Simple Geographical Analysis Language

Loganathan Vijay Sambandhan(CVN)
lvs2116

Introduction

The Simple geographical language is built to easily create simple Objects in a Geographical space, and perform simple spatial analysis like nearest neighbor, objects within a boundary. Using the Language real world features like Schools, Roads County Boundary can be created and the spatial queries about their whereabouts in relation to other features can be answered .

Components of the language

The language will support only the Geographic Coordinate system. All features should have their Coordinates in latitude longitude Decimal Degrees.

There are three kind of spatial objects which can be created,

Point: has just a Latitude Longitude.
Poly-line: will be a collection of points.
Polygon: will also be a collection of Points but will form a closed Object.

Layer: All objects will be part of a Layer and a layer can have only one kind of Object. For example Once Schools in a State are defined as Point features they cannot have Poly-lines and Polygons.

Derived Components : these components will be calculated based on the Features entered.

Minimum Bounding Rectangle(MBD) : each feature has an Minimum Bounding Rectangle associated with it. A minimum bounding rectangle is created the Maximum Coordinates of an Feature.

Length: All poly-lines will have a Length attribute calculated.

Area and Perimeter: All polygons will have an area and perimeter length calculated.

The flow of a Typical program

Load Features in layers like for example Schools (Points), Roads(Polyline), Counties(Polygon), Hospitals(points), Railway Tracks(Polyline). The program will calculate values like MBD

Once all the features are created the Program can answer queries like,

- How far is the Closest road to a School?
- How many schools are there in a given County?
- What is the total length of Rail in a County?
- All Hospitals which are within 50 Length Units of a School.

Sample Code

```
Layer school[Point]
// Loads three Schools
school Load [-79.78 48.32] [-78.78 48.32][-69.78 58.32]

// Loads two roads
Layer road[Polyline]
road Load [-79.78 48.32, -79.76 48.35, -79.86 48.35, -79.76 48.85][-69.78 58.32, -69.76 58.35, -69.86
58.35, -69.76 58.85][-47.08 63.22, -47.76 63.35, -47.86 63.35, -47.76 63.85]

// Loads two Counties
Layer county[Polygon]
county Load [-79.78 48.32, -79.76 48.35, -79.86 48.35, -79.76 48.85][-69.78 58.32, -69.76 58.35,
-69.86 58.35, -69.76 58.85]

// Ask which is the nearest school to the second road.
Nearest road[1] school

// Ask which is schools are within the first county
Within county[0] school
```

In code Latitude and Longitude are separated by a space, and coordinates within a feature are separated by a comma.

The values are indexed in the layer Collection in the order in which they are loaded. The indexing for the features begins at 0.

For example if you access the second feature in School the coordinates will be (-78.78 48.32). And while analyzing the features further they will be accessed based on their index.

Comments will be denoted by //.

All Coordinates will be stored in a Decimal Number which can have precision up to 10 decimal places to support granularity in Data.