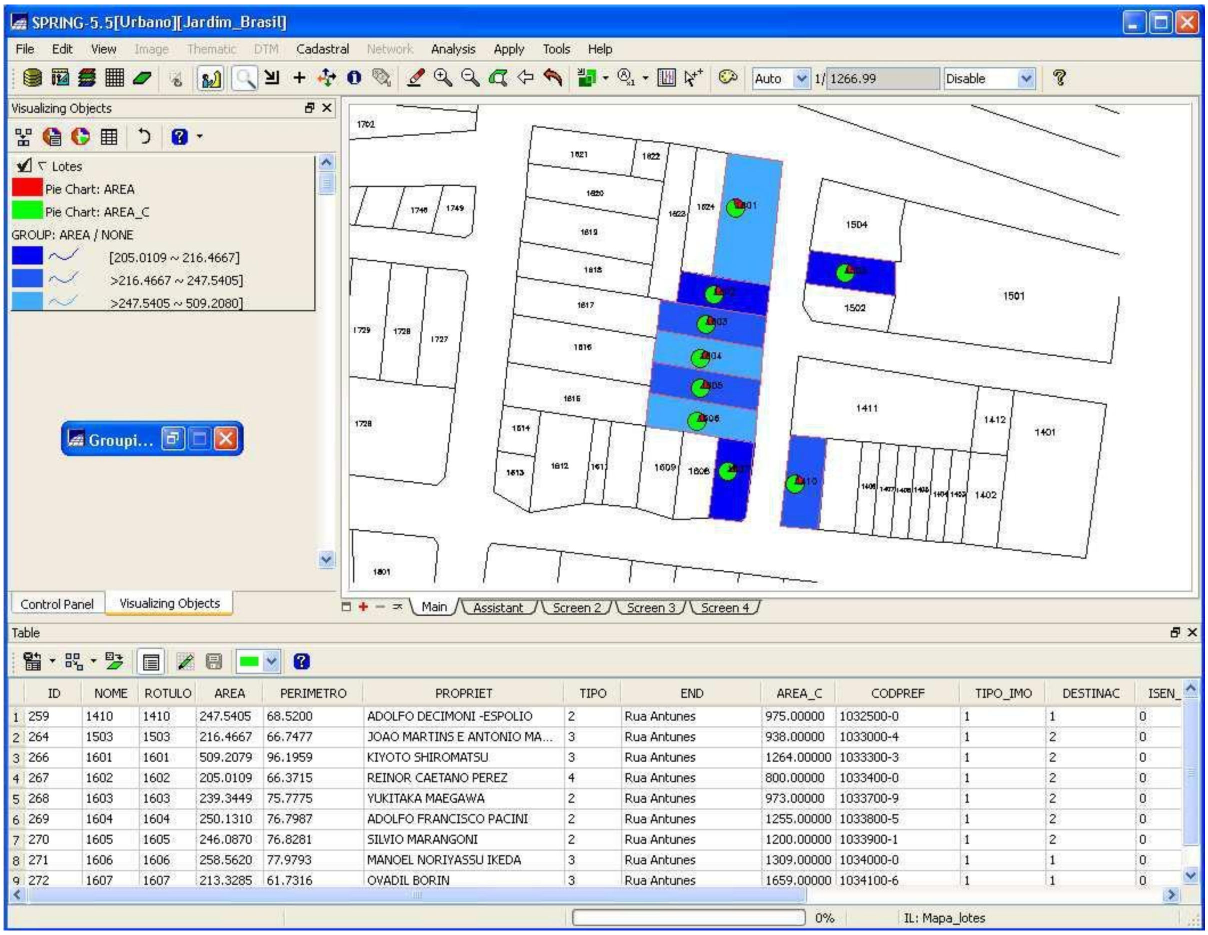
Visualizing Objects

- Press [Grouping...] on the Objects Visualization Window.

Grouping Objects: Lots

- (Mode ⇔ Quantile)
- (Attributes | AREA)
- (Number of Parts ⇔ 3)
- (Color Gradation ⇔ Blue Cyan Green) **or other any color*
- (Group)
- (Apply) **observe the result*
- (Mode ⇔ Pie Chart)
- (Attributes | AREA)
- (Insert)
- (Attributes | AREA_C)
- (Insert)
- (Min. Size: ⇔ 6)
- (Apply)

The figure below presents the result of **Query 2**. Observe on the **Visualizing Objects** window the legend of grouping data and chart.



1.4.3 Query 3

The next exercise will answer the following question – *"Show all lots (plots of land) of type 2 and 3 that are located less than 200 meters from the Brazil Avenue"*. Observe that this query depends only on the attribute "TIPO" (TYPE) and on the spatial relation with other object, in this case "Logradouro"(Public areas). In this query, the order does not matter. Separate the objects lots (plots of land) that are far from the object "logradouro", then the ones of a certain type, or vice-versa.

IMPORTANT: Before starting this exercise, cancel all queries done before [Reset] [Queries].
First of all, create a collection with all paths of Brasil Ave.:

⇒ Defining a collection for Brasil Avenue:

Control Panel

- (Enable | Auxiliary Display)
- (Categories | Public_areas_network)
- (Infolayer | Public_areas_map)
- (Lines), (Objects)
- (Query...)

Generating and Selecting Collections

- (Object Categories | Public_areas)
- {Collection: AvBrazil}
- (Create)
- (Attributes | NOME)
- (Operation ⇔ =)
- (Values ⇔ T) ** to list all values for NAME attribute.*
- (List of values | Av. Brasil)
- (Generate)
- (Apply)

Next, activate the Map_lots to start the query.

⇒ Performing a query by attributes and spatial – Query 3

Control Panel

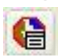
- (Enable | Auxiliary Display)
- (Categories | Cad_Lots)
- (Infolayers | Map_Lots)
- (Lines), (Objects), (Text)
- (Query...)

Generating and Selecting Collections

- (Object Categories | Lots)
- (Collection | ALL)
- (Apply)

** The Visualizing Objects and Table: Lots will show up.*

Visualizing Objects

- Press  [Attributes Query...] on the Objects Visualization Window.

Attributes Query

- (Attributes | TIPO)
- (Operation ⇔ =)
- (Show) – ** wait until all types are presented*
- (Values | 2)
- (OR)
- (Attributes | TIPO)
- (Operation ⇔ =)
- (Values | 3)
- (Apply)

** Observe on the **Table: Lots** window that only types 2 or 3 are shown.*

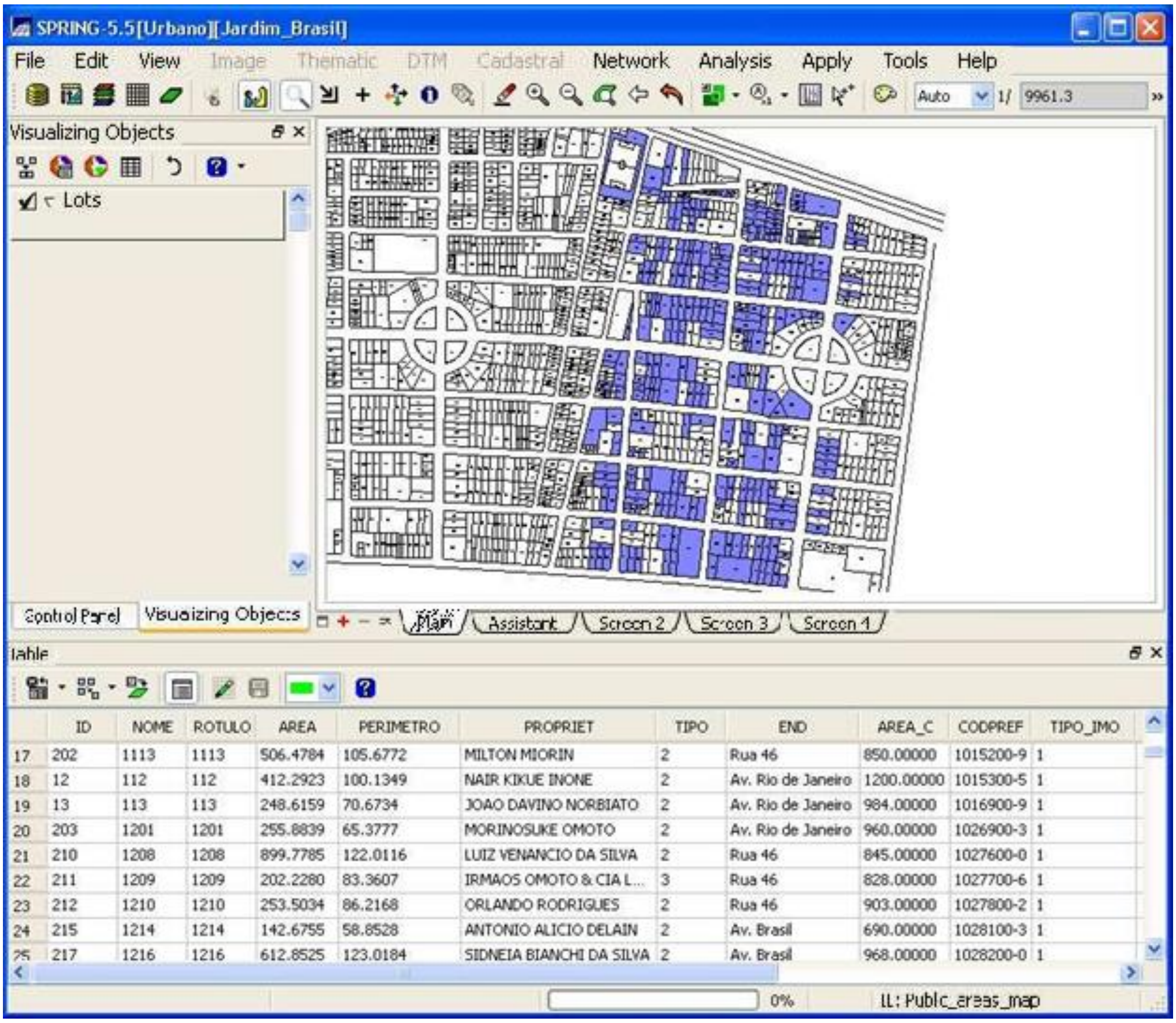
Visualizing Objects

- Press [Spatial Query...] on the Objects Visualization Window.

Spatial Query

- (Operation ⇔ Metric)
- (Distance ⇔ <)
- {Value: 200}
- (Object | Public_areas)
- (Infolayer | Public_areas_map)
- (Collection | AvBrazil)
- (Selection Option ⇔ New)
- (Apply)

The figure below presents the result of Query 3.



NOTE: now try first to perform the spatial query of all objects except 200 of Brasil Avenue and then separate types 2 and 3. The result must be the same.

1.4.4 Query 4

The next exercise will answer the following question – "Show all lots that are neighbors of the one belonging to 'MARIANO CASTRO ALVES' and have a built area greater than 900 square meters" – Notice that this query not only depends on the attributes PROPRIET and AREA_C, but also on the spatial relationship with a certain object.

IMPORTANT: Before starting this exercise, cancel all queries performed before [Reset] [Queries].

⇒ **Defining a collection for Mariano Castro Alves`s property:**
Control Panel
- (Enable | Auxiliary Display)
- (Categories | Cad-Lots)
- (Infolayer | Mapa_lots)
- (Lines), (Objects)
- (Query...)

  Generating and Selecting Collections
  - (Object Categories| Lots)
  - {Collection | Mariano}
  - (Create)
  - (Attributes | PROPRIET)
  - (Operation ⇔ =)
  - (Values ⇔ T) ** to list all values for the NAME attribute*.
  - (List of Values_| MARIANO CASTRO ALVES)
  - (Generate)

  1

⇒ **Performing a query by attributes and spatial – Query 4**
Generating and Selecting Collections
- (Object Categories| Lots)
- (Collection | ALL)
- (Apply)

* Wait for the Table and Visualizing Objects screens.

Visualizing Objects
- Press  [Spatial Query...] on the Objects Visualization Window.

  Spatial Query
  - (Operation ⇔ Topology)
  - (Relationship | Touch)
  - (Object | Lots)
  - (Infolayer| Map_lots)
  - (Collection | Mariano)
  - (Selection Option ⇔ New)
  - (Apply)

  ** Observe on **Table: Lots** that there are (3) lots that are neighbors to 'MARIANO CASTRO ALVES'.*

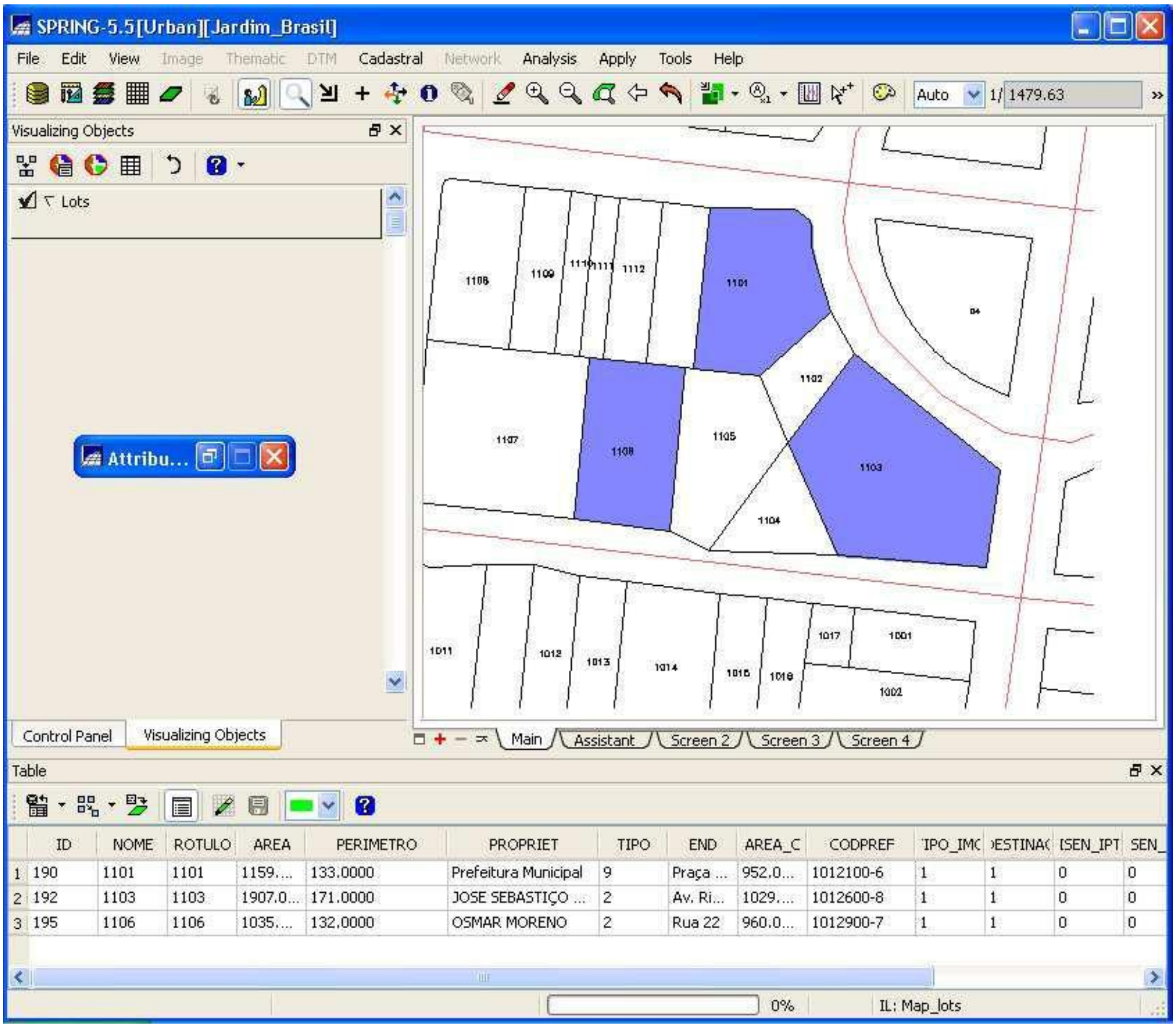
Visualizing Objects

- Press [Attributes Query...] on the Objects Visualization Window.

Attributes Query

- (Attributes | AREA_C)
- (Operation ⇔ >)
- {Value : 900}
- (Apply)

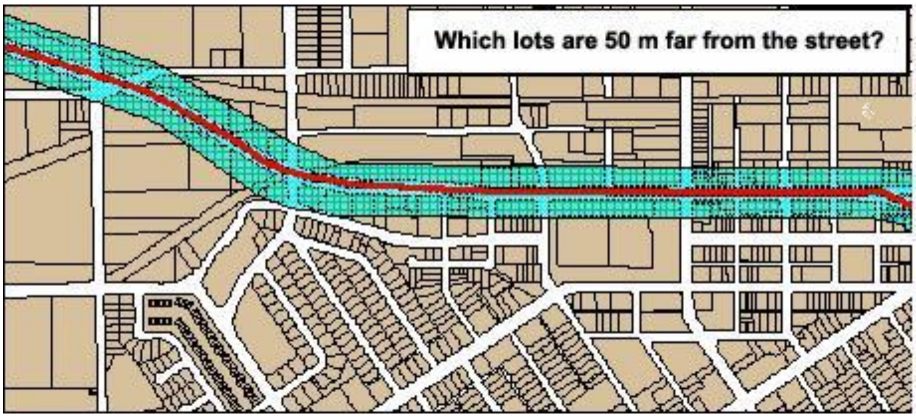
The figure below presents the result of Query 4. Observe that only three "lots" satisfy the new query.

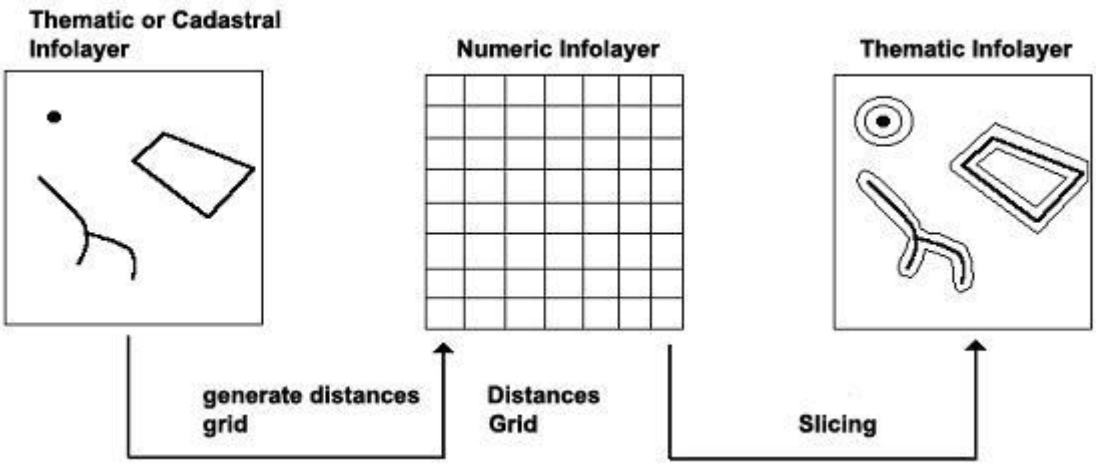


2 – Spatial Analysis

2.1 – Distances Map (buffer)

A distance map is a type of proximity analysis (measuring the distance between objects) that presents regions with specified widths (distances) around one or more elements of a map. Check the illustration below:





Check below the procedures for a distance map:

11. Identify the element (point, line or polygon) that will be used to generate the distance grid. You can use either a Thematic or Cadastral Infolayer;

22. Generate a numeric grid with distance values around the selected element;

33. Slice the grid considering the desired distances.

1

⇒ Generating a Distance Map

Windows: #Start – Spring<version><Language><system> – Spring<version> <Language>

Linux: # Command to be typed on the Console (Shell) – # s_spring

MAC: #Dock – Launchpad – Spring <version> < language >

SPRING

* *Load the database **Course***

* *Load the project **Brasilia***

– Create a category of numeric model named Grid_dist

Control Panel

– (Enable ⇔ Auxiliary Display)

– (Categories | Drainage)

– (Infolayers | Rivers_map) IL created in Class 7.

– (Lines), (Classes)

– (Draw) or – [Apply] [Draw] or button  on the main menu.

SPRING

– [Thematic][Buffering...]

Distance Map

– (Input ⇔ Vector Map)

– (Selection ⇔ Class)

– *Click on the polygon corresponding to the lake on the screen*

– (Category...)

– *Select the output numeric category* **Grid_dist** *created before.*

– {PI: dist-comu}

– {X(m): 200}, {Y(m):200}

– (Apply)

– *Visualize the grid*

– *Slice the grid. Check the slice example given in Class 8.*

2.2 – Area Calculation

⇒ Doing an area calculation

Windows: #Start – Spring<version><Language><system> – Spring<version> <Language>

Linux: # Command to be typed on the Console (Shell) – # s_spring

<Version><Language>

MAC: #Dock – Launchpad – Spring <version> < *language* >

SPRING

** Load the database **Course***

** Load the project **Brasilia***

- Visualize the thematic plan **Soils_map** from the **Soils** category
- [Thematic][Measurements of Classes...]

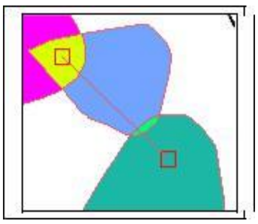
  Measurements of Classes

  - (Thematic Image), (Vector Map)
  - (Unit ⇔ ha)
  - (Apply)
  - (Save...)
  - *Select file to save the outcome*

2.3 – Measurements

Distance between two points

The distance between two points is calculated along a line, using the mouse cursor.



⇒ *Measuring the distance between two points*

Windows: #Start – Spring<version><Language><system> – Spring<version> <Language>

Linux: # Command to be typed on the Console (Shell) – # s_spring

MAC: #Dock – Launchpad – Spring <version> < language > <Version><Language>

SPRING

** Load the database **Course***

** Load the project **Brasilia***

- Visualize any Information Layer (an image, for example)
- [Tools][Measurements...]

  Measurements

  - (Type ⇔ Editing)
  - (Option ⇔ Distance)
  - (Unit ⇔ Km)

- Click in any two points on the active screen and you will see the distance value on the bottom of the Measurements window.

- (Close)

Polygon Area or Perimeter

The **Measurements** window allows us to calculate the area or perimeter of any polygon represented on thematic maps by classes or cadastral maps by objects.

⇒ **Performing the measurement**

Windows: #Start - Spring<version><Language><system> - Spring<version> <Language>

Linux: # Command to be typed on the Console (Shell) - # s_spring

MAC: #Dock - Launchpad - Spring <version> < language >

<Version><Language>

SPRING

* *Load the database* ***Course***
* *Load the project* ***Brasilia***

- Visualize any Information Layer (Rivers_map, for example)
- [Tools][Measurements...]

Measurements

- (Type ⇔ Indication)
- (Option ⇔ Area/Perimeter)
- (Unit ⇔ ha)

- Click on a polygon on the active screen and you will see the area and perimeter values on the bottom of the Measurements window.

- (Close)

2.4 - Cross Tabulation

The cross tabulation operation allows the calculation of the area of the intersections between the classes of two thematic ILs, in raster format preserving the same characteristics: horizontal and vertical resolution, number of lines and columns ("pixels") and same terrain coordinates.

The cross tabulation compares classes from two Information Layers, establishing the distribution of their intersections. The results are depicted in two-dimensional tables.

⇒ ***Performing a cross tabulation calculation***

Windows: #Start – Spring<version><Language><system> – Spring<version> <Language>

Linux: # Command to be typed on the Console (Shell) – # s_spring

MAC: #Dock – Launchpad – Spring <version> < language >

SPRING

* *Load the database **Course***
* *Load the project **Brasilia***

– It is not necessary to have visible Information Layer. Activate the thematic IL **Soils_map** from the category **Soils**.

– [Thematic][Cross Tabulation...]

Cross Tabulation

– (Intersection Plan...)

Categories and ILs

– (Categories | Land_Use)

– (Information Layer | Mapa_uso)

– (Apply)

Cross Tabulation

– (Apply)

* *The outcome is presented on the report window. Click on Save to save the presented data defining a file name.*

3 – LEGAL

A LEGAL program is composed by sentences (command lines) that are structured in three parts: declarations, instantiations and operations.

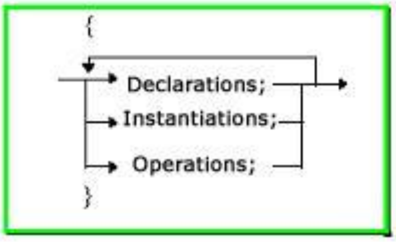
Declaration: this is where the work variables are defined. Each variable must be explicitly declared with a name and associated to a category within the conceptual scheme.

Instantiation: this is where existent data are recovered from the database or a new Information Layer is created. Afterward, this new IL can be associated to the result of LEGAL operations.

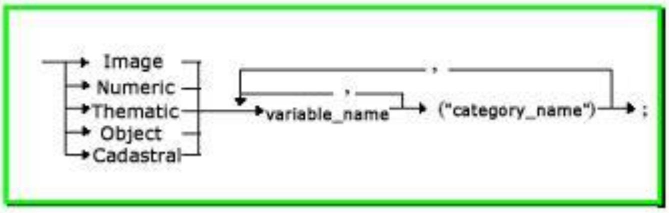
Operation: this is where the maps algebras are performed.

In LEGAL, each sentence might include **symbols** (for example, '{', '(', ';', ','), **operators** (for example, '+', '*', **'&&'**, '||', '<', '<=', '!='), **reserved words** (for example, **New**, **Thematic**, **Name**, **ResX**), **variable**

names and **data names** (ILs). IL names, category and thematic classes must be written between double quotes ("").



Declaration

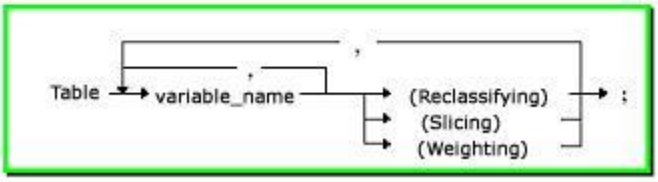


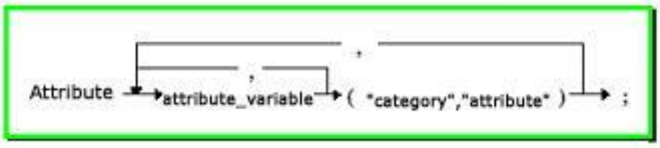
Examples:

Image band3, band4, ivdn("LANDSAT");

Thematic soil ("Type_Soil"), geo("geology");

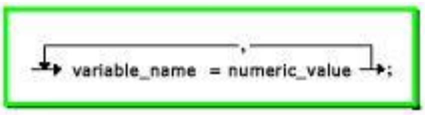
Numeric alti1 ("ALTIMERIA");





Example:

Attribute values ("LOTS","IPTU");

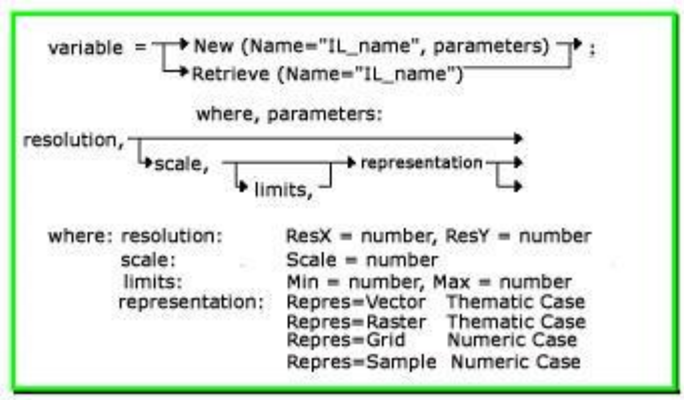


Example:

pi=3.14, d=1.234454;

Instantiation

The language allows the creation of new information layers to store results of expressions that involve other representations, using the reserved word New, or even the retrieval of previously created ILs using the reserved word Retrieve.



Example of IL retrieval:

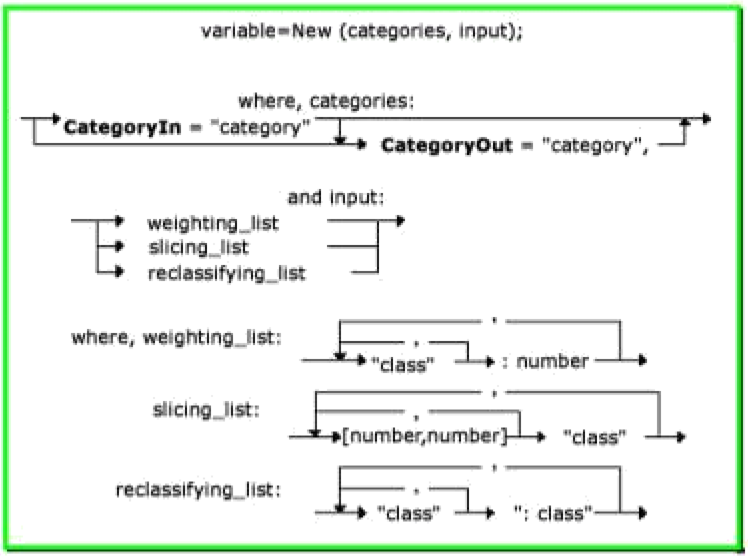
theme = Retrieve (Name = "BasinHydrographic");
alti = Retrieve (Name = "Altimetry"); ima =
Retrieve (Name = "TM4");

Example of IL creation:

soil = New (Name = "Soils_A", ResX=50, ResY=50,
Scale=100000, Repres = Vetor);

alti = New (Name = "Altimetry", ResX=50, ResY=50,
Scale = 1000, Min=0, Max=100);

ima = New (Name = "ImageTM_Res", ResX=30, ResY=30);



Example of reclassification table:

group = New(CategoryIn = "Vegeta", CategoryOut =
"Vegeta", "Da": "AlluvialForest",
"Db", "Ds1", "Ds2", "Ds4", "Dm": "OmbrophilousForest",
"sd", "sp", “sA” : "Savannah",
"Pfm", "Pa", "Pah": "PioneerFormation",
"Ap": "Forest");

Example of slicing table:

group = New(CategoryOut = "Vegetation",
[0.0, 0.2]: "Forest",
[0.2, 0.45], [0.8, 1.0]: "Mata_galeria",
[0.45, 0.8]: "Savannah");

Example of weighting table:

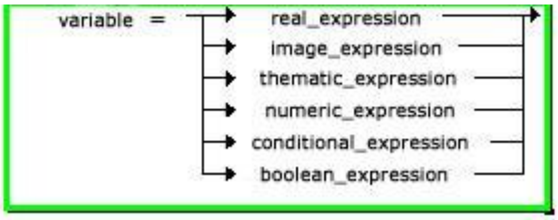
weight1 = New(CategoryIni = "Vegetation",
"Forest": 0.2,
"AlluvialForest", “Forest”: 0.43
"Savannah"): 0.456);

Operation

After the creation and instantiation of variables we present the definition of valid actions on them. In LEGAL, the results of actions are invariably represented by operations. In an operation, a variable receives the result of the processing of expressions that include language operators that act upon variables that have been previously declared and instantiated in the program.

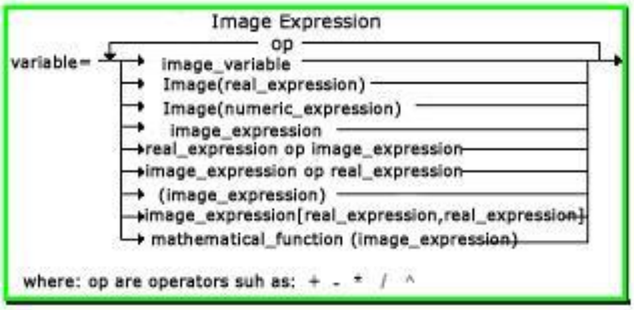
The results of action in Legal are always represented by operations. In an operation a variable receives a result of expressions being processed involving operators that take action over variables previously declared and instantiated.

The figure below shows the possible relationships inside operations.



The arithmetic operators '+', '-', '*', '/' and '^', as well as mathematic functions (sin, tangent, etc.) are taken as punctual or local acting over each element of image matrix representations or numeric grids, or over neighbor elements located in relation to a reference element.

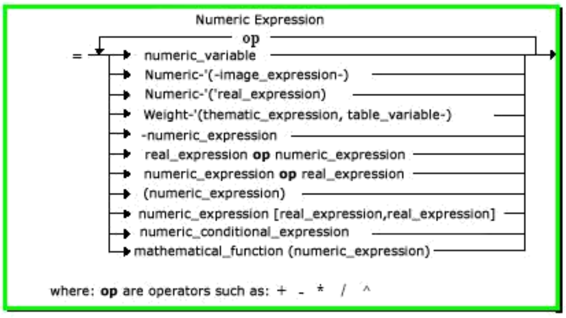
Image Expression



Example of image expressions:

```
ima1 = Image (grid1);
ima3 = ima2 + 20;
res_ima1 = abs(sin(ima1)- 255);
```

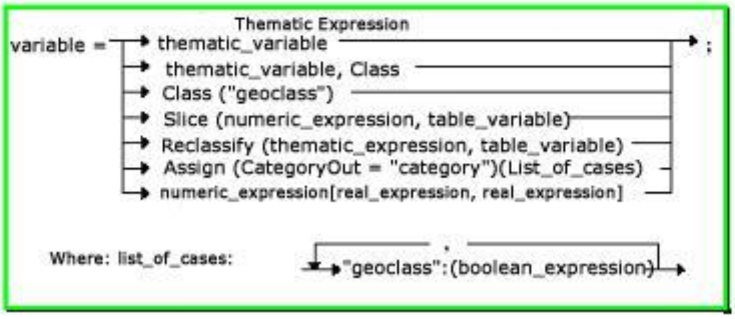
Numeric Expression



Example of numeric expressions:

ph_fe1 = Numeric (band_spot2);
sum_grid = (grid_soil + grid_decl)/2;
grid_sin = sin(grid1);

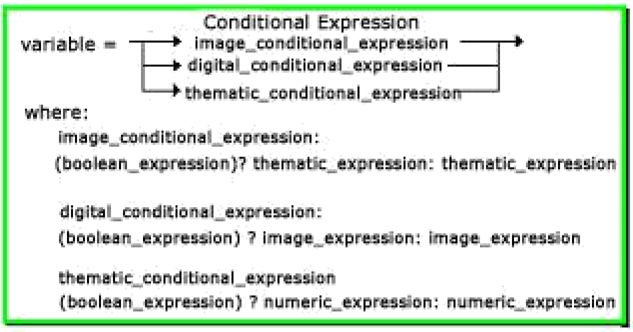
Thematic Expression



Example of thematic expressions:

cl_decliv = Slice(decliv,tab_decliv); deforest=
Reclassify (coverage, tab_recl); suitability=
Assign (CategoryOut = "Suitability")
{
"Good": (soil.ClassI == "Clay" &&
decliv.Class == "O-3"),
"Inapt": (soil.Class == "Sand" &&
decliv.Class == ">8")};

Conditional Expression



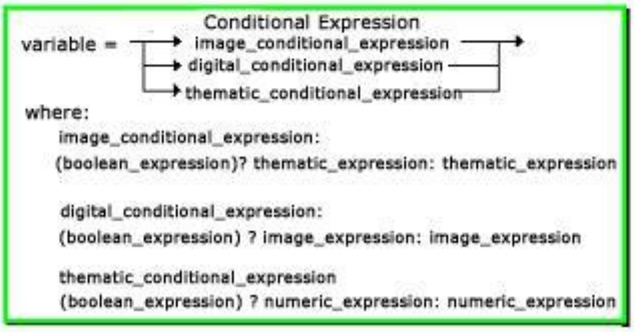
Example of conditional expression:

Imag_out =(ta.Class == “forest”) ? Imagem (TM5): 0;

Boolean Expression

Boolean expressions include all kinds of expressions. The resulting value of such an expression must be TRUE or FALSE, and can be obtained from the comparison of image pixels or grid values by the use of the operators ‘<’, ‘>’, ‘<=’, ‘>=’, ‘==’ and ‘!=’ or by the comparison between classes of Thematic IL’s by the operators ‘==’, and ‘!=’. Up to 40 ILs may be used simultaneously.

Boolean expressions can still be combined by means of operators ‘**&&**’ (logical AND, intersection), ‘||’ (logical OR, union) and ‘!’ or ‘~’ (denial, complement).



3.1 – Editing and Executing a LEGAL program

The program edition can be done in any simple text editor (ASCII format) or even inside SPRING.

⇒ *Editing and running a LEGAL program*

Windows: #Start - Spring<version><Language><system> - Spring<version> <Language>

Linux: # Command to be typed on the Console (Shell) - # s_spring

MAC: #Dock - Launchpad - Spring <version> < language >

<Version><Language>

SPRING

* *Load the database **Course***
* *Load the project **Brasilia***

- Create the thematic category **Suitability**, with the classes **good**, **medium** and **low**.

- [Analysis][LEGAL...]

Algebra

- (Directory...: C:\Tutor_10classes\ Legal_Programs) - Windows
  ~/Tutor_10classes/ Legal_Programs - Linux
  ~/Tutor_10classes/ Legal_Programs - MAC
- {Name: suitability}
- (Create...)

Model Editor

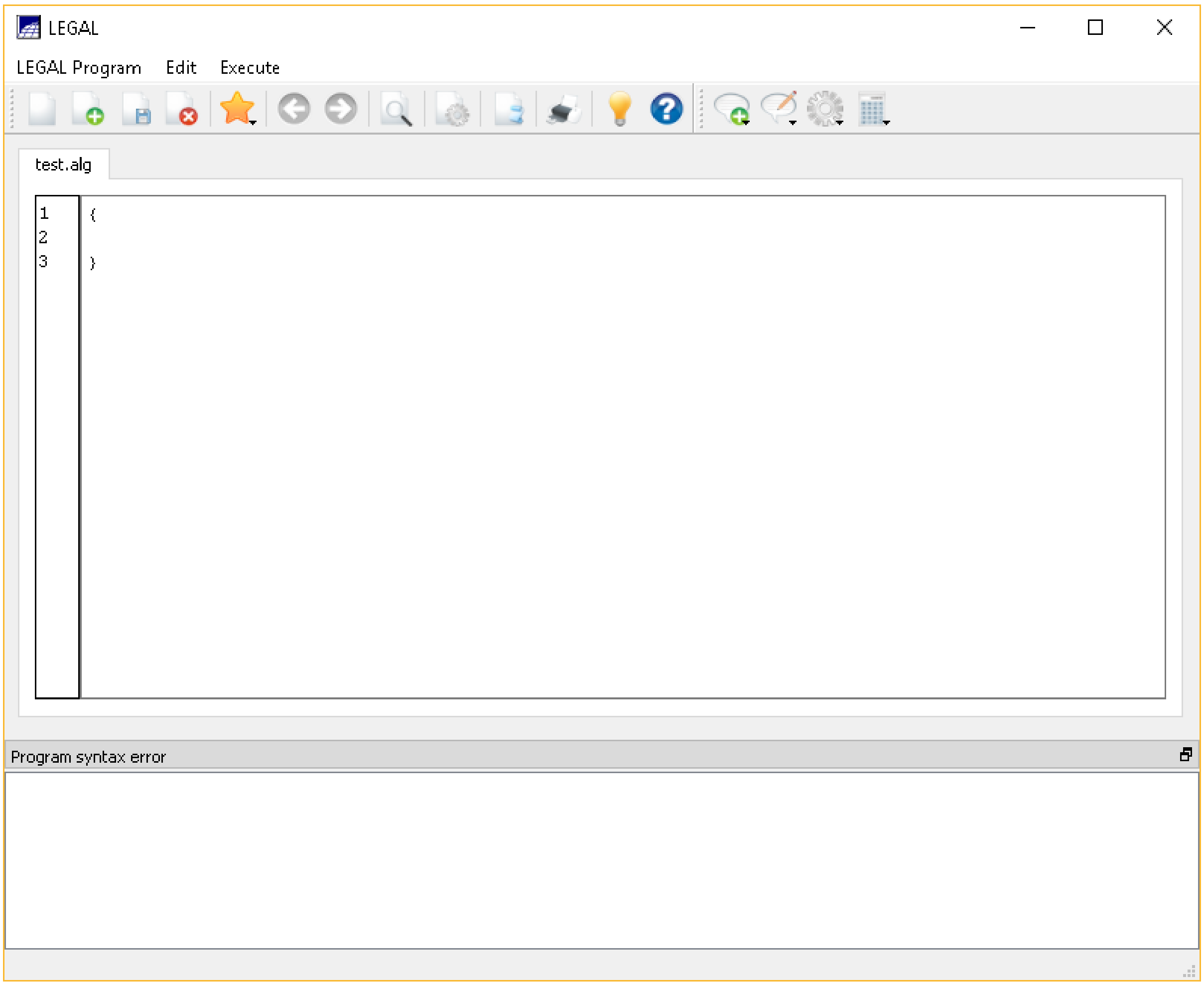
- {PROGRAM: - t*ype the program presented below*}
- (Save)

Algebra

- (Apply)

* *If there is a syntax error the user will be informed and after that the editor window will be shown again. Make the corrections, save and run (“**Apply**”) again.*

In the figure below, shows the editor of LEGAL.




```
//Example – the crossing of 2 thematic plans
{
//Defining variables and their categories
Thematic soil(“Soil”), slop(“Slope”), apt(“Suitability”);

//Recovering information layers
slop=Retrieve (Name = “dec”);
soil=Retrieve (Name = “soi”);

//Creating a new information layer
suit=New(Name=“Suitability”, ResX=200, ResY=200, Scale=100000);

//Defining the relations between classes
suit = Assign (CategoryOut = “Suitability”)
        {
        “good”: (soil.Class == “Cd1“ && slope.Class == “0-3“),
        “medium”: (soil.Class == “Cd1“ && slope.Class == “3-8“),
        “low”: (soil.Class == “Cd1“ && slope.Class == “8-20“)
        };
```

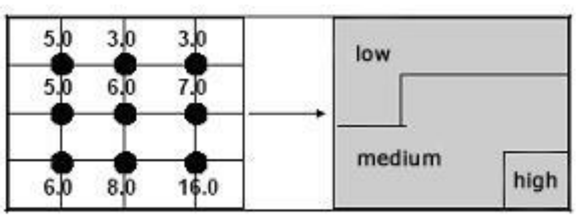
```
}
```

Check some other examples of programs below:
Example of slicing using some soil data

```
{
Numeric grd ("Numeric_slope");
Thematic fat ("Slope");
Table fati (Slicing);

grd = Retrieve(Name="decliv-30x30");
fat = New(Name= "decli_fat_30x30", ResX=30, ResY=30, Scale=100000);

fati=New(CategoryIn="Numeric_slope",CategoryOut =
            Slope", [0.0,3.0] : "A-0a3",
            [3.0, 8.0] : "B-3a8",
            [8.0, 12.0] : "C-8a12",
            [12.0, 20.0] : "D-12a20",
            [20.0, 45.0] : "E-20a45",
            [45.0 ,90.0] : "F>45",
            [90 , 900] : "F>45");
            fat = Slice(grd,fati);
}
```

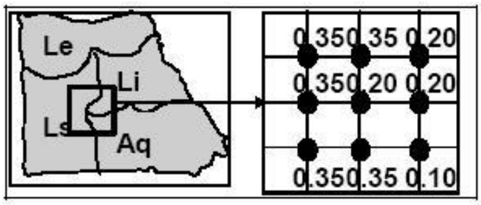


Example of Weighting using soil data:

```
{
Thematic soil("Soils");
Numeric solero ("Erosion");
Image solima ("Image");
Table pond (Weighting);

soil = Retrieve (Name = "solos");
solero=New(Name="soloPond",ResX=30,ResY=30,Scale=100000,Min=0,Max=100)
; solima = New (Name ="soloPond", ResX=30, ResY=30);
pond = New (CategoryIni = "Soils", CategoryOut =
            "Erosion", "LV1" : 0,
            "AR" : 50,
            "LV2" : 100,
            "PV1" : 150);
```

```
solero = Weight (solo,pond);
solima = Image (solero);
}
```



Example of reclassification using soil data:

```
{
Thematic uso, recl ("usoatual");
Table juntar (Reclassificacao);

uso = Retrieve (Name="usoatual");
recl = New (Name= "recla_alg", ResX=30, ResY=30, Scale = 100000);
juntar = New (CategoryIn = "usoatual", CategoryOut = "usoatual",
            "mata": "Veg_nat",
            "cap+mato": "Veg_nat",
            "capoeira" : "Veg_nat",
            "acude": "agua",
            "cultura": "agricola",
            "capoeira+cult" : "agricola",
            "pasto": "pastoril",
            "pasto+pastosujo": "pastoril",
            "pasto+cult": "pastoril",
            "pasto": "pastoril",
            "pastosujo" : "pastoril",
            "eucalipto" : "silvicola",
            "escola" : "urbano",
            "terraco" : "terraco",
            "erosao" : "erosao");

recl = Reclassify (uso, juntar);
}
```

Example: converting a DN-Image to a Image-Apparent reflectance using the general equation proposed by Markham & Baker (1987):

```
{
Image IV255, ima3, ima4, tm3, tm4,IV2("Image_TM");
```

```
Numeric re3("Numeric"), re4("Numeric"), IVNAl ("Numeric") ;

tm3 = Retrieve (Name="tm3_030895");
tm4 = Retrieve (Name="tm4_030895");

re3=New(Name="tm3_refl",ResX=30,ResY=30,Scale=100000,Min=0,Max=50);
re4=New(Name="tm4_refl",ResX=30,ResY=30,Scale=100000,Min=0,Max=50);

IVNAl=New(Name="IV_Refalg",ResX=30,ResY=30,Scale=100000,Min=0,Max=50);
IV2 = New (Name = "IVDN_RefAlg", ResX = 30, ResY = 30); IV255 = New (Name =
"IV255", ResX = 30, ResY = 30);

ima3 = New (Name = "ima3reflect", ResX = 30, ResY = 30);
ima4 = New (Name = "ima4reflect", ResX = 30, ResY = 30);

c1 = 1.0119;
c2 = 0.607735;

re3=Digital((PI*(C1^2)/155.7*C2)*((tm3/255)*(20.43-(-0.12))+(-0.12)));
re4=Digital((PI*(C1^2)/104.7*C2)*((tm4/255)*(20.62-(-0.15))+(-0.15)));

ima3= Image (re3 * 255);
ima4= Image (re4 * 255);

IVNAl = (re4-re3)/(re4+re3);
IV2= Image ((re4-re3)/(re4+re3));
IV255 = Image (255*((re4-re3)/(re4+re3)));
}
```

Example of the vegetation index from a TM Image:

```
{
Image tm3, tm4, viimg("Image_TM");
Digital ndvi("Numeric");
Thematic veget("Vegetation");

tm3 = Retrieve (Name = "tm3_86");
tm4 = Retrieve (Name = "tm4_86");

viimg = New (Name = "Vegetation", ResX = 120, ResY =
120); viimg = 40*((tm4-tm3)/(tm4+tm3))+64;

Table slice(Slicing);
slice = New(CategoryOut = "Vegetation",
            [0.0,0.2] : "Non_forest",
            [0.2,0.5] : "Transition",
            [0.5,1.0] : "Forest");

veget = New(Name="SoilCoverage",ResX=120,ResY=120,Scale=250000);
veget = Slice(Digital((tm4-tm3)/(tm4+tm3)), slice);
```

```
}
```