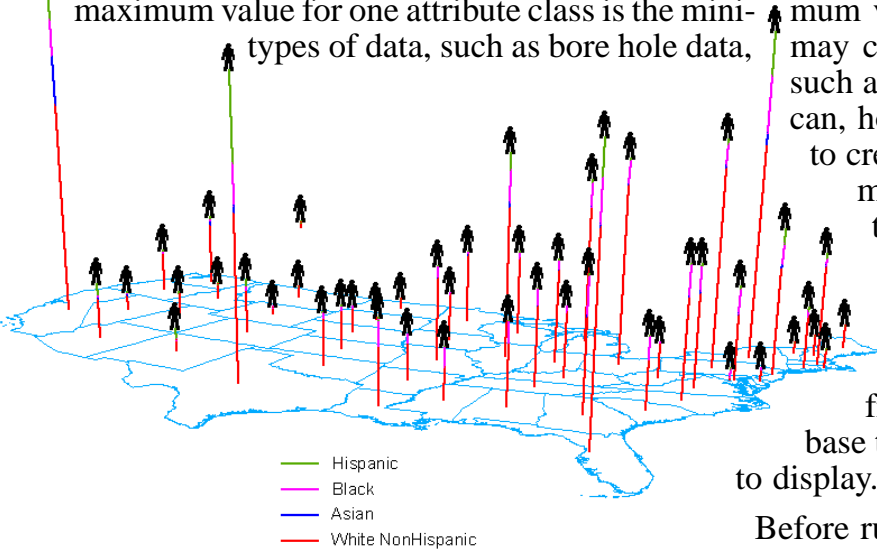# Sample SML Script
Creating Cumulative Data

Display of vector points as stalks or roots that convey information about a single attribute does not require any special data preparation. Display of multiple attributes for a single vector point, however, requires that the point have multiple records attached and that one of the fields in those records presents the desired attribute in a cumulative fashion. In this context, cumulative means that the maximum value for one attribute class is the mini- mum value for the next attribute class. Some types of data, such as bore hole data, may come in this form. Other types of data, such as population data, generally do not. You can, however, run an SML script (see reverse) to create a new table with records and attachments in the form required for style by attribute using a color stack (see the plate entitled *Vector Points on Stalks* if you are not familiar with this feature). This script can be easily adapted to other data sets by simply changing the table and field names to reflect those in the database table associated with the points you want to display.

Hispanic
Black
Asian
White NonHispanic

original table

For every point, the new table should contain a record for each attribute value you want displayed on the stalk or root. The desired attribute values here are White NonHispanic (a computed field), Asian, Black, and Hispanic. The original table may contain many more fields than you want to use for styling.

The relationships between the three tables required for stalk or root display by attribute using the color stack feature are circled in the Database Editor at the right.
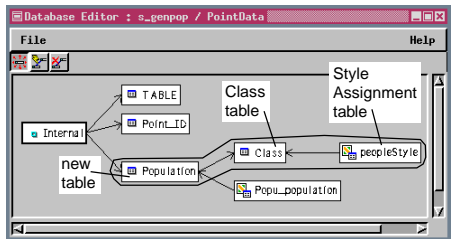

table generated by script

Before running the SML script, you should have set up a class table that lists all attribute values for the field you intend to use to style the stalks or roots. This table need have only one field, the class field, which is a primary key. After the new table is created by the script, you need to relate it to the existing class table. The field that contains the class information in the new table (here called *people*) needs to point to the primary key field in the Class table. You can establish this relationship either by choosing Edit Relations from the Make Table/Form icon for the database or by choosing Edit Definition for the newly generated table. The Key Field selected in the Vector Object Key Attribute Style Assignment window (Vector Layer Controls / 3D panel / Line Style: By Attribute / Specify) also needs to be the primary key field in the Class table. You need to have the relationships set correctly to get the stalk and root styles you assign by attribute to display.

After the V6.50 release, an additional, simpler styling method that allows you to pick multiple fields from a single record and specify their order and assigned color in the stack will be developed. The method described on this page will still be necessary if you want to assign styles by attribute to a point stalk or root, rather than simply assign colors to each attribute value. Assigning styles by attribute with multiple attached records will allow you to also incorporate line patterns as part of the stalk or root style, which will not be available with the yet to be developed feature.

Creating Table with Data in Cumulative Form (cumul.sml)

```
clear();  #clear console

GetInputVector(Vect);

class DATABASE database;
class DBTABLEINFO tableinfo;
class DBTABLEINFO populationinfo;
numeric numPoints;
numeric index;
numeric record;
numeric numRecords;
numeric value;
numeric population;
Array records[1];

database = OpenVectorPointDatabase(Vect);

# Create new table

populationinfo =
     TableCreate(database,"Population","Created by SML
     script");
TableAddFieldInteger(populationinfo,"population",12);
TableAddFieldString(populationinfo,"people",16,16);

# Open old table

tableinfo = DatabaseGetTableInfo(database,"TABLE");

numPoints = NumVectorPoints(Vect);

for i=1 to numPoints {
        numRecords =
TableReadAttachment(tableinfo,i,records,"point");
        if (numRecords <= 0) continue;
        record = records[1];

        # Write White Non Hispanic population
        value =
TableReadFieldNum(tableinfo,"WhiteNonH",record);
        population = value;
        index =
TableNewRecord(populationinfo,population,"WhiteNonH");
        records[1] = index;
        TableWriteAttachment(populationinfo,i,records,1);

        # Write Asian population
        value =
```

```
TableReadFieldNum(tableinfo,"ASIANPOP",record);
        population += value;
        index =
TableNewRecord(populationinfo,population,"Asian");
        records[1] = index;
        TableWriteAttachment(populationinfo,i,records,1);

        # Write Black population
        value =
TableReadFieldNum(tableinfo,"BLKPOP",record);
        population += value;
        index =
TableNewRecord(populationinfo,population,"Black");
        records[1] = index;
        TableWriteAttachment(populationinfo,i,records,1);

        # Write Hispanic population
        value =
TableReadFieldNum(tableinfo,"HISPPOP",record);
        population += value;
        index =
TableNewRecord(populationinfo,population,"Hispanic");
        records[1] = index;
        TableWriteAttachment(populationinfo,i,records,1);


        }


printf("Script Ran to Completion");
```